LinuxCNC V2.8.1, 2020-12-01

Índice general

I Contenido 1

II Acerca de LinuxCNC 2

1. Introduccion 3

2. LinuxCNC History 4

2.1. Origen 4

2.2. Cambio de nombre 5

2.3. Información adicional 5

III Usando LinuxCNC 6

3. Informacion General 7

3.1. Prefacio para el usuario 7

3.2. Introduccion para usuarios de LinuxCNC 8

3.2.1. Como funciona LinuxCNC 8

3.2.2. Interfaces graficas de usuario 10

3.2.3. Paneles de control virtual 14

3.2.4. Idiomas 16

3.2.5. Modos de operacion 16

3.3. Conceptos importantes para el usuario 16

3.3.1. Control de Trayectoria 16

3.3.1.1. Planificacion de trayectoria 16

3.3.1.2. Seguimiento de ruta 17

3.3.1.3. Programacion del planificador 17

3.3.1.4. Planificacion de movimientos 18

3.3.2. Código G 18

3.3.2.1. Valores predeterminados 18

3.3.2.2. Velocidad de Alimentacion 19

3.3.2.3. Offset del radio de la herramienta . . . 19
3.3.3. Homing . . . 19
3.3.4. Cambios de herramientas . . . 19
3.3.5. Sistemas de coordenadas . . . 19
3.3.5.1. Coordenadas Maquina G53 . . . 19
3.3.5.2. Coordenadas de usuario - G54 a G59.3 . . . 19
3.3.5.3. Cuando este perdido . . . 20
3.3.6. Configuraciones de la maquina . . . 20
3.4. Iniciando LinuxCNC . . . 22
3.4.1. Ejecutando LinuxCNC . . . 22
3.4.1.1. Selector de configuracion . . . 23
3.5. Descripcion general de una máquina CNC . . . 23
3.5.1. Componentes mecanicos . . . 23
3.5.1.1. Ejes . . . 23
3.5.1.2. Husillo . . . 23
3.5.1.3. Refrigerante . . . 23
3.5.1.4. Controles de velocidad y alimentacion . . . 24
3.5.1.5. Interruptor de borrado de bloque . . . 24
3.5.1.6. Interruptor de parada de programa opcional . . . 24
3.5.2. Componentes de control y datos . . . 24
3.5.2.1. Ejes lineales . . . 24
3.5.2.2. Ejes de rotacion . . . 24
3.5.2.3. Punto controlado . . . 24
3.5.2.4. Movimiento lineal coordinado . . . 25
3.5.2.5. Velocidad de avance . . . 25
3.5.2.6. Refrigerante . . . 25
3.5.2.7. Dwell . . . 25
3.5.2.8. Unidades . . . 25
3.5.2.9. Posicion actual . . . 25
3.5.2.10. Plano seleccionado . . . 26
3.5.2.11. Carrusel de herramientas . . . 26
3.5.2.12. Cambio de herramienta . . . 26
3.5.2.13. Pallet Shuttle . . . 26
3.5.2.14. Modo de control de ruta . . . 26
3.5.3. Interaccion del interprete con los interruptores . . . 26
3.5.3.1. Interruptores de porcentaje de alimentacion y velocidad . . . 26
3.5.3.2. Interruptor de borrado de bloque . . . 26
3.5.3.3. Interruptor de parada opcional de programa . . . 27
3.5.4. Tabla de herramientas . . . 27

3.5.5. Parametros . . . 27
3.6. Ejecutando LinuxCNC . . . 28
3.6.1. Invocando LinuxCNC . . . 28
3.6.2. Selector de Configuración . . . 28
3.6.3. Próximos pasos en la configuración . . . 30
3.6.4. Configuraciones del simulador . . . 30
3.6.5. Recursos de configuración . . . 31
3.7. Asistente de configuracion Stepconf . . . 31
3.7.1. Introduccion . . . 31
3.7.2. Pagina de entrada . . . 32
3.7.3. Informacion Basica . . . 35
3.7.4. Prueba de latencia . . . 36
3.7.5. Ajustes del puerto Paralelo . . . 38
3.7.6. Configuracion del puerto paralelo 2 . . . 40
3.7.7. Opciones . . . 41
3.7.8. Configuracion de los Ejes . . . 42
3.7.8.1. Probar este eje . . . 44
3.7.9. Configuracion del husillo . . . 46
3.7.9.1. Control de velocidad del eje . . . 46
3.7.9.2. Movimiento sincronizado con el husillo . . . 47
3.7.9.3. Determinacion de la calibracion del husillo . . . 47
3.7.10. Configuracion de la maquina completa . . . 48
3.7.11. Recorrido de eje y home . . . 48
3.7.11.1. Operando sin interruptores de limite . . . 48
3.7.11.2. Operando sin Switches Home . . . 48
3.7.11.3. Opciones de cableado del interruptor de home y de limite . . . 48
3.8. Asistente de configuracion Mesa . . . 50
3.8.1. Instrucciones punto por punto . . . 51
3.8.2. Crear o editar . . . 51
3.8.3. Informacion basica de la maquina . . . 52
3.8.4. Configuracion externa . . . 54
3.8.5. Configuracion de GUI . . . 56
3.8.6. Configuracion de tarjetas Mesa . . . 59
3.8.7. Configuracion de E/S Mesa . . . 60
3.8.8. Configuracion de Puerto Paralelo . . . 64
3.8.9. Configuración de eje . . . 65
3.8.10. Configuración del eje . . . 70
3.8.11. Opciones avanzadas . . . 71
3.8.12. componentes HAL . . . 71

3.8.13. Uso avanzado de PNCconf . . . 72
3.9. Linux FAQ . . . 73
3.9.1. Login automatico . . . 73
3.9.2. Inicio automatico de LinuxCNC . . . 74
3.9.3. Terminal . . . 74
3.9.4. Paginas de manual . . . 74
3.9.5. Lista de modulos . . . 74
3.9.6. Edicion de archivos de root . . . 74
3.9.6.1. Con la linea de comandos . . . 75
3.9.6.2. Usando la interface grafica . . . 75
3.9.6.3. Acceso tipo super usuario . . . 75
3.9.7. Comandos en la terminal . . . 75
3.9.7.1. Directorio de trabajo . . . 75
3.9.7.2. Cambiar de directorios . . . 75
3.9.7.3. Listado de los archivos en un directorio . . . 75
3.9.7.4. Encontrar un archivo . . . 76
3.9.7.5. Busqueda de texto . . . 76
3.9.7.6. Mensaje de arranque . . . 76
3.9.8. Items convenientes . . . 77
3.9.8.1. Iniciador de terminal . . . 77
3.9.9. Problemas de Hardware . . . 77
3.9.9.1. Informacion Hardware . . . 77
3.9.9.2. Resolucion del monitor . . . 77
3.9.10. Paths . . . 77
3.10. Informacion para Usuarios de Torno . . . 77
3.10.1. Modo Torno . . . 77
3.10.2. Tabla de herramientas de torno . . . 78
3.10.3. Orientacion de la herramienta de torno . . . 78
3.10.4. Tool Touch Off . . . 79
3.10.4.1. X Touch Off . . . 80
3.10.4.2. Z Touch Off . . . 80
3.10.4.3. El offset Z de la maquina . . . 81
3.10.5. Movimiento sincronizado del husillo . . . 81
3.10.6. Arcos . . . 81
3.10.6.1. Diseno de arcos en torno . . . 81
3.10.6.2. Modo de radio y diametro . . . 82
3.10.7. Ruta de la herramienta . . . 82
3.10.7.1. Punto de control . . . 82
3.10.7.2. Corte de angulos sin Compensacion . . . 82

3.10.7.3. Cortando un radio . . . 84
3.10.7.4. Usando Compensacion del Cortador . . . 86
3.11. Basicos del Corte por Plasma para Usuarios de LinuxCNC . . . 86
3.11.1. ¿Qué es el plasma? . . . 86
3.11.2. Inicialización de arco . . . 87
3.11.2.1. Inicio por alta frecuencia . . . 87
3.11.2.2. Inicio por Blowback . . . 87
3.11.3. Plasma y CNC . . . 88
3.11.4. Elegir una máquina de plasma para operaciones CNC . . . 89
3.11.5. Tipos de control de altura de antorcha . . . 90
3.11.6. Señal Arc OK . . . 90
3.11.7. Detección de altura inicial . . . 90
3.11.7.1. Interruptores flotantes . . . 91
3.11.7.2. Detección óhmica . . . 91
3.11.8. Hipersensibilidad con MESA THCAD-5 . . . 91
3.11.9. Ejemplo de código HAL para hipersensibilidad . . . 92
3.11.10. Retraso THC . . . 93
3.11.11. Muestreo de voltaje de antorcha . . . 93
3.11.12. Ruptura de antorcha . . . 94
3.11.13. Esquinas; anti-inmersión por baja velocidad . . . 94
3.11.14. Cruce de vacío / Kerf . . . 94
3.11.15. Agujero y corte de forma pequeña . . . 94
3.11.16. Pines de E/S para controladores de plasma . . . 95
3.11.16.1. Arco OK (entrada) . . . 95
3.11.16.2. Antorcha ON (salida) . . . 96
3.11.16.3. Interruptor flotante (entrada) . . . 96
3.11.16.4. Activación del sensor óhmico (salida) . . . 96
3.11.16.5. Detección óhmica (entrada) . . . 96
3.11.16.6. Breakaway de la antorcha . . . 97
3.11.17. Código G para controladores de plasma . . . 97
3.11.17.1. Seleccione la configuración del material en PlasmaC y use la velocidad de avance para ese material: . . . 97
3.11.17.2. Habilitar/deshabilitar la operación del THC: . . . 97
3.11.17.3. Reducir las velocidades de corte: (por ejemplo, para cortar agujeros) . . . 97
3.11.17.4. Compensación del cortador: . . . 97
3.11.18. Offsets externos y corte por plasma . . . 97
3.11.19. Lectura de voltaje de arco con el THCAD de Mesa . . . 98
3.11.19.1. Conexiones THCAD . . . 99
3.11.19.2. Prueba inicial de THCAD . . . 99

3.11.19.3. Qué modelo THCAD usar . . . 99
3.11.20. Post procesadores y anidamiento . . . 100
3.11.21. Diseñando para entornos eléctricos ruidosos . . . 100
3.11.22. Mesas de agua . . . 101
3.11.23. Mesas de tiro descendente . . . 101
3.11.24. Diseñando para velocidad y aceleración . . . 101
3.11.25. Distancia recorrida por revolución del motor . . . 101
3.11.26. PlasmaC, Configuración de plasma LinuxCNC . . . 101
3.11.27. Control Hypertherm RS485 . . . 102
3.11.28. Postprocesadores para corte por plasma . . . 102

4. Interfaces de usuario **103**
4.1. GUI AXIS . . . 103
4.1.1. Introducción . . . 103
4.1.2. Primeros pasos . . . 104
4.1.2.1. Una sesión típica . . . 104
4.1.3. Pantalla AXIS . . . 105
4.1.3.1. Elementos del menú . . . 105
4.1.3.2. Botones de la barra de herramientas . . . 109
4.1.3.3. Área de visualización gráfica . . . 110
4.1.3.4. Área de visualización de texto . . . 111
4.1.3.5. Control manual . . . 112
4.1.3.6. MDI . . . 113
4.1.3.7. Porcentaje de alimentacion . . . 114
4.1.3.8. Porcentaje de velocidad del husillo . . . 114
4.1.3.9. Velocidad de Jog . . . 114
4.1.3.10. Velocidad máxima . . . 115
4.1.4. Controles del teclado . . . 115
4.1.5. Mostrar estado de LinuxCNC (linuxcnctop) . . . 116
4.1.6. Interfaz MDI . . . 116
4.1.7. axis-remote . . . 117
4.1.8. Cambio de herramienta manual . . . 117
4.1.9. Módulos de Python . . . 117
4.1.10. Usando AXIS en el modo Torno . . . 118
4.1.11. Usando AXIS en el modo de corte de espuma . . . 118
4.1.12. Configuración avanzada . . . 119
4.1.12.1. Filtros de programa . . . 120
4.1.12.2. La base de datos de recursos X . . . 121
4.1.12.3. ~/.axisrc . . . 121

4.1.12.4. USER_COMMAND_FILE . . . 121
4.1.12.5. user_live_update() . . . 121
4.1.12.6. Editor externo . . . 122
4.1.12.7. Panel de control virtual . . . 122
4.1.12.8. Control de vista previa . . . 122
4.1.12.9. Pines Axisui . . . 122
4.1.13. Sugerencias de personalización de Axis . . . 123
4.1.13.1. La función de actualización . . . 123
4.1.13.2. Deshabilitar el cuadro de diálogo Cerrar . . . 123
4.1.13.3. Cambiar la fuente del texto . . . 123
4.1.13.4. Modificar velocidad rápida con atajos de teclado . . . 124
4.1.13.5. Leer el archivo INI . . . 124
4.1.13.6. Leer el estado de linuxcnc . . . 124
4.1.13.7. Cambiar la vista actual . . . 124
4.1.13.8. Crear nuevos pines AXISUI HAL . . . 124
4.1.13.9. Crear nuevos componentes y pines HAL . . . 125
4.1.13.10. Cambiar pestañas con pines HAL . . . 125
4.1.13.11. Agregar un botón de inicio GOTO . . . 126
4.1.13.12. Agregar botón al marco manual . . . 126
4.1.13.13. Lectura de variables internas . . . 127
4.1.13.14. Ocultar widgets . . . 128
4.1.13.15. Cambiar una etiqueta . . . 128
4.1.13.16. Redirigir un comando existente . . . 128
4.1.13.17. Cambiar el color DRO . . . 128
4.1.13.18. Cambiar los botones de la barra de herramientas . . . 129
4.2. GMOCCAPY . . . 129
4.2.1. Introducción . . . 129
4.2.2. Requisitos . . . 130
4.2.3. Cómo obtener gmoccapy . . . 131
4.2.4. Configuración básica . . . 131
4.2.4.1. La sección DISPLAY . . . 132
4.2.4.2. La sección RS274NGC . . . 139
4.2.4.3. La sección MACRO . . . 139
4.2.4.4. La sección TRAJ . . . 142
4.2.5. Pines HAL . . . 142
4.2.5.1. Listas de botones derecha e inferior . . . 143
4.2.5.2. Velocidades y porcentajes . . . 147
4.2.5.3. Pines Jog Hal . . . 149
4.2.5.4. Velocidades Jog y pin hal jog-tortuga . . . 150

4.2.5.5. Pines hal de incremento de jog . . . 150
4.2.5.6. Pin de desbloqueo hardware . . . 151
4.2.5.7. Pines de error . . . 151
4.2.5.8. Pines HAL de Mensajes creados por el usuario . . . 151
4.2.5.9. Pines de realimentación del husillo . . . 152
4.2.5.10. Pines para indicar información sobre el progreso del programa . . . 152
4.2.5.11. Pines relacionado con la herramienta . . . 152
4.2.6. Medición automática de herramientas . . . 154
4.2.6.1. Pines de medición de herramientas . . . 155
4.2.6.2. Modificaciones de archivos INI en medición de herramienta . . . 156
4.2.6.3. Archivos Necesarios . . . 156
4.2.6.4. Conexiones Hal necesarias . . . 157
4.2.7. La página de configuración . . . 157
4.2.7.1. Apariencia . . . 158
4.2.7.2. Hardware . . . 163
4.2.7.3. Configuración avanzada . . . 165
4.2.8. Sección específica del torno . . . 167
4.2.9. Sección específica de plasma . . . 170
4.2.10. Video en You Tube . . . 170
4.2.10.1. Uso básico . . . 171
4.2.10.2. Volantes Jog simulados . . . 171
4.2.10.3. Página de configuración . . . 171
4.2.10.4. Botón de hardware simulado . . . 171
4.2.10.5. Pestañas de usuario . . . 171
4.2.10.6. Videos de Medicion de Herramientas . . . 171
4.2.11. Problemas conocidos . . . 171
4.2.11.1. Números extraños en el área de información . . . 171
4.2.11.2. Macro no finalizada . . . 172
4.3. NGCGUI . . . 173
4.3.1. Descripción general . . . 173
4.3.2. Configuraciones de demostración . . . 174
4.3.3. Ubicaciones de la biblioteca . . . 176
4.3.4. Uso independiente . . . 177
4.3.4.1. NGCGUI independiente . . . 177
4.3.4.2. PYNGCGUI independiente . . . 177
4.3.5. Incrustar NGCGUI . . . 178
4.3.5.1. Incrustar NGCGUI en Axis . . . 178
4.3.5.2. Incrustar PYNGCGUI como una pestaña gladevcp en una GUI . . . 178
4.3.5.3. Items adicionales del archivo INI necesarios para ngcgui o pyngcgui . . . 179

4.3.5.4. Truetype Tracer . . . 180
4.3.5.5. Especificaciones de ruta en archivo INI . . . 180
4.3.5.6. Resumen de los detalles de elementos del archivo INI para usar NGCGUI . . . 181
4.3.6. Requisitos de archivo para compatibilidad NGCGUI . . . 184
4.3.6.1. Requisitos de subrutina de Gcode de un solo archivo (.ngc) . . . 184
4.3.6.2. Requisitos del archivo Gcode-meta-compiler (.gcmc) . . . 186
4.3.7. DB25 Ejemplo . . . 187
4.4. GUI táctil . . . 188
4.4.1. Configuración del panel . . . 189
4.4.1.1. Conexiones HAL . . . 189
4.4.1.2. Recomendado para cualquier configuración . . . 190
4.4.2. Configuración . . . 190
4.4.2.1. Habilitar Touchy . . . 190
4.4.2.2. Preferencias . . . 190
4.4.2.3. Macros . . . 190
4.5. Gscreen . . . 191
4.5.1. Introducción . . . 191
4.5.1.1. Archivo Glade . . . 196
4.5.1.2. PyGTK . . . 196
4.5.2. GladeVCP . . . 196
4.5.2.1. Descripción general . . . 197
4.5.2.2. Construir un Panel GladeVCP . . . 198
4.5.3. Construyendo una simple pantalla limpia personalizada . . . 199
4.5.4. Ejemplo de archivo handler . . . 201
4.5.4.1. Agregar funciones de combinación de teclas . . . 202
4.5.4.2. Linuxcnc State Status . . . 203
4.5.4.3. Teclas de Jogging . . . 203
4.5.5. Gscreen Start Up . . . 204
4.5.6. Configuración INI . . . 205
4.5.7. Mensajes de diálogo del usuario . . . 205
4.5.7.1. Copie el archivo de serie Handler/Glade para su modificación . . . 206
4.6. TkLinuxCNC GUI . . . 207
4.6.1. Introduction . . . 207
4.6.2. Getting Started . . . 207
4.6.2.1. A typical session with TkLinuxCNC . . . 207
4.6.3. Elements of the TkLinuxCNC window . . . 208
4.6.3.1. Main buttons . . . 208
4.6.3.2. Offset display status bar . . . 209
4.6.3.3. Coordinate Display Area . . . 209

4.6.3.4. Automatic control . . . 209
4.6.3.5. Manual Control . . . 209
4.6.3.6. Code Entry . . . 210
4.6.3.7. Jog Speed . . . 211
4.6.3.8. Feed Override . . . 211
4.6.3.9. Spindle speed Override . . . 211
4.6.4. Keyboard Controls . . . 211
4.7. Guía del usuario de PlasmaC . . . 211
4.7.1. Licencia . . . 211
4.7.2. Introducción . . . 212
4.7.3. Requisitos de E/S . . . 212
4.7.4. Instalar LinuxCNC . . . 213
4.7.4.1. Si no tiene Linux instalado . . . 213
4.7.4.2. Si tiene LinuxCNC v2.7 . . . 214
4.7.4.3. Prueba de instalación de LinuxCNC . . . 215
4.7.4.4. Hacer una configuración de trabajo de máquina base . . . 215
4.7.5. Agregar una configuración de PlasmaC a la máquina base . . . 216
4.7.5.1. Configurar . . . 216
4.7.5.2. Ajuste inicial . . . 218
4.7.6. Filtro LOWPASS . . . 219
4.7.7. Antirebote de contactos . . . 219
4.7.8. Lanzador de escritorio . . . 220
4.7.9. Archivos Plasmac . . . 220
4.7.10. Paneles GUI PlasmaC . . . 221
4.7.10.1. Panel de configuración . . . 221
4.7.10.2. Ejecutar panel . . . 224
4.7.10.3. Panel Monitor . . . 226
4.7.10.4. Panel de botones . . . 227
4.7.10.5. Panel de control . . . 228
4.7.10.6. Panel de estadísticas . . . 228
4.7.10.7. Panel de extras . . . 229
4.7.11. Usando PlasmaC . . . 230
4.7.11.1. Coordenadas . . . 230
4.7.11.2. Tolerancia de ruta . . . 231
4.7.11.3. Pausa al final del corte . . . 231
4.7.11.4. Herramientas múltiples . . . 231
4.7.11.5. Manejo de materiales . . . 232
4.7.11.6. Materiales que se agregan y editan desde el código G . . . 232
4.7.11.7. THC . . . 233

4.7.11.8. Compensación del cortador . . . 235
4.7.11.9. Salto de detección de altura inicial . . . 235
4.7.11.10. Sondeo . . . 235
4.7.11.11. Tipos de corte . . . 235
4.7.11.12. Corte de agujeros - Introducción . . . 236
4.7.11.13. Corte de agujeros - Comandos de código G . . . 236
4.7.11.14. Corte de agujeros - Automático . . . 237
4.7.11.15. Corte individual . . . 238
4.7.11.16. Modo de malla (metal expandido) . . . 240
4.7.12. Biblioteca de formas conversacionales . . . 240
4.7.12.1. Configuración . . . 241
4.7.12.2. Forma única . . . 242
4.7.12.3. Líneas y arcos . . . 242
4.7.12.4. Grupo de formas . . . 242
4.7.12.5. Rotar . . . 243
4.7.12.6. Matriz . . . 243
4.7.12.7. Guardar un trabajo . . . 243
4.7.12.8. Ejecutar un trabajo . . . 243
4.7.13. Grabador . . . 243
4.7.14. Punteo . . . 244
4.7.15. Mensajes de error . . . 245
4.7.15.1. Errores críticos . . . 245
4.7.15.2. Errores de advertencia . . . 246
4.7.16. Archivo INI . . . 247
4.7.16.1. Común . . . 247
4.7.16.2. Especificos de GUI Axis . . . 248
4.7.16.3. Específicos de GUI Gmoccapy . . . 249
4.7.17. Botones de usuario personalizados . . . 250
4.7.17.1. Nombres de botones . . . 250
4.7.17.2. Código de botón . . . 250
4.7.17.3. Imagen de botón . . . 251
4.7.18. Archivo de material . . . 252
4.7.19. Configuraciones de ejemplo . . . 253
4.7.20. Muestras de NGC . . . 253
4.7.21. Panel de prueba . . . 253
4.7.22. Actualizar una configuración existente de LinuxCNC . . . 253
4.7.22.1. Actualizar LinuxCNC . . . 253
4.7.23. Actualizar PlasmaC . . . 253
4.7.23.1. Actualización automática . . . 253

4.7.23.2. Actualización manual . . . 253
4.7.24. Reconfigurar una configuración de PlasmaC existente . . . 254
4.7.25. Cambiar tipo de instalación de LinuxCNC . . . 255
4.7.26. Materialverter . . . 256
4.7.27. Comunicaciones Hypertherm Powermax . . . 257
4.7.28. Mesa THCAD . . . 258
4.7.28.1. Configuración de puerto paralelo . . . 258
4.7.28.2. Configurar Mesa 7i96: . . . 259
4.7.28.3. Obtención de la relación de división . . . 259
4.7.28.4. Valores de calibración . . . 259
4.7.29. Soporte . . . 260
4.7.30. Apéndice . . . 260
4.7.30.1. Códigos específicos de Plasmac . . . 260
4.7.30.2. Ejemplos de Plasmac Gcode . . . 261
4.7.30.3. Conexiones RS485 . . . 263
4.7.30.4. Arc OK con un relé reed . . . 265

5. Programacion **267**
5.1. Sistemas de coordenadas . . . 267
5.1.1. Introducción . . . 267
5.1.2. Sistema de Coordenadas de la Máquina . . . 267
5.1.3. Sistemas de coordenadas . . . 268
5.1.3.1. Sistema de Coordenadas Predeterminado . . . 269
5.1.3.2. Configuración de Offsets del Sistema de Coordenadas . . . 269
5.1.4. Offsets Locales y Globales . . . 269
5.1.4.1. El comando G52 . . . 269
5.1.4.2. Los Comandos G92 . . . 270
5.1.4.3. Configuración de valores G92 . . . 271
5.1.4.4. Precauciones de Persistencia G92 . . . 271
5.1.4.5. Precauciones de Interacción G92 y G52 . . . 272
5.1.5. Programas de Muestra usando Offsets . . . 272
5.1.5.1. Programa de Muestra utilizando Offsets de Coordenadas de Pieza . . . 272
5.1.5.2. Programa de muestra usando offsets G52 . . . 273
5.2. Códigos G . . . 273
5.2.1. Convenciones . . . 273
5.2.2. Tabla de referencia rápida de código G . . . 273
5.2.3. G0 Movimiento rápido . . . 274
5.2.3.1. Velocidad rápida . . . 275
5.2.4. G1 Movimiento lineal . . . 275

5.2.5. G2, G3 Movimiento de arco . . . 276
5.2.5.1. Arcos de formato centro . . . 276
5.2.5.2. Ejemplos de formato centro . . . 278
5.2.5.3. Arcos de formato radio . . . 279
5.2.6. G4 Dwell . . . 280
5.2.7. G5 Spline cúbico . . . 280
5.2.8. G5.1 Spline cuadrático . . . 281
5.2.9. G5.2 G5.3 Bloque NURBS . . . 281
5.2.10. Modo de diámetro de torno G7 . . . 283
5.2.11. Modo de radio de torno G8 . . . 283
5.2.12. G10 L1 Establecer tabla de herramientas . . . 284
5.2.13. G10 L2 Establecer sistema de coordenadas . . . 284
5.2.14. G10 L10 Establecer tabla de herramientas . . . 286
5.2.15. G10 L11 Establecer tabla de herramientas . . . 286
5.2.16. G10 L20 Establecer sistema de coordenadas . . . 287
5.2.17. G17 - G19.1 Seleccionar plano . . . 287
5.2.18. Unidades G20, G21 . . . 287
5.2.19. G28, G28.1 Ir/Establecer posición predefinida . . . 288
5.2.20. G30, G30.1 Ir/Establecer posición predefinida . . . 288
5.2.21. Movimiento sincronizado del husillo G33 . . . 289
5.2.22. G33.1 Roscado rígido . . . 290
5.2.23. G38.n Sonda recta . . . 291
5.2.24. Compensación G40 desactivada . . . 292
5.2.25. G41, G42 Compensación del cortador . . . 292
5.2.26. G41.1, G42.1 Compensación dinámica del cortador . . . 293
5.2.27. Compensación de longitud de herramienta G43 . . . 293
5.2.28. G43.1 Offset de longitud dinámica de herramienta . . . 294
5.2.29. G43.2 Aplicar offset de longitud de herramienta adicional . . . 294
5.2.30. G49 Cancelar compensación de longitud de herramienta . . . 295
5.2.31. G52 Compensación del sistema de coordenadas local . . . 295
5.2.32. G53 Mover en coordenadas de máquina . . . 295
5.2.33. G54-G59.3 Seleccionar sistema de coordenadas . . . 295
5.2.34. Modo de ruta exacta G61 . . . 296
5.2.35. G61.1 Modo de parada exacta . . . 296
5.2.36. Mezcla de ruta G64 . . . 296
5.2.37. Ciclo de acabado del torno G70 . . . 297
5.2.38. G71 G72 Ciclo de desbaste en Torno . . . 298
5.2.39. G73 Ciclo de taladrado con rotura de viruta . . . 299
5.2.40. G74 Ciclo de roscado izquierdo, con Dwell . . . 299

5.2.41. Ciclo de roscado G76 . . . 300
5.2.42. Ciclos Fijos . . . 302
5.2.42.1. Palabras comunes . . . 302
5.2.42.2. Palabras Sticky . . . 302
5.2.42.3. Repetir ciclo . . . 303
5.2.42.4. Modo de retracción . . . 303
5.2.42.5. Errores de ciclo fijo . . . 303
5.2.42.6. Movimientos preliminares e intermedios . . . 303
5.2.42.7. ¿Por qué usar un ciclo fijo? . . . 304
5.2.43. G80 Cancelar ciclo fijo . . . 305
5.2.44. G84 Ciclo de roscado a derecha, con Dwell . . . 311
5.2.45. G85 Ciclo mandrinado, salida a avance . . . 312
5.2.46. G86 Ciclo de taladrado, parada del husillo, salida a rápido. . . . 312
5.2.47. G87 Ciclo de mandrinado posterior . . . 312
5.2.48. Ciclo de taladrado G88, parada del husillo, salida manual . . . 312
5.2.49. G89 Ciclo de mandrinado, con Dwell, salida a avance . . . 313
5.2.50. G90, G91 Modo de distancia . . . 313
5.2.51. G90.1, G91.1 Modo de distancia de arco . . . 313
5.2.52. G92 Offset de sistema de coordenadas G92 . . . 313
5.2.53. G92.1, G92.2 Restablecer compensaciones G92 . . . 314
5.2.54. G92.3 Restaurar compensaciones G92 . . . 314
5.2.55. Modo de velocidad de alimentación G93, G94, G95 . . . 315
5.2.56. G96, G97 Modo de control del husillo . . . 315
5.2.57. G98, G99 Nivel de retorno del ciclo fijo . . . 316
5.3. Códigos M . . . 316
5.3.1. Tabla de referencia rápida de códigos M . . . 316
5.3.2. M0, M1 Pausa del programa . . . 316
5.3.3. M2, M30 Fin del Programa . . . 317
5.3.4. Pausa de cambio de palet M60 . . . 317
5.3.5. M3, M4, M5 Control de husillo . . . 318
5.3.6. Cambio de herramienta M6 . . . 318
5.3.6.1. Cambio manual de herramienta . . . 318
5.3.6.2. Cambiador de herramientas . . . 318
5.3.7. M7, M8, M9 Control de refrigerante . . . 319
5.3.8. M19 Orientacion del Husillo . . . 319
5.3.9. M48, M49 Control de ajustes de velocidad y alimentación . . . 320
5.3.10. M50 Control de ajuste de alimentación . . . 320
5.3.11. M51 Control de ajuste de velocidad del husillo . . . 320
5.3.12. M52 Control de alimentación adaptable . . . 320

5.3.13. M53 Control de parada de alimentación . . . 321
5.3.14. M61 Establecer herramienta actual . . . 321
5.3.15. M62 - Control de salida digital M65 . . . 321
5.3.16. M66 Esperar en entrada . . . 322
5.3.17. M67 Salida analógica, sincronizada . . . 322
5.3.18. M68 Salida analógica, inmediata . . . 323
5.3.19. M70 Guardar estado modal . . . 323
5.3.20. M71 Invalidar estado modal almacenado . . . 324
5.3.21. M72 Restaurar estado modal . . . 324
5.3.22. M73 Guardar y Autorestaurar estado modal . . . 325
5.3.23. M98 y M99 . . . 326
5.3.23.1. Restauración selectiva del estado modal . . . 326
5.3.24. M100 - M199 Comandos definidos por el usuario . . . 326
5.4. Códigos O . . . 328
5.4.1. Numeración . . . 328
5.4.2. Comentarios . . . 328
5.4.3. Subrutinas . . . 328
5.4.3.1. Programas numerados al estilo Fanuc . . . 329
5.4.4. Bucles . . . 331
5.4.5. Condicionales . . . 331
5.4.6. Repeat . . . 332
5.4.7. Indirección . . . 332
5.4.8. Llamando a archivos . . . 333
5.4.9. Valores de retorno de subrutina . . . 333
5.4.10. Errores . . . 334
5.5. Otros códigos . . . 334
5.5.1. F: Establecer velocidad de alimentación . . . 334
5.5.2. S: Establecer la velocidad de husillo . . . 334
5.5.3. T: Seleccionar herramienta . . . 334
5.6. Ejemplos de código G . . . 335
5.6.1. Ejemplos de fresado . . . 335
5.6.1.1. Fresado helicoidal . . . 335
5.6.1.2. Ranurado . . . 335
5.6.1.3. Sondeo en cuadrícula . . . 335
5.6.1.4. Sonda inteligente . . . 335
5.6.1.5. Sondeo de longitud de herramienta . . . 335
5.6.1.6. Sonda de agujero . . . 336
5.6.1.7. Compensación del cortador . . . 336
5.6.2. Ejemplos de torno . . . 336
5.6.2.1. Roscado . . . 336
5.7. Diferencias RS274/NGC . . . 336
5.7.1. Cambios desde RS274/NGC . . . 336
5.7.2. Adiciones a RS274/NGC . . . 337

6. Compensacion de Herramientas **338**
6.1. Compensación de herramientas . . . 338
6.1.1. Compensación de longitud de herramienta . . . 338
6.1.1.1. Touch Off . . . 338
6.1.1.2. Usando G10 L1/L10/L11 . . . 339
6.1.2. Tabla de herramientas . . . 339
6.1.2.1. Formato de tabla de herramientas . . . 339
6.1.2.2. Cambiadores de herramientas . . . 341
6.1.3. Compensación del cortador . . . 341
6.1.3.1. Descripción general . . . 342
6.1.3.2. Ejemplos . . . 344
6.2. GUI del Editor de Herramientas . . . 345
6.2.1. Descripción general . . . 345
6.2.2. Ordenación por columnas . . . 346
6.2.3. Selección de columnas . . . 346
6.2.4. Uso independiente . . . 347

IV Configuracion **348**

7. Informacion General **349**
7.1. Conceptos para Integradores . . . 349
7.1.1. Ubicaciones de archivos . . . 349
7.1.1.1. LinuxCNC Instalado . . . 349
7.1.1.2. LinuxCNC desde la Línea de Comandos . . . 350
7.1.2. Archivos . . . 350
7.1.3. Sistemas steppers . . . 350
7.1.3.1. Período Base . . . 350
7.1.3.2. Temporizacion de pasos . . . 351
7.1.4. Servosistemas . . . 351
7.1.4.1. Operación básica . . . 351
7.1.4.2. Término proporcional . . . 353
7.1.4.3. Término integral . . . 353
7.1.4.4. Término derivado . . . 353
7.1.4.5. Ajuste de bucle . . . 354
7.1.4.6. Sintonización manual . . . 354
7.1.5. RTAI . . . 354
7.1.5.1. ACPI . . . 354
7.2. Test de Latencia . . . 354
7.3. Afinación de Steppers . . . 357

7.3.1. Obtener el máximo rendimiento del Software de Stepping . . . 357
7.3.1.1. Ejecutar una prueba de latencia . . . 357
7.3.1.2. Descubra lo que esperan sus drivers . . . 358
7.3.1.3. Eligir su BASE_PERIOD . . . 358
7.3.1.4. Uso de steplen, stepspace, dirsetup y/o dirhold . . . 359
7.3.1.5. ¡No hacer suposiciones! . . . 359
7.4. Diagnósticos para steppers . . . 360
7.4.1. Problemas comunes . . . 360
7.4.1.1. El Stepper se mueve solo un paso . . . 360
7.4.1.2. Los steppers no se mueven en absoluto . . . 360
7.4.1.3. Distancia no correcta . . . 360
7.4.2. Mensajes de error . . . 360
7.4.2.1. Error de seguimiento . . . 360
7.4.2.2. Error RTAPI . . . 361
7.4.3. Pruebas . . . 361
7.4.3.1. Temporización de pasos . . . 361

8. Configuracion **362**
8.1. Configuración INI . . . 362
8.1.1. Los Componentes del Archivo INI . . . 362
8.1.1.1. Comentarios . . . 362
8.1.1.2. Secciones . . . 363
8.1.1.3. Variables . . . 363
8.1.1.4. Secciones y variables personalizadas . . . 364
8.1.1.5. Archivos include . . . 364
8.1.2. Secciones del Archivo INI . . . 365
8.1.2.1. Sección [EMC] . . . 365
8.1.2.2. Sección [DISPLAY] . . . 365
8.1.2.3. Sección [FILTER] . . . 368
8.1.2.4. Sección [RS274NGC] . . . 369
8.1.2.5. Sección [EMCMOT] . . . 371
8.1.2.6. Sección [TASK] . . . 371
8.1.2.7. Sección [HAL] . . . 371
8.1.2.8. Sección [HALUI] . . . 372
8.1.2.9. Sección [APPLICATIONS] . . . 373
8.1.2.10. [TRAJ] Sección . . . 373
8.1.2.11. Sección [KINS] . . . 375
8.1.2.12. Sección [AXIS_<letter>] . . . 376
8.1.2.13. Sección [JOINT_<num>] . . . 376

8.1.2.14. Sección [EMCIO] . . . 382
8.1.3. Descripción general . . . 382
8.1.4. Prerrequisitos . . . 383
8.1.5. Ejemplo de Diseño de interruptor home separado . . . 383
8.1.6. Ejemplo de Diseño de Límite/Home compartido . . . 384
8.1.7. Secuencia Homing . . . 385
8.1.8. Configuración . . . 387
8.1.8.1. HOME_SEARCH_VEL . . . 387
8.1.8.2. HOME_LATCH_VEL . . . 387
8.1.8.3. HOME_FINAL_VEL . . . 387
8.1.8.4. HOME_IGNORE_LIMITS . . . 388
8.1.8.5. HOME_USE_INDEX . . . 388
8.1.8.6. HOME_INDEX_NO_ENCODER_RESET . . . 388
8.1.8.7. HOME_OFFSET . . . 388
8.1.8.8. HOME . . . 388
8.1.8.9. HOME_IS_SHARED . . . 389
8.1.8.10. HOME_ABSOLUTE_ENCODER . . . 389
8.1.8.11. HOME_SEQUENCE . . . 389
8.1.8.12. VOLATILE_HOME . . . 390
8.1.8.13. LOCKING_INDEXER . . . 390
8.1.8.14. Homing inmediato . . . 390
8.1.8.15. Inhibición de home . . . 391
8.1.9. Plano predeterminado . . . 392
8.1.10. Configuración INI . . . 392
8.2. Archivos HALTCL . . . 393
8.2.1. Compatibilidad . . . 393
8.2.2. Comandos Haltcl . . . 393
8.2.3. Variables Ini Haltcl . . . 394
8.2.4. Convertir archivos .hal a archivos .tcl . . . 395
8.2.5. Notas para Haltcl . . . 395
8.2.6. Ejemplos de Haltcl . . . 395
8.2.7. Haltcl Interactivo . . . 396
8.2.8. Ejemplos Haltcl de la distribución de (sim) . . . 396
8.3. Remap Extendiendo el código G . . . 396
8.3.1. Introducción: Extensión del intérprete RS274NGC mediante remapeado de códigos . . . 396
8.3.1.1. Definición: Remapeado de Códigos . . . 396
8.3.1.2. ¿Por qué querría extender el intérprete RS274NGC? . . . 396
8.3.2. Comenzando . . . 398
8.3.2.1. Escogiendo un código . . . 398

8.3.2.2. Manejo de parámetros . . . 398
8.3.2.3. Manejo de resultados . . . 398
8.3.2.4. Secuenciación de ejecución . . . 399
8.3.2.5. Un ejemplo mínimo de código remapeado . . . 399
8.3.3. Configuración del remapeo . . . 399
8.3.3.1. La sentencia REMAP . . . 399
8.3.3.2. Combinaciones útiles de opciones de REMAP . . . 400
8.3.3.3. El parámetro argspec . . . 401
8.3.4. Actualización de una configuración existente para remapeado . . . 404
8.3.5. Remapeo de códigos relacionados con el cambio de herramienta: T, M6, M61 . . . 404
8.3.5.1. Descripción general . . . 404
8.3.5.2. Entender el rol de iocontrol con códigos de cambio de herramienta remapeados . . . 405
8.3.5.3. Especificando el reemplazo M6 . . . 406
8.3.5.4. Configurando iocontrol con un M6 remapeado . . . 407
8.3.5.5. Escribiendo el cambio y preparando procedimientos O-word . . . 408
8.3.5.6. Haciendo cambios mínimos a los códigos incorporados, incluyendo `M6` . . . 408
8.3.5.7. Especificando el reemplazo de T (preparar) . . . 409
8.3.5.8. Manejo de errores: tratando con abort . . . 410
8.3.5.9. Manejo de errores: error en un procedimiento NGC de remapeado de código . . . 410
8.3.6. Reasignando otros códigos existentes: S, M0, M1, M60 . . . 411
8.3.6.1. Selección automática de marcha reasignando S (ajuste de la velocidad del husillo) . . . 411
8.3.6.2. Ajustando el comportamiento de M0, M1, M60 . . . 411
8.3.7. Creando nuevos ciclos de codigo G . . . 411
8.3.8. Configurando Python Embebido . . . 412
8.3.8.1. Plugin Python : configuración de archivos ini . . . 412
8.3.8.2. Ejecutando sentencias de Python desde el intérprete . . . 413
8.3.9. Programación de Python Embebido en el intérprete RS274NGC . . . 413
8.3.9.1. El espacio de nombres del plugin Python . . . 413
8.3.9.2. El intérprete visto desde Python . . . 413
8.3.9.3. Las funciones del intérprete `__init__` y `__delete__` . . . 413
8.3.9.4. Convenciones de llamada: NGC a Python . . . 414
8.3.9.5. Convenciones de llamada: Python a NGC . . . 416
8.3.9.6. Módulos Integrados . . . 418
8.3.10. Agregando Parámetros Nombrados Predefinidos . . . 419
8.3.11. Rutinas estándar de union . . . 419
8.3.11.1. T: `prepare_prolog` y `prepare_epilog` . . . 419
8.3.11.2. M6: `change_prolog` y `change_epilog` . . . 420
8.3.11.3. Ciclos de código G: `cycle_prolog` y `cycle_epilog` . . . 421
8.3.11.4. S (Establecer velocidad): `setspeed_prolog` y `setspeed_epilog` . . . 422

8.3.11.5. F (Establecer Alimentacion): setfeed_prolog y setfeed_epilog 422
8.3.11.6. M61 Establecer el número de herramienta: settool_prolog y settool_epilog 422
8.3.12. Ejecución del código remapeado 422
8.3.12.1. Entorno de llamada a procedimiento NGC durante remapeados 422
8.3.12.2. Códigos reasignados anidados 422
8.3.12.3. Número de secuencia durante remapeos 422
8.3.12.4. Banderas de depuracion 422
8.3.12.5. Depuración de código Python embebido 423
8.3.13. Vista previa de Axis y Ejecución de código remapeado 424
8.3.14. Códigos remapeables 424
8.3.14.1. Códigos existentes que pueden ser remapeados 424
8.3.14.2. Códigos G actualmente sin asignar: 424
8.3.14.3. Códigos M actualmente sin asignar: 428
8.3.14.4. tiempo de lectura anticipada y tiempo de ejecución 428
8.3.14.5. plugin / pickle hack 428
8.3.14.6. Módulo, métodos, clases, etc. referencia. 428
8.3.15. Introducción: Extender la ejecución de tareas 429
8.3.15.1. ¿Por qué quieres cambiar la ejecución de tareas? 429
8.3.15.2. Un diagrama: tarea, interp, iocontrol, UI (??) 429
8.3.16. Modelos de ejecución de tareas 429
8.3.16.1. Ejecución tradicional de iocontrol / iocontrolv2 429
8.3.16.2. Redefiniendo procedimientos de IO 429
8.3.16.3. Procedimientos de Python en tiempo de ejecución 429
8.3.17. Una breve inspección de la ejecución del programa LinuxCNC 429
8.3.17.1. Estado del intérprete 429
8.3.17.2. Interacción de Task e Interpreter, puesta en cola y lectura anticipada 430
8.3.17.3. Predecir la posición de la máquina 430
8.3.17.4. El destructor de colas rompe la predicción de la posición 430
8.3.17.5. ¿Cómo se tratan los busters de cola? 431
8.3.17.6. Orden de palabras y orden de ejecución 431
8.3.17.7. Análisis 431
8.3.17.8. Ejecución 431
8.3.17.9. Ejecución de procedimientos 431
8.3.17.10. Cómo funciona el cambio de herramienta actualmente 432
8.3.17.11. Cómo funciona Tx (Prepare Tool) 432
8.3.17.12. Cómo funciona M6 (cambio de herramienta) 433
8.3.17.13. Cómo funciona M61 (Cambiar número de herramienta) 434
8.3.18. Cambios 434
8.3.19. Depuración 434

8.4. Componente moveoff . . . 434
8.4.1. Modificar una configuración existente . . . 436
8.5. Configuración de steppers . . . 438
8.5.1. Introducción . . . 438
8.5.2. Velocidad de paso máxima . . . 439
8.5.3. Pinout . . . 439
8.5.3.1. Pinout estándar HAL . . . 439
8.5.3.2. Descripción general . . . 441
8.5.3.3. Cambiar standard_pinout.hal . . . 441
8.5.3.4. Cambio de polaridad de una señal . . . 441
8.5.3.5. Agregar control de velocidad PWM al husillo . . . 441
8.5.3.6. Agregar una señal de habilitación . . . 442
8.5.3.7. Botón externo ESTOP . . . 442

9. Paneles de Control **443**

10. Interfaces de Usuario **444**
10.1. Ejemplos Halui . . . 444
10.1.1. Arranque remoto . . . 444
10.1.2. Pausar y Reanudar . . . 445
10.2. Interfaz de Python . . . 445
10.2.1. El módulo linuxcnc Python . . . 445
10.2.2. Patron de uso de la interfaz NML LinuxCNC . . . 446
10.2.3. Lectura del Estado de LinuxCNC . . . 446
10.2.3.1. Atributos en linuxcnc.stat . . . 446
10.2.3.2. El diccionario axis . . . 451
10.2.3.3. El diccionario joint . . . 451
10.2.4. El diccionario spindle . . . 453
10.2.5. Preparacion para enviar comandos . . . 453
10.2.6. Enviar comandos a través de `linuxcnc.command` . . . 454
10.2.6.1. Atributos `linuxcnc.command` . . . 454
10.2.6.2. métodos `linuxcnc.command`: . . . 455
10.2.7. Lectura del canal de error . . . 457
10.2.8. Lectura de valores de archivo ini . . . 458
10.2.9. El tipo `linuxcnc.positionlogger` . . . 458
10.2.9.1. miembros . . . 458
10.2.9.2. métodos . . . 459

11. Drivers **460**
11.1. Controlador de puerto paralelo . . . 460
11.1.1. Carga . . . 460
11.1.2. Dirección del puerto PCI . . . 462
11.1.3. Pines . . . 462
11.1.4. Parámetros . . . 462
11.1.5. Funciones . . . 463
11.1.6. Problemas comunes . . . 463
11.1.7. Usando DoubleStep . . . 463

12. Ejemplos de Drivers **464**
12.1. Puerto paralelo PCI . . . 464
12.2. Control de husillo . . . 465
12.2.1. Velocidad del husillo 0-10v . . . 465
12.2.2. Velocidad del husillo PWM . . . 465
12.2.3. Habilitación del husillo . . . 465
12.2.4. Dirección del husillo . . . 465
12.2.5. Arranque suave del husillo . . . 466
12.2.6. Feedback del husillo . . . 466
12.2.6.1. Movimiento sincronizado del husillo . . . 466
12.2.6.2. Husillo a velocidad . . . 467
12.3. Husillo con GS2 . . . 468

13. PLC **470**

14. HAL **471**
14.1. Introducción a HAL . . . 471
14.1.1. HAL se basa en técnicas de diseño de sistemas tradicionales . . . 471
14.1.1.1. Selección de componentes . . . 472
14.1.1.2. Diseño de interconexiones . . . 473
14.1.1.3. Implementación . . . 473
14.1.1.4. Pruebas . . . 473
14.1.1.5. Resumen . . . 473
14.1.2. Conceptos HAL . . . 474
14.1.3. Componentes HAL . . . 476
14.1.3.1. Programas externos con "enganches" HAL . . . 476
14.1.3.2. Componentes internos . . . 476
14.1.3.3. Controladores de hardware . . . 476
14.1.3.4. Herramientas y utilidades . . . 477
14.1.4. Problemas de sincronización en HAL . . . 477

14.2. Referencia básica HAL . . . 478
14.2.1. Comandos HAL . . . 478
14.2.1.1. loadrt . . . 479
14.2.1.2. addf . . . 479
14.2.1.3. loadusr . . . 480
14.2.1.4. net . . . 480
14.2.1.5. setp . . . 481
14.2.1.6. sets . . . 482
14.2.1.7. unlinkp . . . 482
14.2.1.8. Comandos obsoletos . . . 482
14.2.2. Datos HAL . . . 483
14.2.2.1. Bit . . . 483
14.2.2.2. Float . . . 483
14.2.2.3. s32 . . . 483
14.2.2.4. u32 . . . 483
14.2.3. Archivos HAL . . . 483
14.2.4. Componentes HAL . . . 483
14.2.5. Componentes lógicos . . . 484
14.2.5.1. and2 . . . 484
14.2.5.2. not . . . 484
14.2.5.3. or2 . . . 485
14.2.5.4. xor2 . . . 485
14.2.5.5. Ejemplos de lógica . . . 485
14.2.6. Componentes de conversión . . . 486
14.2.6.1. weighted_sum . . . 486
14.3. HAL TWOPASS . . . 486
14.3.1. TWOPASS . . . 486
14.3.2. Excluyendo archivos .hal . . . 488
14.3.3. Post GUI . . . 489
14.3.4. Ejemplos . . . 489
14.4. Tutorial HAL . . . 489
14.4.1. Introducción . . . 489
14.4.1.1. Notación . . . 489
14.4.1.2. Completado por tabulador . . . 489
14.4.1.3. El entorno RTAPI . . . 490
14.4.2. Un ejemplo simple . . . 490
14.4.2.1. Cargando un componente . . . 490
14.4.2.2. Examinando el HAL . . . 490
14.4.2.3. Hacer correr el código en tiempo real . . . 492

14.4.2.4. Cambiar los parámetros . . . 493
14.4.2.5. Guardar la configuración HAL . . . 494
14.4.2.6. Saliendo de halrun . . . 494
14.4.2.7. Restaurando la configuración HAL . . . 494
14.4.2.8. Eliminando HAL de la memoria . . . 494
14.4.3. Halmeter . . . 495
14.4.4. Ejemplo Stepgen . . . 497
14.4.4.1. Instalación de los componentes . . . 497
14.4.4.2. Conexión de pines con señales . . . 498
14.4.4.3. Configuración de la ejecución en tiempo real: hilos y funciones . . . 499
14.4.4.4. Parámetros de configuración . . . 500
14.4.4.5. ¡Ejecutando! . . . 501
14.4.5. Halscope . . . 501
14.4.5.1. Conectando las sondas . . . 503
14.4.5.2. Capturando nuestras primeras formas de onda . . . 506
14.4.5.3. Ajustes verticales . . . 507
14.4.5.4. Disparando . . . 508
14.4.5.5. Ajustes horizontales . . . 510
14.4.5.6. Más canales . . . 511
14.4.5.7. Más muestras . . . 512
14.5. Componentes principales . . . 514
14.5.1. Motion (motmod) . . . 515
14.5.1.1. Opciones . . . 515
14.5.1.2. Pines . . . 516
14.5.1.3. Parámetros . . . 518
14.5.1.4. Funciones . . . 519
14.5.2. Pines y parámetros de ejes y articulaciones . . . 519
14.5.3. iocontrol . . . 519
14.5.3.1. Pines . . . 519
14.5.4. Configuración ini . . . 520
14.5.4.1. Pines . . . 520
14.6. Interfaces de dispositivos canónicos . . . 521
14.6.1. Introducción . . . 521
14.6.2. Entrada digital . . . 521
14.6.2.1. Pines . . . 521
14.6.2.2. Parámetros . . . 521
14.6.2.3. Funciones . . . 521
14.6.3. Salida digital . . . 521
14.6.3.1. Pines . . . 521

14.6.3.2. Parámetros . . . 521
14.6.3.3. Funciones . . . 521
14.6.4. Entrada Analógica . . . 521
14.6.4.1. Pines . . . 522
14.6.4.2. Parámetros . . . 522
14.6.4.3. Funciones . . . 522
14.6.5. Salida analógica . . . 522
14.6.5.1. Pines . . . 522
14.6.5.2. Parámetros . . . 522
14.6.5.3. Funciones . . . 523
14.6.6. Halshow . . . 524
14.6.7. Halscope . . . 525
14.6.8. Sim Pin . . . 525
14.6.9. Sonda simulada . . . 526
14.6.10.Hal Histogram . . . 527
14.6.11.Halreport . . . 528
14.7. Halshow . . . 530
14.7.1. Comenzando Halshow . . . 530
14.7.2. Area de Arbol HAL . . . 530
14.7.3. Area Show HAL . . . 532
14.7.4. Pestaña WATCH . . . 535
14.8. Componentes HAL . . . 536
14.8.1. Comandos y componentes de espacio de usuario . . . 536
14.8.2. Lista de componentes en tiempo real . . . 537
14.8.2.1. Componentes principales de LinuxCNC . . . 538
14.8.2.2. Componentes lógicos y enfocados a bits (bitwise) . . . 538
14.8.2.3. Componentes aritméticos y de punto flotante . . . 539
14.8.2.4. Conversion de tipos . . . 540
14.8.2.5. Controladores de hardware . . . 541
14.8.2.6. Cinemática . . . 541
14.8.2.7. Control del motor . . . 542
14.8.2.8. BLDC y control de motores trifásicos . . . 542
14.8.2.9. Otros componentes . . . 542
14.8.3. Llamadas API HAL . . . 544
14.8.4. Llamadas RTAPI . . . 544
14.9. Descripciones de Componentes HAL . . . 545
14.9.1. Stepgen . . . 545
14.9.2. PWMgen . . . 554
14.9.3. Encoder . . . 555

14.9.4. PID . . . 558
14.9.5. Encoder simulado . . . 560
14.9.6. Debounce . . . 561
14.9.7. Siggen . . . 562
14.9.8. lut5 . . . 562
14.10 Ejemplos HAL . . . 565
14.10.1 Conexión de dos salidas . . . 565
14.10.2 Cambio manual de herramientas . . . 565
14.10.3 Cálculo de la velocidad . . . 566
14.10.4 Arranque suave . . . 567
14.10.5 Stand Alone HAL . . . 569
14.10.6 HAL independiente . . . 570
14.11 El Generador de Componentes HAL . . . 570
14.11.1 Introducción . . . 570
14.11.2 Instalación . . . 571
14.11.3 Usando un componente . . . 571
14.11.4 Definiciones . . . 571
14.11.5 Creación de instancias . . . 572
14.11.6 Parámetros implícitos . . . 572
14.11.7 Sintaxis . . . 572
14.11.7.1. Funciones HAL . . . 574
14.11.7.2. Opciones . . . 574
14.11.7.3. Licencia y autoría . . . 575
14.11.7.4. Almacenamiento de datos por instancia . . . 575
14.11.7.5. Comentarios . . . 575
14.11.8 Restricciones . . . 575
14.11.9 Macros de Conveniencia . . . 576
14.11.10 Componentes con una sola función . . . 576
14.11.11 Personalidad del componente . . . 576
14.11.12 Compilando . . . 577
14.11.13 Compilación de componentes en tiempo real fuera del árbol fuente . . . 577
14.11.14 Compilación de componentes de espacio de usuario fuera del árbol de fuentes . . . 577
14.11.15 Ejemplos . . . 578
14.11.15.1 constant . . . 578
14.11.15.2 sincos . . . 578
14.11.15.3 out8 . . . 578
14.11.15.4 hal_loop . . . 579
14.11.15.5 arraydemo . . . 579
14.11.15.6 rand . . . 579

14.11.15.7 logic . . . 580
14.11.15.8 funciones generales . . . 581
14.11.16 Uso de línea de comando . . . 581
14.12 Crear Componentes Python de Espacio de Usuario . . . 581
14.12.1 Uso Básico . . . 581
14.12.2 Componentes de espacio de usuario y retrasos . . . 582
14.12.3 Crear pines y parámetros . . . 582
14.12.3.1 Cambiar el Prefijo . . . 583
14.12.4 Lectura y Escritura de Pines y Parámetros . . . 583
14.12.4.1 Manejo de Pines de Salida (HAL_OUT) . . . 583
14.12.4.2 Manejo de pines bidireccionales (HAL_IO) . . . 583
14.12.5 Salida . . . 583
14.12.6 Funciones útiles . . . 584
14.12.6.1 component_exists . . . 584
14.12.6.2 component_is_ready . . . 584
14.12.6.3 get_msg_level . . . 584
14.12.6.4 set_msg_level . . . 584
14.12.6.5 connect . . . 584
14.12.6.6 get_value . . . 584
14.12.6.7 new_signal . . . 584
14.12.6.8 pin_has_writer . . . 584
14.12.6.9 get_name . . . 585
14.12.6.10 get_type . . . 585
14.12.6.11 get_dir . . . 585
14.12.6.12 get . . . 585
14.12.6.13 set . . . 585
14.12.6.14 is_pin . . . 585
14.12.6.15 sampler_base . . . 585
14.12.6.16 stream_base . . . 585
14.12.6.17 stream . . . 585
14.12.6.18 set_p . . . 586
14.12.7 Constantes . . . 586
14.12.8 Información del sistema . . . 586
14.12.9 Usar con hal_glib en el handler GladeVCP . . . 587
14.12.10 Usar con hal_glib en el handler QtVCP . . . 587
14.12.11 Ideas de proyectos . . . 588

V Advanced Topics 589

15. Cinemática 590
15.1. Introducción . . . 590
15.1.1. Articulaciones vs. Ejes . . . 590
15.2. Cinemática Trivial . . . 590
15.3. Cinemática no trivial . . . 592
15.3.1. Transformación directa . . . 592
15.3.2. Transformación inversa . . . 593
15.4. Detalles de implementación . . . 594

16. Configuración de parámetros "modificados" Denavit-Hartenberg (DH) para 595
16.1. Preludio . . . 595
16.2. General . . . 595
16.3. Parámetros DH modificados . . . 596
16.4. Parámetros DH modificados usados en *genserkins* . . . 596
16.5. Numeración de articulaciones y parámetros . . . 596
16.6. Cómo comenzar . . . 597
16.7. Casos especiales . . . 597
16.8. Ejemplo detallado (RV-6SL) . . . 597
16.9. Créditos . . . 597

17. Cinemática de 5 ejes 598
17.1. Introducción . . . 598
17.2. Configuraciones de máquinas herramienta de 5 ejes . . . 598
17.3. Orientación y localizacion de la herramienta . . . 598
17.4. Matrices de translacion y rotación . . . 599
17.5. Configuraciones de 5 ejes con mesas giratorias/inclinables . . . 600
17.5.1. Transformaciones para una máquina herramienta xyzac-trt con offsets de pieza de trabajo . . . 602
17.5.1.1. Transformación directa . . . 603
17.5.1.2. Transformación inversa . . . 605
17.5.2. Transformaciones para una máquina xyzac-trt con offsets en ejes rotativos . . . 606
17.5.2.1. Transformación directa . . . 607
17.5.2.2. Transformación inversa . . . 608
17.5.3. Transformaciones para una máquina xyzbc-trt con offsets de ejes rotativos . . . 609
17.5.3.1. Transformación directa . . . 610
17.5.3.2. Transformación inversa . . . 611
17.6. Ejemplos de mesa giratoria/inclinable . . . 612
17.6.1. Modelos de simulación de Vismach . . . 612
17.6.2. Offsets de longitud de la herramienta . . . 612
17.7. Componentes cinemáticos personalizados . . . 612
17.8. Figuras . . . 614
17.9. REFERENCIAS . . . 616

18. Ajuste PID **617**
18.1. Controlador PID . . . 617
18.1.1. Conceptos básicos de lazo de control . . . 617
18.1.2. Teoría . . . 618
18.1.3. Afinacion del lazo . . . 618

19. Offsets Externos de Ejes **620**
19.1. Configuración del archivo ini . . . 620
19.2. Pines Hal . . . 620
19.2.1. Pines Hal de movimiento por eje . . . 620
19.2.2. Otros Pines Hal Motion . . . 621
19.3. Uso . . . 621
19.3.1. Cálculo de offset . . . 621
19.3.2. Máquina apagada/encendida . . . 621
19.3.3. Límites soft . . . 621
19.3.4. Notas . . . 622
19.3.5. Advertencia . . . 622
19.4. Componentes de Hal relacionados . . . 622
19.4.1. eoffset_per_angle.comp . . . 622
19.5. Pruebas . . . 623
19.6. Ejemplos . . . 623
19.6.1. eoffsets_demo.ini . . . 623
19.6.2. jwp_z.ini . . . 624
19.6.3. dynamic_offsets.ini . . . 624
19.6.4. opa.ini (eoffset_per_angle) . . . 624

20. Intérprete independiente **625**
20.1. Uso . . . 625
20.2. Ejemplo . . . 625

VI Glosario **627**

VII Seccion Legal **634**

21. Terminos de Copyright **636**

22. Licencia GNU de Documentation Libre **637**

23. Índice alfabético **641**

Parte I

Contenido

Parte II

Acerca de LinuxCNC

Capítulo 1

Introduccion

Este manual es un trabajo en progreso. Si puedes ayudar con redacción, edición o preparación gráfica, comuníquese con cualquier miembro del equipo de redacción o únete y envía un correo electrónico a emc-users@lists.sourceforge.net.

Copyright © 2000-2016 LinuxCNC.org

Se otorga permiso para copiar, distribuir y / o modificar este documento bajo los términos de la Licencia de Documentación Libre de GNU, Versión 1.1 o cualquier versión posterior publicada por la Free Software Foundation; sin secciones invariantes, sin textos de portada y sin textos de contraportada. Se incluye una copia de la licencia en la sección titulada "GNU Licencia de documentación gratuita ".

LINUX® es la marca registrada de Linus Torvalds en los EE. UU. Y otros países. La marca registrada Linux® se utiliza de conformidad con una sublicencia de LMI, el licenciatario exclusivo de Linus Torvalds, propietario de la marca en un base mundial.

El proyecto LinuxCNC no está afiliado a Debian®. *Debian* es una marca registrada propiedad de Software in the Public Interest, Inc.

El proyecto LinuxCNC no está afiliado a UBUNTU®. *UBUNTU* es una marca registrada propiedad de Canonical Limited.

This handbook is a work in progress. If you are able to help with writing, editing, or graphic preparation please contact any member of the writing team or join and send an email to emc-users@lists.sourceforge.net.

Copyright © 2000-2020 LinuxCNC.org

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

LINUX® is the registered trademark of Linus Torvalds in the U.S. and other countries. The registered trademark Linux® is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a world-wide basis.

The LinuxCNC project is not affiliated with Debian®. *Debian* is a registered trademark owned by Software in the Public Interest, Inc.

The LinuxCNC project is not affiliated with UBUNTU®. *UBUNTU* is a registered trademark owned by Canonical Limited.

Capítulo 2

LinuxCNC History

2.1. Origen

EMC (controlador de máquina mejorado) fue creado por el NIST, el Instituto Nacional de Normas y Tecnología, que es una agencia del Departamento de Comercio del Gobierno de los Estados Unidos.

El NIST primero se interesó en escribir un paquete de control de movimiento para plataforma de prueba de conceptos y estándares. El patrocinio de General Motors resultó en una adaptación de la versión incipiente de EMC usando tarjetas de control inteligentes PMAC, ejecutandose en una versión "en tiempo real" de Windows NT y controlando una gran fresadora.

Todos los *productos de trabajo* de los empleados del gobierno federal de EE.UU., el software resultante y el informe al respecto deben ponerse en el dominio público y, por ello, fué debidamente publicado un informe al respecto, incluso en Internet. Fue allí donde Matt Shaver descubrió EMC. Contactó con el NIST y entró en conversaciones con Fred Proctor sobre la adaptación del código para utilizarlo para controlar hardware menos costoso que se utilizaria para actualizaciones y retrofits de controles CNC que estaban obsoletos o simplemente muertos. NIST quedo intrigado porque ellos también querían algo menos costoso. Para lanzar este esfuerzo cooperativo, se creó un acuerdo formal que garantizó que el código resultante y el diseño permanecerían en el dominio público.

Las primeras consideraciones se centraron en reemplazar el costoso y temperamental Sistema de Windows NT "en tiempo real". Se propuso que se intentará (en aquel momento) con una relativamente nueva extensión en tiempo real del sistema operativo Linux. Esta idea fue llevada a cabo con éxito. Lo siguiente fue el tema de las costosas tarjetas inteligentes de control de movimiento. En esas fechas, el poder de procesamiento de una PC era considerado lo suficientemente grande como para tomar el control directo de las rutinas de movimiento. Una búsqueda rápida de hardware disponible resultó en la selección de las tarjetas de interfaz Servo-To-Go como la primera plataforma para permitir que un PC controle directamente los motores. Se agregó a la interfaz de usuario existente software para planificación de trayectoria, control de bucle PID y un intérprete RS274. Matt utilizó con éxito esta versión para actualizar un par de máquinas con controles muertos y este se convirtió en el sistema EMC que primero llamó la atención del mundo exterior. La mención de EMC en el el grupo de noticias USENET rec.crafts.metalworking resultó en los primeros adoptantes , como Jon Elson, constructores de sistemas para aprovechar EMC.

NIST creó una lista de correo para las personas interesadas en EMC. Con el paso del tiempo, otros, ademas del NIST, se interesaron en mejorar EMC. Mucha gente solicitó pequeñas mejoras en el código. Ray Henry quería refinar el interfaz de usuario. Como Ray era reacio a intentar alterar el código C en el que se escribió la interfaz, se buscó un método más simple. Fred Proctor, del NIST, sugirió un lenguaje de secuencias de comandos y escribió código para interactuar con el lenguaje de script Tcl/Tk para las comunicaciones internas NML de EMC. Con esta herramienta Ray pasó a escribir un programa Tcl/Tk que se convirtió en la interfaz de usuario predominante para EMC en el momento.

Desde la perspectiva del NIST, vea esto paper escrito por William Shackleford y Frederick Proctor, describiendo la historia de EMC y su transición a código abierto.

En este momento, el interés en EMC comienza a subir sustancialmente. Cuantas más y más personas intentaron instalar EMC, la dificultad de parchear un kernel de Linux con las extensiones en tiempo real y compilar el código EMC se volvió notoriamente obvia. Hubo muchos intentos de documentar el proceso y se intentaron escribir scripts, algunos con éxito moderado. El problema de hacer coincidir la versión correcta de los parches y compiladores con la versión seleccionada de Linux siguia apareciendo. Paul Corner vino al rescate con el BDI (Brain Dead Install) que era un CD con el cual podían instalarse un sistema de trabajo

completo (Linux, parches y EMC). El enfoque BDI abrió el mundo de EMC a una comunidad de usuarios mucho más grande. A medida que esta comunidad continuó creciendo, la lista de correo de EMC y los archivos de código se movieron a SourceForge y se estableció el sitio web LinuxCNC.

Con una comunidad más grande de usuarios participando, EMC se convirtió en un foco principal de interés en las exposiciones CNC en curso en NAMES y NAMES se convirtió en el evento anual de reunión para EMC. Durante los primeros años, las reuniones solo sucedieron porque las partes interesadas estaban en NAMES. En 2003, la comunidad de usuarios de EMC tuvo su primera reunión pública anunciada. Se celebró el lunes después de NAMES en el vestíbulo donde se celebró el espectáculo NAMES. La organización era floja, pero nació la idea de una capa de abstracción de hardware (HAL) y fue propuesto el movimiento para reestructurar el código para facilitar el desarrollo (EMC2) .

2.2. Cambio de nombre

En la primavera de 2011, la Junta Directiva de LinuxCNC fue contactada por una firma de abogados que representa a EMC Corporation (www.emc.com) sobre el uso de "EMC" y "EMC2" para identificar el software ofrecido en linuxcnc.org. EMC Corporation ha registrado varias marcas comerciales relacionadas con EMC y EMC^2 (EMC con numeral superíndice dos).

Después de varias conversaciones con el representante de EMC Corporation, el resultado final fue que, comenzando con el próximo gran lanzamiento del software, linuxcnc.org dejará de identificarlo usando "emc" o "EMC", o aquellos términos seguidos de dígitos. En la medida que la Junta Directiva de LinuxCNC controla los nombres utilizados para identificar el software ofrecido en linuxcnc.org, la junta ha aceptado esto.

Como resultado, fue necesario elegir un nuevo nombre para el software. De las opciones que la junta consideró, hubo consenso de que "LinuxCNC" era la mejor opción, ya que ese ha sido el nombre de nuestro sitio web durante años.

En preparación para el nuevo nombre, hemos recibido una sublicencia de Marca registrada LINUX® de la Fundación Linux (www.linuxfoundation.org), protegiendo nuestro uso del nombre LinuxCNC. (LINUX® es la marca registrada de Linus Torvalds en los EE. UU. y otros países).

El esfuerzo de cambio de marca incluirá el sitio web linuxcnc.org, canales IRC y versiones del software y la documentación, comenzando con la 2.5.0.

2.3. Información adicional

NIST publicó un artículo que describe el idioma RS274NGC y el resumen del centro de mecanizado que controla, así como una implementación temprana de EMC. El documento también está disponible en http://linuxcnc.org/files/RS274NGCv3.pdf

NIST también publicó un artículo sobre la historia de EMC y sus transición a open source. El documento también está disponible en http://linuxcnc.org/files/Use-of-Open-Source-Distribution-for-a-Machine-Tool-Controller.pdf

Parte III

Usando LinuxCNC

Capítulo 3

Informacion General

3.1. Prefacio para el usuario

LinuxCNC es modular y flexible. Estos atributos llevan a muchas personas a verlo como un revoltijo confuso de pequeñas cosas y se preguntan por qué es así. Esta página intenta responder esa pregunta antes de entrar en las profundidades del asunto.

LinuxCNC comenzó en el Instituto Nacional de Estándares y Tecnología (NIST) en EE.UU. Creció usando Unix como su sistema operativo. Unix lo hizo diferente. Entre los primeros desarrolladores de Unix creció un conjunto de ideas de escritura de código que algunos llaman "la manera Unix". Estos primeros autores de LinuxCNC siguieron esas maneras.

Eric S. Raymond, en su libro The Art of Unix Programming, resume la filosofía Unix como una filosofía de ingeniería ampliamente utilizada, "Mantenlo simple, estúpido "("Keep it Simple, Stupid" o Principio KISS). Luego describe cómo cree que esta filosofía general se aplica como una norma cultural en Unix aunque, como era de esperar, no es difícil encontrar violaciones graves de la mayoría de las siguientes prácticas en la realidad de Unix:

- Regla de modularidad: escribir partes simples conectadas por interfaces claras.
- Regla de claridad: la claridad es mejor que el ingenio.
- Regla de composición: diseñar programas para conectarse a otros programas.
- Regla de separación: Separar las políticas de los mecanismos; separar interfaces de motores.[1]

El Sr. Raymond ofreció varias reglas más, pero estas cuatro describen las características esenciales del sistema de control de movimiento LinuxCNC.

La regla de **modularidad** es crítica. A lo largo de estos manuales encontrará grandes parrafos sobre el intérprete o el planificador de tareas o el de movimiento o sobre HAL. Cada uno de estos es un módulo o colección de módulos. Es esta modularidad la que permite conectar solo las partes que usted necesita para hacer funcionar su máquina (n.t. sin llenar su pantalla de florituras, ocupar su CPU en mantenerlas en funcionamiento ni darle mas oportunidades que las precisas a Murphy y sus reglas).

La regla **Claridad** tambien es esencial. LinuxCNC es un trabajo en progreso, esta sin terminar ni lo estara jamás. Es lo suficientemente completo como para manejar la mayoría de tipos de máquinas a las que va destinado. Gran parte de ese progreso se logra porque muchos usuarios y desarrolladores de código pueden ver el trabajo de otros y construir cosas nuevas sobre lo ya hecho.

La regla **Composicion** nos permite construir un sistema de control predecible con muchos de los módulos disponibles, que son conectables. La conectividad se logra configurando interfaces estándar para conjuntos de módulos y siguiendo esos estándares.

La regla **Separación** requiere que se construyan partes distintas que hagan cosas pequeñas. Separando funciones, la depuración es mucho más fácil y los reemplazos de módulos se pueden introducir en el sistema y hacer comparaciones fácilmente.

¿Qué significa "la manera Unix" para usted como usuario de LinuxCNC?. Significa que usted puede tomar decisiones sobre cómo usará el sistema. Muchas de estas elecciones se haran durante la integración de la máquina, pero muchas también afectaran a la

[1] Encontrado en http://en.wikipedia.org/wiki/Unix_philosophy, 07/06/2008

forma en que se usará la máquina. Durante la lectura de la documentacion, encontrará muchos lugares donde tendrán que hacer comparaciones. Eventualmente se deberan tomar decisiones, "Usaré esta interfaz en lugar de esa otra" o "Voy a escribir offsets de piezas de esta manera en lugar de hacerlo de esa otra". A lo largo de estos manuales describimos el rango de posibilidades disponibles actualmente.

Al comenzar su viaje hacia el uso de LinuxCNC, le ofrecemos dos consideraciones:[2]

- Parafraseando las palabras de Doug Gwyn en UNIX: "LinuxCNC no fue diseñado para evitar que sus usuarios hagan cosas estúpidas, ya que eso también les impediria hacer cosas ingeniosas".
- Tambien las palabras de Steven King: "LinuxCNC es fácil de usar. Simplemente no se descuida con respecto a qué tipo de usuarios es amigable".

3.2. Introduccion para usuarios de LinuxCNC

3.2.1. Como funciona LinuxCNC

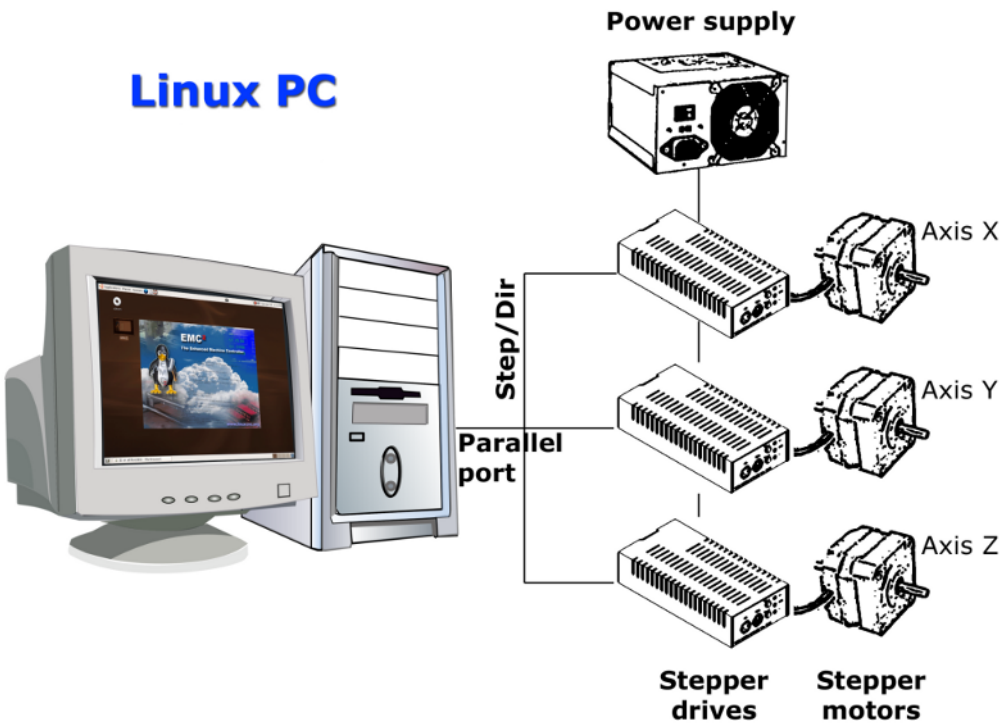
LinuxCNC es un conjunto de aplicaciones altamente personalizables para el control de fresadoras y tornos de control numerico mediante una computadora (CNC), impresoras 3D, robots, cortadores láser, cortadores de plasma y otros dispositivos automatizados. Es capaz de proporcionar el control coordinado de hasta 9 ejes de movimiento.

En esencia, LinuxCNC consta de varios componentes clave que se integran para formar un sistema completo:

- una interfaz gráfica de usuario (GUI), que forma la interfaz básica entre el operador, el software y la máquina CNC en sí misma;
- una capa de abstracción de hardware (HAL), que proporciona un método para vincular las diversas señales virtuales internas generadas y recibidas por LinuxCNC con el mundo exterior.
- controladores de alto nivel que coordinan la generación y ejecución del control de movimiento de la máquina CNC, es decir,el controlador de movimiento (EMCMOT), el controlador de entrada/salida discretas (EMCIO) y el ejecutor de tareas (EMCTASK).

La siguiente ilustración es un diagrama de bloques simple que muestra una fresadora típica CNC de 3 ejes con motores paso a paso:

[2] Encontrado en http://en.wikipedia.org/wiki/Unix_philosophy, 07/06/2008



El ordenador que ejecuta LinuxCNC envía una secuencia de impulsos a través del puerto paralelo a las unidades paso a paso. Cada una de esas unidades tiene un motor paso a paso conectado. Cada unidad recibe dos señales independientes; una señal para ordenar al accionamiento que mueva su motor paso a paso asociado en sentido horario o antihorario (señal de direccion), y una segunda señal (tren de pulsos) que define la velocidad a la que gira el motor paso a paso.

Se ha ilustrado un sistema de motor paso a paso bajo control del puerto paralelo, pero un sistema LinuxCNC también puede conectarse a una amplia variedad de interfaces hardware dedicadas de control de movimiento para una mayor velocidad o capacidades de E/S. Se puede encontrar una lista completa de interfaces compatibles con LinuxCNC en la página Wiki Hardware soportado

En la mayoría de los casos, los usuarios crearán una configuración específica para la configuración de su maquina utilizando el Asistente de Configuracion de Steppers (para sistemas CNC que funcionan con puerto paralelo) o el Asistente de Hardware Mesa (para sistemas más avanzados que utilizan tarjetas PCI Mesa Anything I/O). La ejecución de cualquiera de los asistentes creará varias carpetas en el disco duro del ordenador que contiene una cierta cantidad de archivos de configuración específicos para esa máquina CNC, y un ícono colocado en el escritorio para permitir el lanzamiento fácil de LinuxCNC.

Por ejemplo, si se usó el Asistente de Configuración de Steppers para crear una configuración para la fresadora CNC de 3 ejes ilustrada mas arriba y se titulo *Mi_CNC*, las carpetas creadas por el asistente contendrían generalmente los siguientes archivos:

- Carpeta: Mi_CNC
  - Mi_CNC.ini
    - El archivo INI contiene toda la información básica del hardware relacionado con el funcionamiento de la fresadora CNC, tales como el número de pasos que debe realizar cada motor paso a paso para completar una revolución completa, la velocidad máxima en que cada paso a paso puede operar, los límites de recorrido de cada eje o la configuración y el comportamiento de interruptores de límite en cada eje.
  - Mi_CNC.hal
    - Este archivo HAL contiene información que le dice a LinuxCNC cómo vincular las señales virtuales internas a conexiones físicas exteriores a la computadora. Por ejemplo, especificando el pin 4 en el puerto paralelo para enviar la señal de dirección de paso del eje Z, o instruir a LinuxCNC para dejar de mover el motor del eje X cuando se activa el interruptor de límite conectado al pin 13 del puerto paralelo.
  - custom.HAL

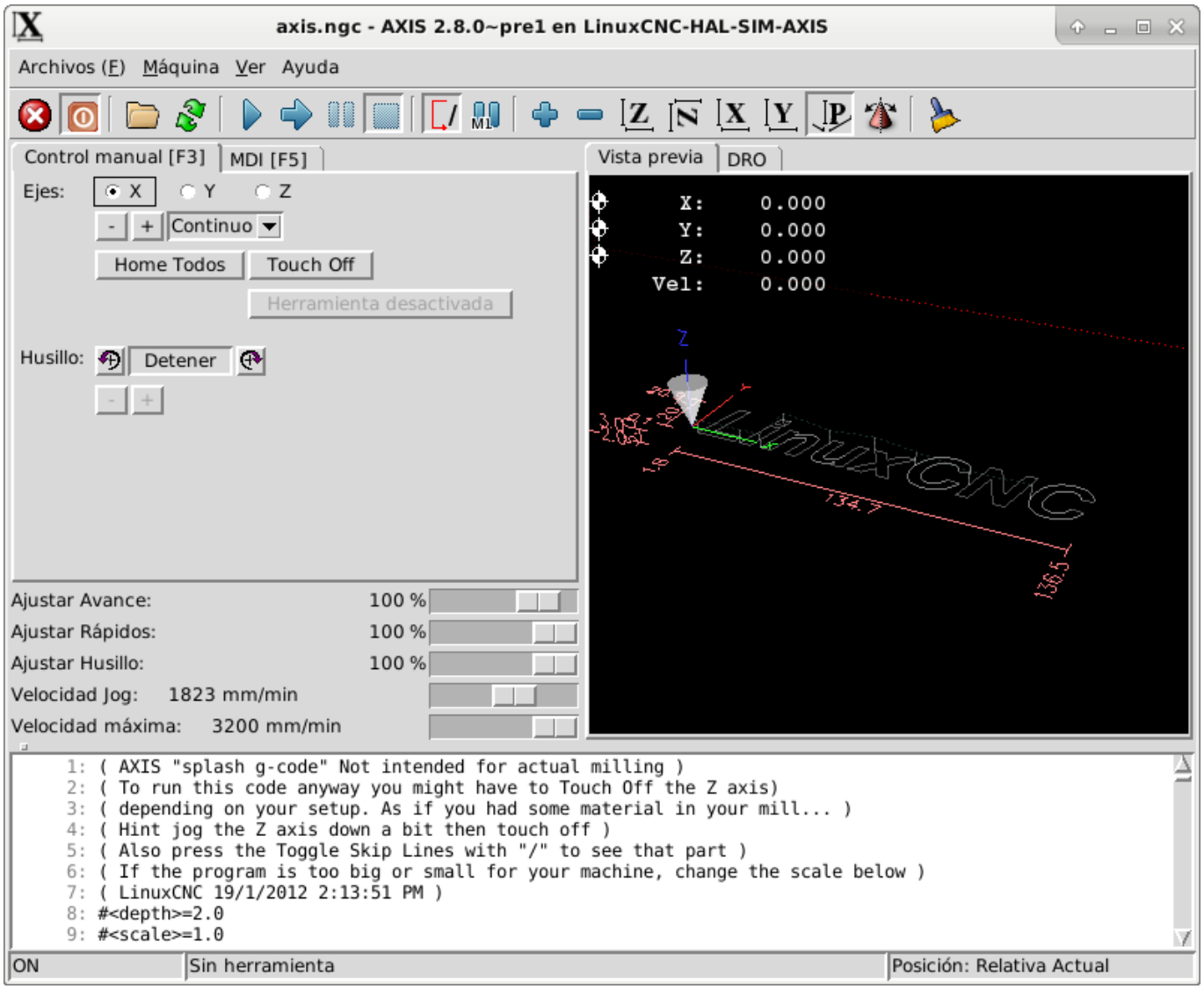
  - Las personalizaciones para la configuración de la fresadora, más allá del alcance del asistente, pueden realizarse incluyendo enlaces adicionales a otros puntos virtuales dentro de LinuxCNC en este archivo HAL. Al iniciar una sesión de LinuxCNC, este archivo se lee y se procesa antes de cargar la GUI. Por ejemplo, puede incluir iniciar las comunicaciones Modbus con el driver del husillo para que se confirme como operativo antes de que se muestre la GUI.
- custom_postgui.hal
  - El archivo HAL custom_postgui permite una mayor personalización de LinuxCNC, pero difiere de custom.HAL en que se procesa después de que se muestre la GUI. Por ejemplo, después de establecer comunicaciones Modbus al driver del motor del husillo en custom.hal, LinuxCNC puede usar el archivo custom_postgui para vincular la lectura de la velocidad del husillo desde el driver del motor con un gráfico de barras que se muestra en la GUI.
- postgui_backup.hal
  - Proporciona una copia de seguridad del archivo custom_postgui.hal para permitir al usuario restaurar rápidamente una configuración HAL postgui que funcionaba anteriormente. Esto es especialmente útil si el usuario desea ejecutar el Asistente de configuración nuevamente bajo el mismo nombre, *Mi_CNC*,para modificar algunos parámetros. Al guardar la nueva configuración con el asistente, se sobrescribirá el archivo custom_postgui existente mientras que el archivo postgui_backup queda intacto.
- tool.tbl
  - Un archivo de tabla de herramientas contiene una lista parametrizada de cualquier herramienta de corte utilizada. Estos parámetros puede incluir el diámetro y la longitud del cortador, y se utiliza para proporcionar un catálogo de datos que le dice a LinuxCNC cómo compensar su movimiento para herramientas de diferentes tamaños dentro de una operación de fresado.

- Carpeta: nc_files
  - La carpeta nc_files se proporciona como la ubicación predeterminada para almacenar los programas de código G utilizados en la maquina. También incluye varias subcarpetas con ejemplos de código G.

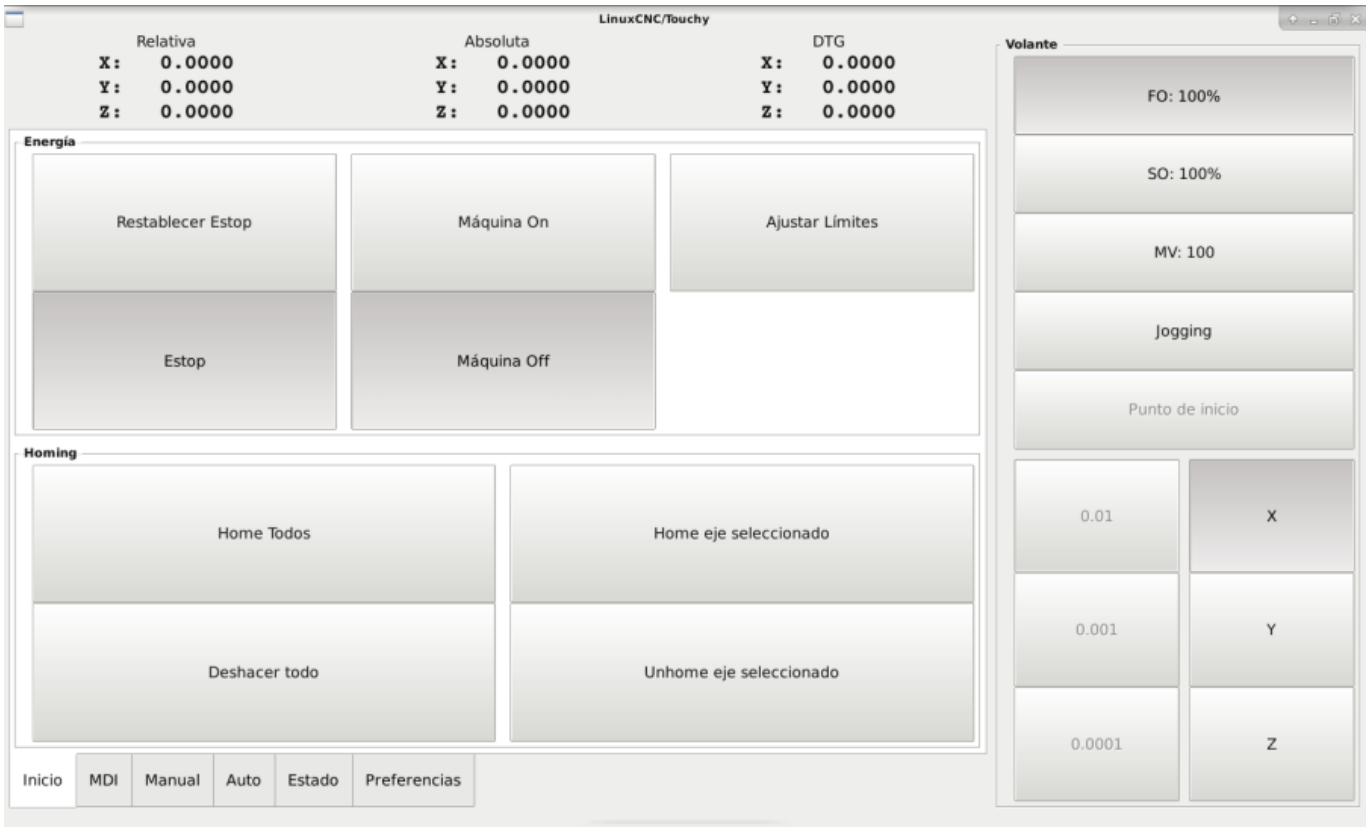
3.2.2. Interfaces graficas de usuario

Una interfaz gráfica de usuario es la parte del LinuxCNC con la que interactúa el operador de la máquina herramienta. LinuxCNC viene con varios tipos de interfaces de usuario que se pueden elegir al editar ciertos campos contenidos en el <<cha:ini-configuration,archivo INI:

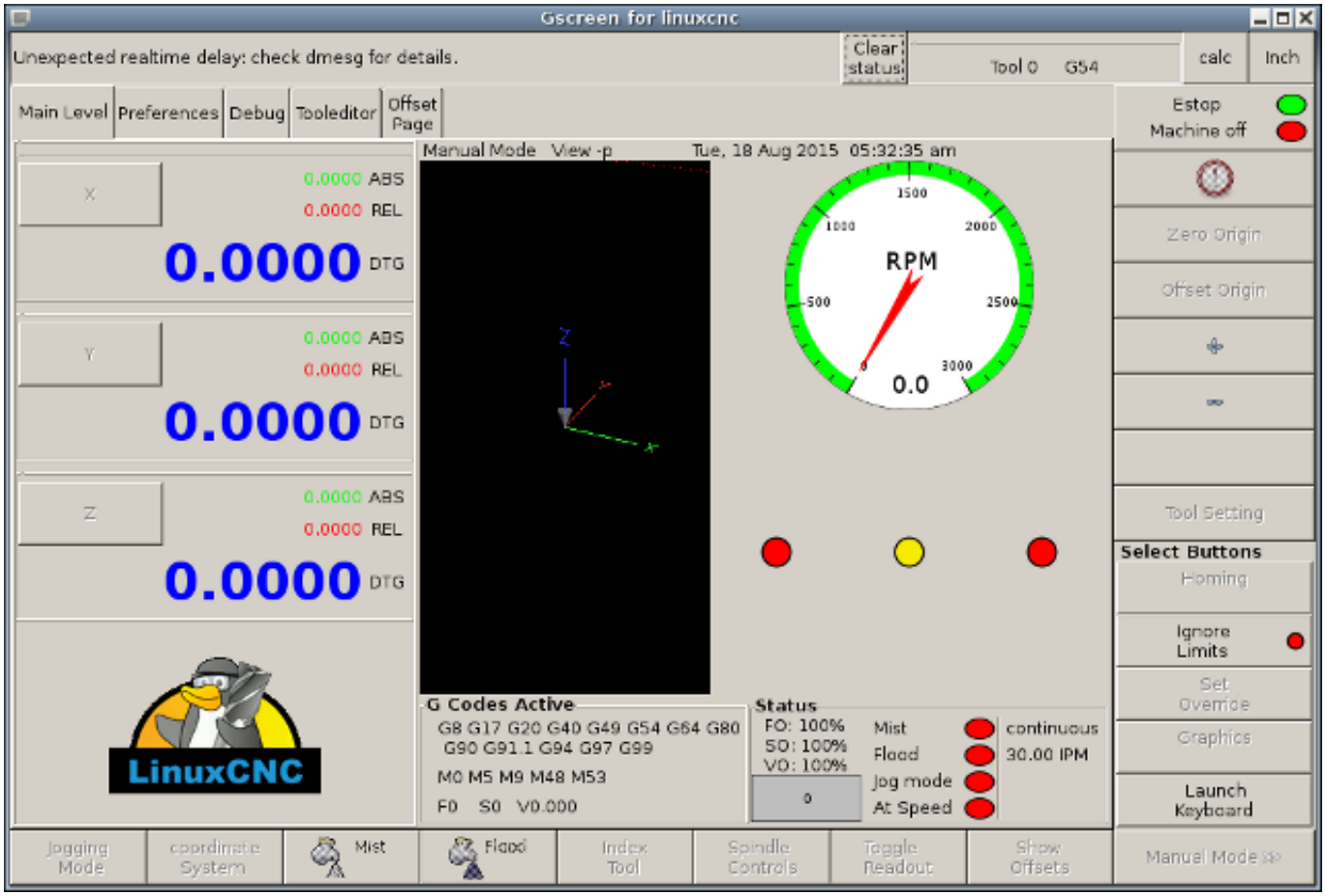
Axis, es la interfaz GUI de teclado estándar. Es también la GUI predeterminada que se inicia cuando se usa el Asistente de Configuración para crear un icono lanzador en escritorio:



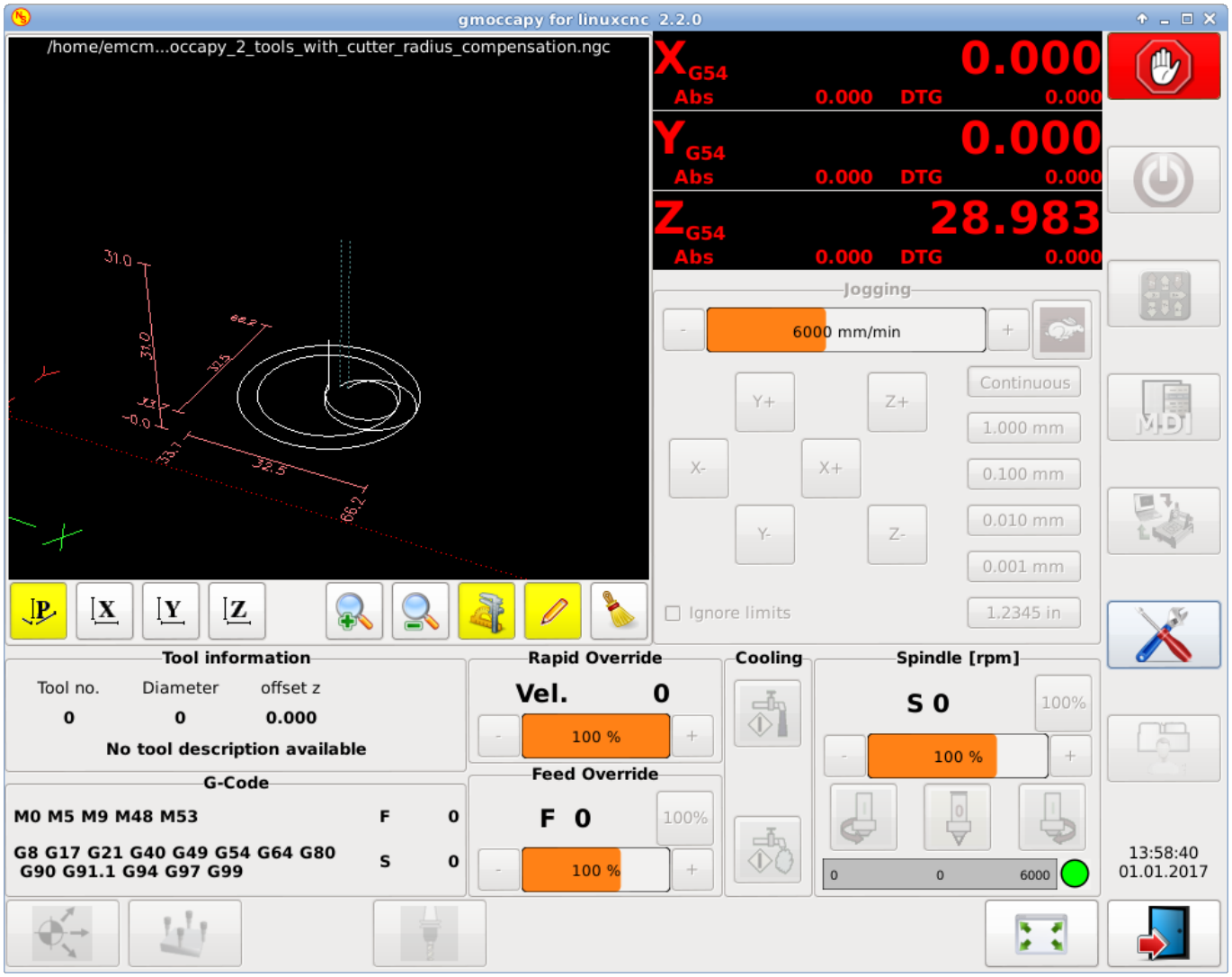
Touchy es una GUI para usar con pantalla táctil:



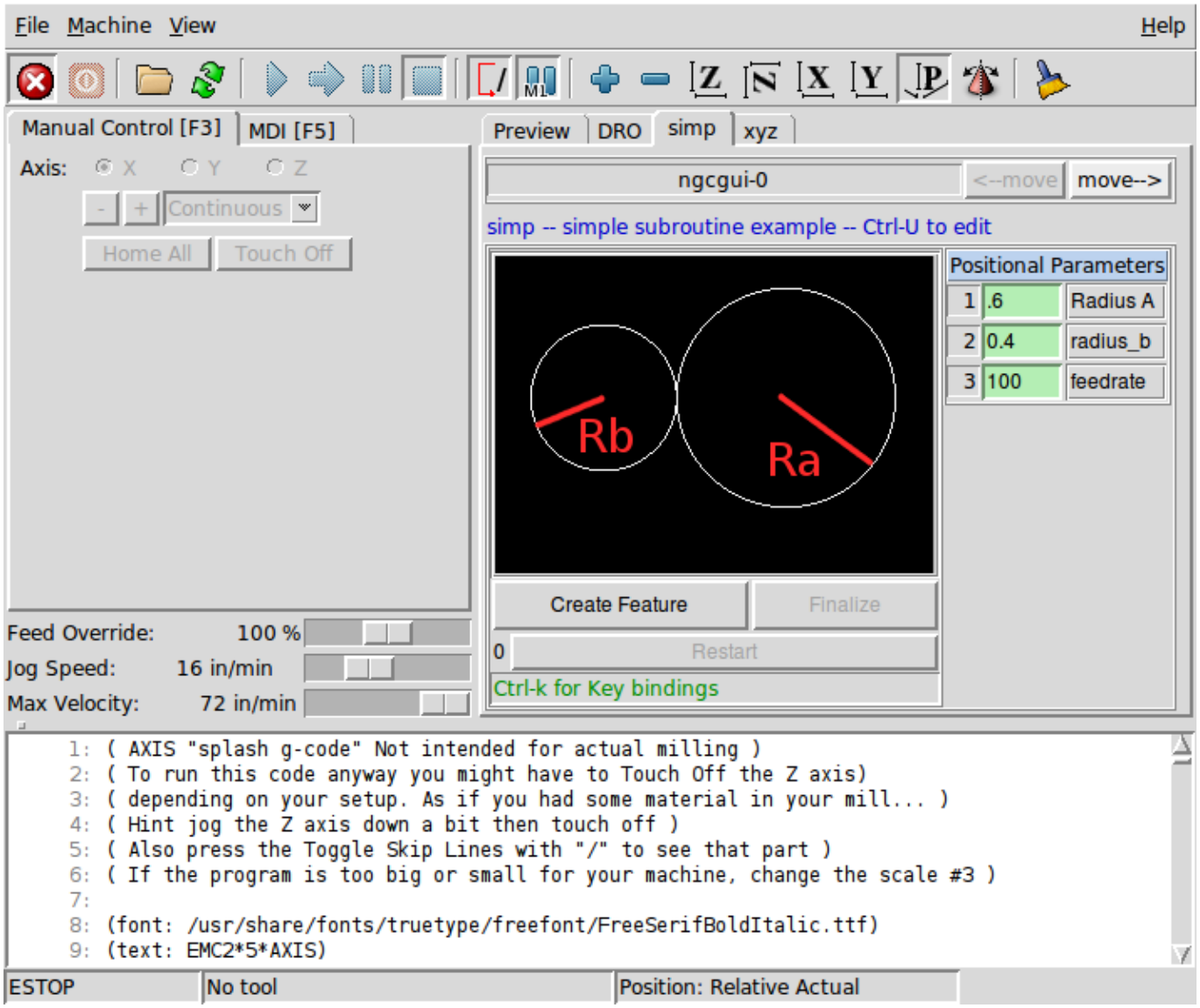
Gscreen una GUI de pantalla táctil configurable por el usuario:



GMOCCAPY una GUI de pantalla táctil basada en Gscreen. GMOCCAPY también está diseñada para funcionar tambien en aplicaciones donde el teclado y el mouse son los métodos preferidos para controlar la GUI:



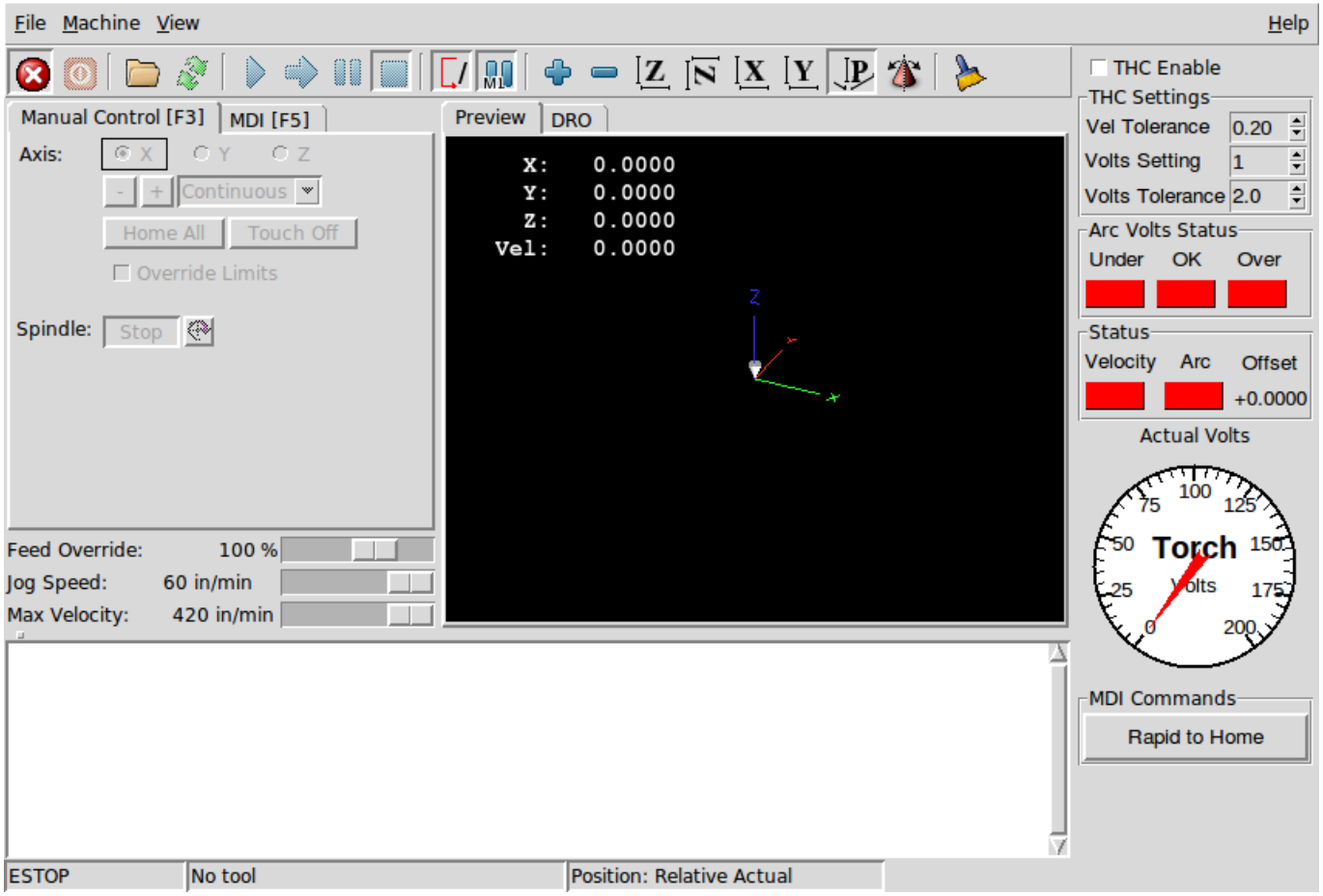
NGCGUI una GUI de subrutinas que proporciona una programación de código G, de tipo asistente. NGCGUI puede ejecutarse como un programa independiente o incrustado en otra GUI como una serie de pestañas. La siguiente captura de pantalla muestra a NGCGUI incrustado en Axis:



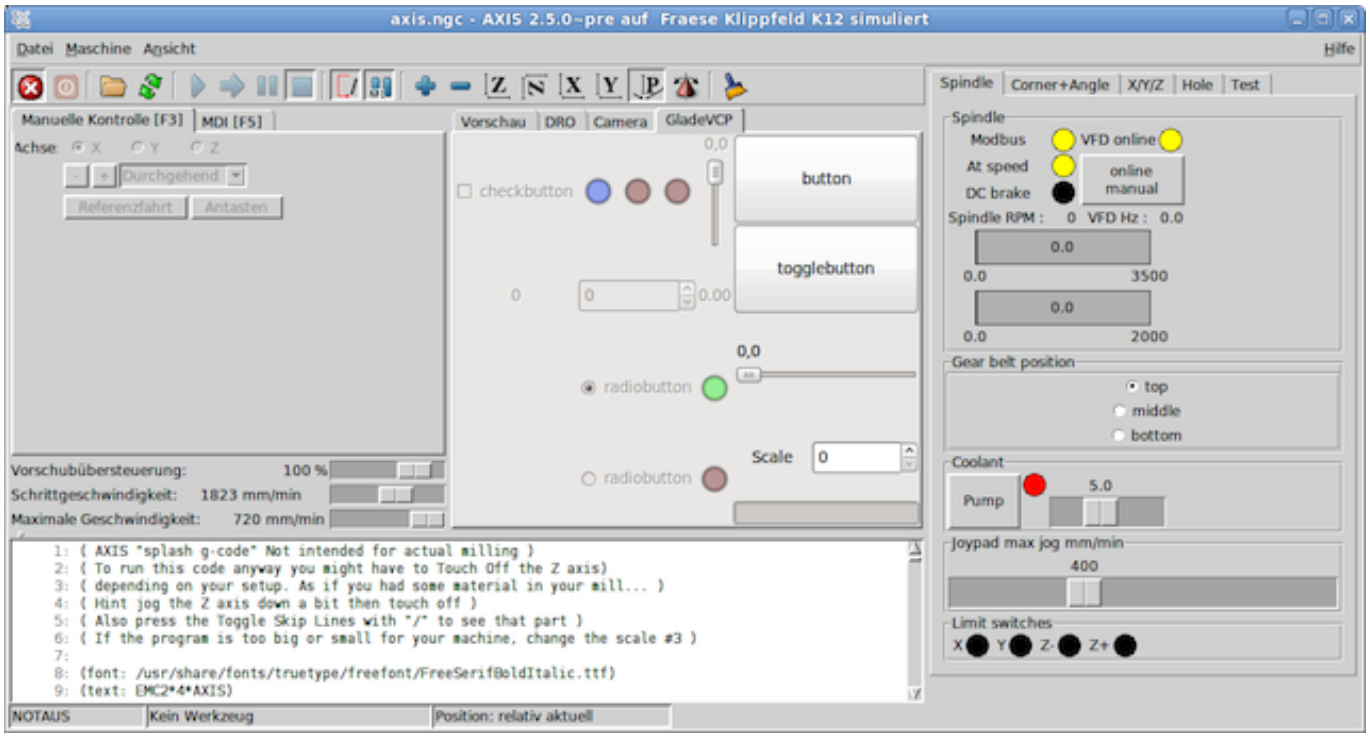
3.2.3. Paneles de control virtual

Como se mencionó anteriormente, muchas de las GUI de LinuxCNC pueden ser personalizadas por el usuario. Esto se puede hacer para agregar indicadores, salidas de lectores, interruptores o controles deslizantes a la apariencia básica de una de las GUI para aumentar su flexibilidad o funcionalidad. Se ofrecen dos estilos de Paneles de Control Virtual en LinuxCNC:

PyVCP un panel de control virtual basado en Python que se puede agregar a la GUI Axis. PyVCP utiliza solo señales virtuales contenidas dentro de la capa de abstracción de hardware HAL, como el indicador de velocidad del husillo o la señal de salida de Parada de Emergencia, y tiene una apariencia sencilla, sin lujos. Esto lo hace una excelente opción si el usuario desea agregar un Panel de Control Virtual con un mínimo esfuerzo.



GladeVCP un panel de control virtual basado en Glade que se puede agregar a las GUIs Axis o Touchy. GladeVCP tiene la ventaja sobre PyVCP en que no se limita a la visualización o control de señales virtuales HAL, sino que puede incluir otras interfaces externas fuera de LinuxCNC, como ventanas o eventos de redes. GladeVCP también es más flexible en cuanto a cómo se puede configurar para que aparezca en la GUI:



3.2.4. Idiomas

LinuxCNC utiliza archivos de traducción para traducir las interfaces de usuario de LinuxCNC a muchos idiomas, incluidos francés, español, alemán, italiano, finlandés, ruso, rumano, portugués y chino. Suponiendo que una traducción ha sido creada, LinuxCNC usará automáticamente cualquier idioma nativo con el que inicie la sesión en el sistema operativo Linux. Si su idioma no ha sido traducido, contacte a un desarrollador en el IRC, la lista de correo o el Foro de usuarios para obtener ayuda.

3.2.5. Modos de operacion

Cuando LinuxCNC se está ejecutando, hay tres modos principales, diferentes entre si, utilizados para ingresar comandos. Son los modos manual, automático y MDI (entrada manual de datos). Al cambiar de un modo a otro, hay una gran diferencia en la forma en que se comporta LinuxCNC. Hay cosas específicas que se pueden hacer en un modo y que no pueden hacerse en otro. Un operador puede hacer home en un eje en modo manual pero no en modo automático o MDI, o puede hacer que la máquina ejecute un archivo completo de código G en el modo automático pero no en manual o MDI.

En el modo manual, cada comando se ingresa por separado. En términos humanos, un comando manual podría ser encender el refrigerante o mover el eje X a 25 pulgadas por minuto. Estos serian, más o menos, equivalentes a pulsar un interruptor o girar el volante de un eje. Estos comandos se manejan normalmente en una de las interfaces gráficas, presionando un botón con el mouse o presionando una tecla en el teclado. En el modo automático, un botón similar o presionar una tecla podría usarse para cargar o iniciar la ejecución de un programa completo de código G almacenado en un archivo. En el Modo MDI, el operador puede escribir un bloque de código y decirle a la máquina que lo ejecute al presionar en el teclado <return> o <enter>.

Algunos comandos de control de movimiento están disponibles en todo momento y causarán los mismos cambios de movimiento en todos los modos. Estos incluyen Abort (abortar), Emergency Stop (parada de emergencia) y Feed Rate Override (mando manual de alimentacion). Comandos como estos deben ser en si mismos explicativos.

La interfaz de usuario AXIS oculta algunas de las distinciones entre Auto y los otros modos, haciendo disponibles comandos Auto en muchas ocasiones. También difumina la distinción entre Manual y MDI porque algunos de los comandos manuales como Touch Off se implementan realmente mediante el envío de comandos MDI. Esto lo hace cambiando automáticamente al modo que se necesite para la acción que el usuario ha solicitado.

3.3. Conceptos importantes para el usuario

Este capítulo cubre conceptos importantes de que deben ser entendidos por el usuario antes de intentar manejar una máquina CNC con código g.

3.3.1. Control de Trayectoria

3.3.1.1. Planificacion de trayectoria

La planificación de trayectorias, en general, es el medio por el cual LinuxCNC sigue la ruta especificada por su programa de Código G, mientras opera dentro de los límites de la maquina.

Un programa de Código G nunca puede ser completamente obedecido. Por ejemplo, imagine que especifica un programa de solo de línea con el siguiente movimiento:

```
G1 X1 F10 (G1 es movimiento lineal, X1 es el destino, F10 es la velocidad)
```

En realidad, todo el movimiento no se puede hacer a F10, ya que la máquina debe acelerar desde el reposo, avanzar hacia X=1, y luego desacelerar para parar de nuevo. A veces, parte del movimiento se realiza a F10, pero para muchos movimientos, especialmente los cortos, nunca se alcanza en absoluto la velocidad de avance especificada. Tener movimientos cortos en su Código G puede hacer que su máquina se ralentice y luego acelere para los movimientos más largos si no se emplea *naive cam detector* con G64 Pn.

La aceleración y desaceleración básica descrita anteriormente no es compleja y no hay ningún compromiso que tomar. En el archivo INI se han especificado las restricciones de la máquina, como la velocidad y aceleración máxima de ejes, que deben ser obedecidas por el planificador de trayectoria.

Para obtener más información sobre las opciones ini del planificador vea la <<sec:traj-section,Sección Trayectoria en el capítulo INI.

3.3.1.2. Seguimiento de ruta

Un problema menos sencillo es el seguimiento de ruta. Cuando se programa una esquina en Código G, el planificador de trayectoria puede hacer varias cosas, todas ellas correctas según el caso:

- Puede desacelerar hasta detenerse exactamente en las coordenadas de la esquina, y luego acelerar en la nueva dirección.
- También puede hacer lo que se llama fusión (blending), que consiste en mantener la velocidad de avance mientras pasa por la esquina, por lo que es necesario redondear dicha esquina para obedecer las restricciones de la máquina.

Puede ver que aquí si hay un compromiso; se puede reducir la velocidad para obtener un mejor seguimiento del camino, o mantener la velocidad con peor seguimiento. Dependiendo del corte particular, el material, las herramientas, etc, el programador puede querer comprometerse de una manera u otra.

Los movimientos rápidos también obedecen el control de trayectoria actual. Con movimientos suficientemente largos para alcanzar la velocidad máxima en una máquina con baja aceleración y si no se especifica la tolerancia de ruta, puede obtener una esquina bastante redonda.

3.3.1.3. Programacion del planificador

Los comandos de control de trayectoria son los siguientes:

- *G61* - (Modo de ruta exacta) visita el punto programado exactamente, aunque eso signifique que la maquina podría detenerse temporalmente para cambiar de dirección hacia el siguiente punto programado.
- *G61.1* - (Modo de parada exacta) le dice al planificador que se detenga exactamente en cada final de segmento.
- *G64* - (Fusión sin modo de tolerancia) G64 es la configuración predeterminada cuando se inicia LinuxCNC. G64 es simplemente fusión sin *naive cam detector* habilitado. G64 y G64 P0 le dicen al planificador que sacrifique la precisión de seguimiento de ruta para mantener la velocidad de avance. Esto es necesario para algunos tipos de materiales o herramientas donde las paradas exactas son dañinas y puede funciona bien siempre que el programador no olvide que la ruta de la herramienta será algo más curvilínea que lo que especifica el programa. Al usar movimientos G0 (rápidos) con G64, tenga cuidado con los movimientos a puntos seguros (clareance) y permita suficiente distancia para evitar obstáculos en función de las capacidades de aceleración de su máquina.
- *G64 P- Q-* - (Fusión con modo de tolerancia) Esto habilita *naive cam detector* y permite fusiones con tolerancia. Si se programa, por ejemplo, G64 P0.05, se le dice al planificador que se desea alimentación continua, pero en esquinas programadas quiere que desacelere lo suficiente para que la ruta de la herramienta pueda mantenerse a no mas de 0.05 unidades de usuario de la ruta programada. La cantidad exacta de desaceleración depende de la geometría de la esquina programada y las limitaciones de la máquina, pero lo único de lo que el programador necesita preocuparse es de la tolerancia. Esto le da al programador control total sobre el compromiso de seguimiento de camino. La tolerancia de la fusión se puede cambiar a lo largo de todo el programa según sea necesario. Tenga en cuenta que una especificación G64 P0 tiene el mismo efecto que G64 solo, necesario para compatibilidad con los viejos programas de G Code. Vea la <<gcode:g64,sección G64 del capítulo de Código G.
- *Fusión sin tolerancia* - El punto controlado tocará cada movimiento especificado en al menos un punto. La máquina nunca se moverá a una velocidad tal que no pueda llegar a una parada exacta al final del movimiento actual (o el próximo movimiento, si hace una pausa cuando la fusión ya ha comenzado). La distancia desde el punto final del movimiento sera tan grande como deba ser para mantener la mejor alimentación para el contorno.
- *Naive Cam Detector* - Varios movimientos G1 sucesivos que involucran solo a los ejes XYZ y que se desvíen menos de una cantidad Q- de una línea recta que los contenga, se fusionaran en una única línea recta. Este movimiento combinado reemplaza los movimientos individuales G1 cuando se quiere fusionar con tolerancia. Entre movimientos sucesivos, el punto controlado no se desviara más de P- desde los puntos finales reales de los movimientos. El punto controlado tocará al menos un punto en cada movimiento. La máquina nunca se moverá a una velocidad tal que no puede llegar a una parada exacta al final del movimiento actual (o siguiente movimiento, si hace una pausa cuando la fusión ya ha comenzado). En movimientos G2/G3,

que se muevan en el plano G17 (XY), cuando la desviación máxima de un arco desde un línea recta es menor que la tolerancia Q-, el arco se divide en dos líneas (desde el inicio del arco hasta el punto medio y desde el punto medio hasta el final). Esas líneas están entonces sujetas al algoritmo *Naive Cam Detector* para líneas. Por lo tanto, casos de arco a línea, arco a arco y línea a arco, así como línea a línea, podrán beneficiarse de *naive cam detector*. Esto mejora el rendimiento de contorneado simplificando el camino.

En la siguiente figura, la línea azul representa la velocidad real de la máquina. Las líneas rojas son la capacidad de aceleración de la máquina. Las líneas horizontales debajo de cada grafico son el movimiento planificado. La parte superior de la gráfica muestra cómo el planificador de trayectoria ralentizará la máquina cuando se encuentren movimientos cortos para mantenerse dentro de los límites de los ajuste de aceleración de la máquina para poder detenerse exactamente al final del próximo movimiento. La gráfica inferior muestra el efecto de Naive Cam Detector al combinar los movimientos y hacer un mejor trabajo manteniendo el velocidad según lo planeado.

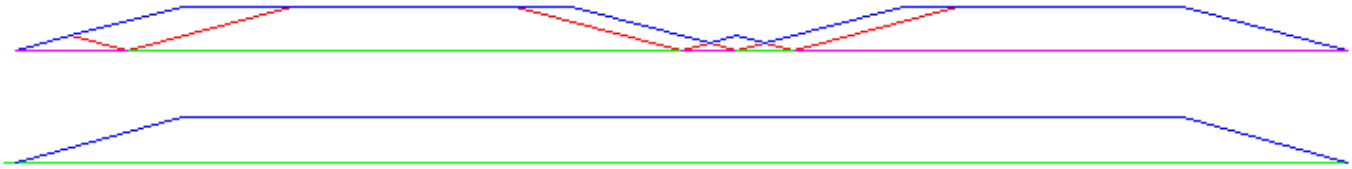


Figura 3.1: Naive Cam Detector

3.3.1.4. Planificacion de movimientos

Asegúrese de que los movimientos sean "lo suficientemente largos" para adaptarse a su máquina/material. Principalmente debido a la regla de que la máquina nunca se moverá a una velocidad tal que no pueda detenerse por completo al final del movimiento actual, hay una longitud de movimiento mínima que permitirá a la máquina mantener una velocidad de avance solicitada para un ajuste de aceleración dado.

Las fases de aceleración y desaceleración utilizan cada una la mitad de MAX_ACCELERATION del archivo ini. En una combinación que sea una inversión exacta, esto causara que la aceleración total del eje iguale el valor MAX_ACCELERATION del archivo ini. En otros casos, la aceleración real de la máquina es algo menor que esta aceleración ini.

Para mantener la velocidad de alimentación, el movimiento debe ser más largo que la distancia necesaria para acelerar de cero a la velocidad de avance deseada y luego detenerse de nuevo. Usando A como **1/2** del valor de MAX_ACCELERATION del archivo ini, y **F** como velocidad de avance **en unidades por segundo**, el tiempo de aceleración es $\mathbf{t_a = F/A}$ y la distancia de aceleración es $\mathbf{d_a = F * t_a/2}$. En desaceleración, el tiempo y la distancia es la misma, haciendo que la distancia crítica sea $\mathbf{d = d_a + d_d = 2 * d_a = F^2/A}$.

Por ejemplo, para una velocidad de avance de 1 pulgada por segundo y una aceleración de **10 pulgadas/seg^2**, la distancia crítica es $\mathbf{1^2/10 = 1/10 = 0.1}$ **pulgadas**.

Para una velocidad de avance de 0.5 pulgadas por segundo, la distancia crítica es $\mathbf{5^2/100 = 25/100 = 0.025}$ pulgadas.

3.3.2. Código G

3.3.2.1. Valores predeterminados

Cuando LinuxCNC se arranca, se cargan muchos códigos G y M de manera predeterminada. Los códigos actuales activos G y M se pueden ver en la pestaña MDI de la ventana *Códigos G activos:* en la interfaz AXIS. Estos códigos G y M define el comportamiento de LinuxCNC y es importante que entienda qué hace cada uno de ellos antes de ejecutar LinuxCNC. Los valores predeterminados se pueden cambiar cuando se ejecuta un archivo G-Code y pueden quedar en un estado diferente al del comienzo de sesión de LinuxCNC. La mejor práctica es establecer los valores predeterminados necesarios para el trabajo en un preámbulo de su archivo G-Code, sin suponer que los valores predeterminados no han cambiado. Imprimir la página de Referencia Rápida de códigos G puede ayudarle a recordar que es cada uno.

3.3.2.2. Velocidad de Alimentacion

La forma en la que se aplica la velocidad de alimentación depende de si un eje involucrado con el movimiento es un eje giratorio. Lea y comprenda la sección Velocidad de Avance si tiene un eje giratorio o un torno.

3.3.2.3. Offset del radio de la herramienta

El offset del radio de la herramienta (G41/G42) requiere que la herramienta sea capaz de tocar en algún lugar a lo largo de cada movimiento programado sin estropear los dos movimientos adyacentes. Si eso no es posible con el diámetro actual de la herramienta, se obtendrá un error. Una herramienta de menor diámetro puede trabajar sin error en el mismo camino. Esto significa que puede programar una herramienta para pasar por un camino que es más estrecho que la herramienta sin ningún error aparente. Para más información, ver la Sección Compensación de la Herramienta.

3.3.3. Homing

Después de iniciar LinuxCNC, cada eje debe tener definida una posición home antes de ejecutar un programa o un comando MDI.

Si su máquina no tiene interruptores home, una marca de coincidencia en cada eje puede ayudar a colocar las coordenadas de la máquina en el mismo lugar.

Una vez que los ejes están en home, se usarán los límites software establecidos en el archivo ini.

Si desea desviarse del comportamiento predeterminado, o desea usar el interfaz Mini, necesitará configurar la opción NO_FORCE_HOM = 1 en la sección [TRAJ] de su archivo ini. Se puede obtener más información sobre homing en Velocidad de Avance.

3.3.4. Cambios de herramientas

Hay varias opciones al hacer cambios de herramientas manuales. Ver la Sección [EMCIO] para obtener información sobre la configuración de estas opciones. Ver también las secciones G28 y G30 del capítulo de Código G.

3.3.5. Sistemas de coordenadas

Los sistemas de coordenadas pueden ser confusos al principio. Antes de manejar una máquina CNC debe comprender los conceptos básicos de los sistemas de coordenadas utilizados por LinuxCNC. La información detallada sobre los sistemas de coordenadas está en la sección Sistema de Coordenadas de este manual.

3.3.5.1. Coordenadas Maquina G53

Al hacer una secuencia homing, LinuxCNC configura el sistema de coordenadas de la máquina, G53, a cero para cada uno de los eje en la secuencia.

- Ningún otro sistema de coordenadas u offsets de herramientas se cambian por la secuencia homing.

La única forma de moverse en el sistema de coordenadas de la máquina es cuando se programa un G53 en la misma línea que el movimiento. Normalmente se está en el sistema de coordenadas G54.

3.3.5.2. Coordenadas de usuario - G54 a G59.3

Normalmente se usa el sistema de coordenadas G54. Cuando se aplica un offset al actual sistema de coordenadas de usuario, una pequeña bola azul con líneas estará en el origen de maquina cuando su DRO esté mostrando *Posición: Relativa Actual* en Axis. Si sus offsets son temporales, use Zero Coordinate System del menú o programe la máquina con *G10 L2 P1 X0 Y0 Z0* al final de su archivo de código G. Cambie el número *P* para que se adapte al sistema de coordenadas en el que desea borrar el offset.

- Los offsets almacenados en un sistema de coordenadas de usuario se conservan cuando LinuxCNC se apaga.
- Usando el botón *Touch Off* en Axis, se establece un offset para el Sistema de coordenadas elegido de usuario.

3.3.5.3. Cuando este perdido

Si tiene problemas para obtener 0,0,0 en el DRO cuando piense que debería mostrarse, puede que tenga algunos offsets programados y necesita eliminarlos.

- Muévase al origen de la máquina con G53 G0 X0 Y0 Z0
- Borre cualquier offset de G92 con G92.1
- Use el sistema de coordenadas G54
- Establezca el sistema de coordenadas G54 para que sea el mismo que el sistema de coordenadas máquina con G10 L2 P1 X0 Y0 Z0 R0
- Desactivar los offsets de herramienta con G49
- Activar la pantalla de coordenadas relativas desde el menú

Ahora debe estar en el origen de la máquina X0 Y0 Z0 y el sistema de coordenadas relativo debe ser el mismo que el sistema de coordenadas de máquina.

3.3.6. Configuraciones de la maquina

El siguiente diagrama muestra una fresadora típica que muestra la dirección del recorrido de la herramienta, la mesa y los interruptores de límite. Observe cómo la mesa se mueve en la dirección opuesta a las flechas del sistema de coordenadas cartesianas mostrado por la imagen *Dirección Herramienta*. Esto hace que la herramienta *se mueva* en el dirección correcta en relación con el material.

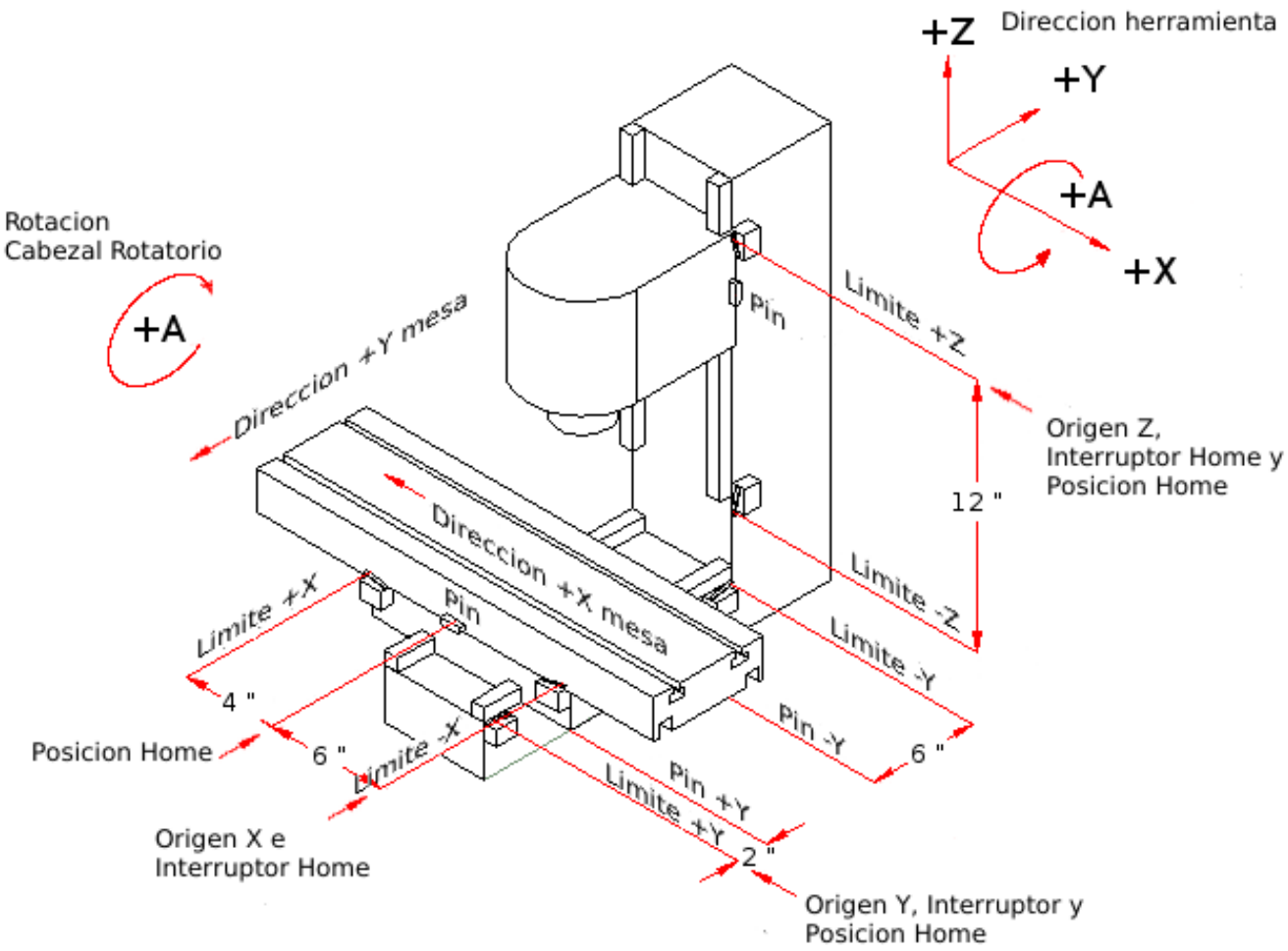


Figura 3.2: Configuración de fresadora

El siguiente diagrama muestra un torno típico que muestra la dirección del recorrido de la herramienta y los interruptores de límite.

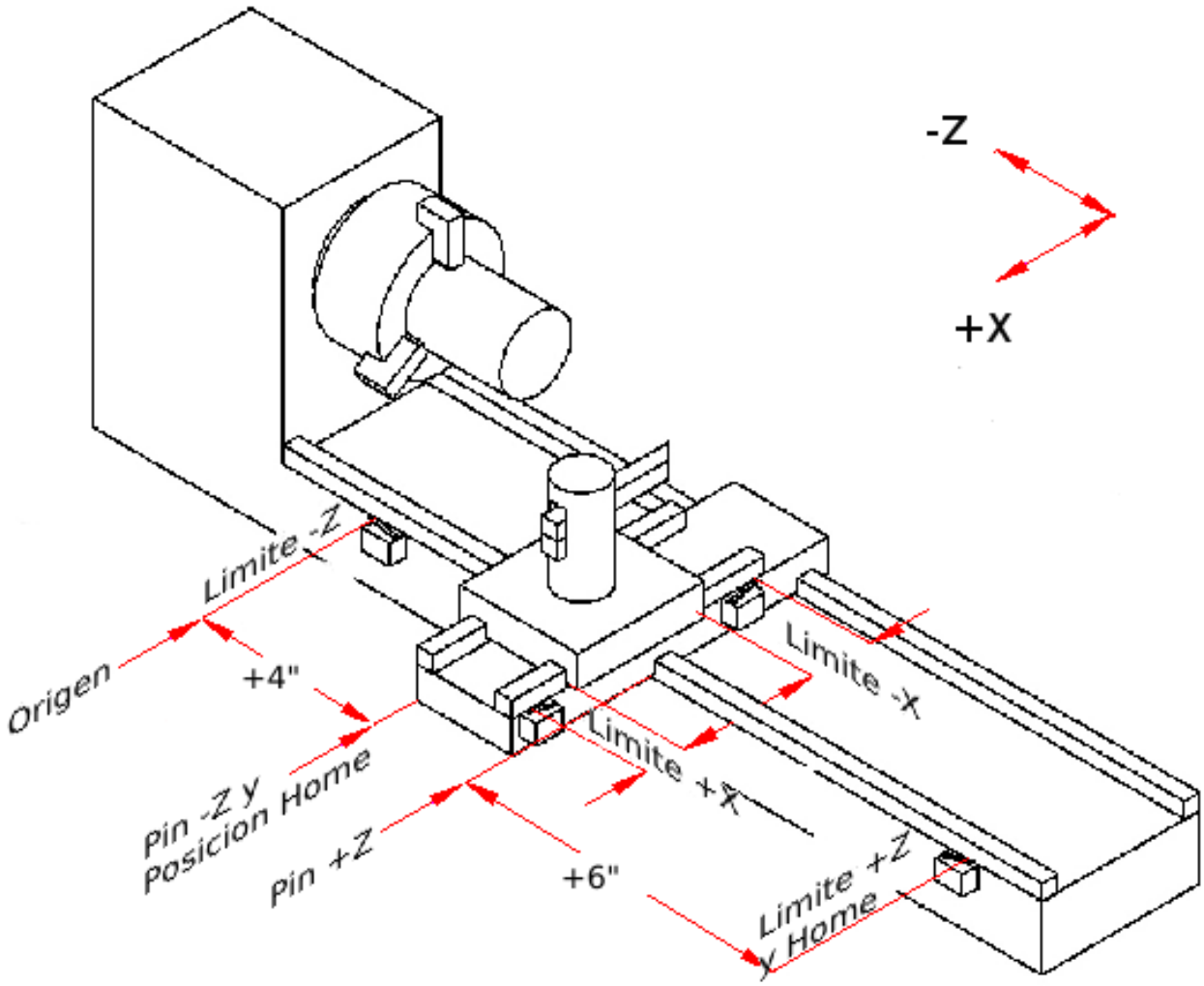


Figura 3.3: Configuración de Torno

3.4. Iniciando LinuxCNC

3.4.1. Ejecutando LinuxCNC

LinuxCNC se inicia con el archivo script *linuxcnc*.

```
linuxcnc [opciones] [<ini-file>]
```

OPCIONES DE SCRIPT LINUXCNC

- *-v* = verbose - imprime información mientras funciona
- *-d* = hace eco de los comandos del script en la pantalla para detectar fallos

Si al script *linuxcnc* se le pasa un archivo .ini como argumento, lo lee y se inicia LinuxCNC. La sección [HAL] del archivo .ini especifica el orden de carga de archivos HAL, si se usa más de uno. Una vez que se cargan los archivos HAL=xxx.hal, se carga

la GUI y luego se carga el archivo POSTGUI=.xxx.hal. Si se han creado objetos PyVCP o GladeVCP con pines HAL, debe usar el archivo HAL postgui para hacer cualquier conexion a esos pines. Consulte la sección [HAL] de la configuración INI para más información.

3.4.1.1. Selector de configuracion

Si no se pasa ningún archivo .ini al script linuxcnc, se carga el selector de configuración para que pueda elegir y guardar una configuración de ejemplo. Una vez que una configuración de ejemplo se ha guardado, se puede modificar para adaptarse a su aplicación. Los archivos de configuración se guardan en el directorio linuxcnc/configs.

Selector de configuración de LinuxCNC

Figura 3.4: Selector de configuración

3.5. Descripcion general de una máquina CNC

Esta sección proporciona una breve descripción de una máquina CNC desde los puntos de vista de entrada y salida del intérprete.

3.5.1. Componentes mecanicos

Una máquina CNC tiene muchos componentes mecánicos que pueden ser controlados o que pueden afectar a la forma en que se ejerce el control. Esta sección describe el subconjunto de aquellos componentes que interactúan con el interprete. Los componentes mecánicos que no interactúan directamente con el intérprete, como los botones de jog, no se describen aquí, incluso si afectan al control.

3.5.1.1. Ejes

Cualquier máquina CNC tiene uno o más ejes. Diferentes tipos de máquinas CNC pueden tener diferentes combinaciones. Por ejemplo, una *fresadora de 4 ejes* puede tener ejes XYZA o XYZB. Un torno normalmente tiene los ejes XZ. Una máquina de corte de espuma puede tener ejes XYUV. En LinuxCNC, el caso de una máquina *pórtico* XYYZ, con dos motores para un eje, se maneja mejor con la cinemática en lugar de con un segundo eje lineal. [3]

Ejes lineales primarios ejesprimarios lineales primarios lineales Los ejes X, Y y Z producen movimiento lineal en tres direcciones mutuamente ortogonales.

Ejes lineales secundarios ejessecundarios lineales secundarios lineales Los ejes U, V y W producen movimiento lineal en tres direcciones ortogonales. Típicamente, X y U son paralelos, Y y V son paralelo, y Z y W son paralelos.

Ejes Rotacionales ejesrotacionales rotacionales Los ejes A, B y C producen movimiento angular (rotación). Normalmente, A gira alrededor de una línea paralela a X, B gira alrededor de una línea paralela a Y, y C gira alrededor de una línea paralela a Z.

3.5.1.2. Husillo

Una máquina CNC tiene generalmente un husillo que contiene una herramienta de corte, una sonda, o el material en el caso de un torno. El husillo puede ser controlado por el software CNC.

3.5.1.3. Refrigerante

Una máquina CNC puede tener componentes para proporcionar refrigerante por niebla y/o inundación. El refrigerante puede ser controlado por códigos G.

[3] si el el movimiento de los componentes mecánicos no es independiente, como en máquinas hexapod, el lenguaje RS274/NGC y las funciones de mecanizado canónicas seguirán siendo utilizables, siempre que los niveles inferiores de control sepan cómo controlar los mecanismos reales para producir el mismo movimiento relativo de herramienta y pieza de trabajo como el que se produciría por ejes independientes. Esto se llama *cinemática*.

3.5.1.4. Controles de velocidad y alimentacion

Una máquina CNC puede tener controles separados de velocidades rapida y de avance, que permite al operador especificar la velocidad de avance real, o la velocidad utilizada en el husillo, en porcentaje de la velocidad programada.

3.5.1.5. Interruptor de borrado de bloque

Una máquina CNC puede tener un interruptor de eliminación de bloques de codigo G marcados para ello. Vea la seccion Eliminar Bloques.

3.5.1.6. Interruptor de parada de programa opcional

Una máquina CNC puede tener un interruptor de parada opcional de programa. Ver la seccion parada opcional de programa.

3.5.2. Componentes de control y datos

3.5.2.1. Ejes lineales

Los ejes X, Y y Z forman un sistema de coordenadas estándar de mano derecha de ejes lineales ortogonales. Las posiciones de los tres mecanismos de movimiento lineal se expresan usando coordenadas en estos ejes.

Los ejes U, V y W también forman un sistema de coordenada a derechas estándar. X y U son paralelos, Y y V son paralelos, y Z y W son paralelo (cuando el giro en A, B y C es cero).

3.5.2.2. Ejes de rotacion

Los ejes de rotación se miden en grados como ejes lineales de revolucion en los que la dirección de la rotación positiva es en sentido antihorario cuando se ven desde el extremo positivo del eje X, Y o Z correspondiente. Por "eje lineal de revolucion", nos referimos a uno en el que la posición angular puede aumentar sin límite (va hacia más-infinito) a medida que el eje gira en sentido antihorario y decrece sin límite (va hacia menos-infinito) a medida que el eje gira en el sentido de las agujas del reloj. Se utilizan ejes lineales de revolucion independientemente de si hay o no un límite mecánico en la rotación.

El sentido horario o antihorario se toma desde el punto de vista de la pieza de trabajo. Si la pieza de trabajo está sujeta a un plato giratorio que gira en un eje de rotación, un giro en sentido antihorario desde el punto de vista de la pieza de trabajo se logra girando el plato giratorio en una dirección que (para la mayoría de las configuraciones de máquina comunes) se ve en el sentido de las agujas del reloj desde el punto de vista de alguien parado al lado de la máquina. [4]

3.5.2.3. Punto controlado

El punto controlado es el punto cuya posición y velocidad de movimiento están controlados. Cuando el offset de la longitud de la herramienta es cero (el valor predeterminado), este es un punto en el eje del husillo (a menudo llamado punto calibrado) que es una distancia fija más allá del final del husillo, generalmente cerca del extremo del portaherramientas que encaja en el husillo. La ubicación del punto controlado se puede mover a lo largo del eje del husillo especificando una cantidad positiva para el offset de la longitud de la herramienta. Esta cantidad es normalmente la longitud de la herramienta de corte en uso, por lo que el punto controlado está al final de la herramienta de corte. En un torno, los offsets de longitud de herramienta se pueden especificar para los ejes X y Z, y el punto controlado esta en la punta de la herramienta o ligeramente fuera de ella (donde las líneas perpendiculares, alineadas a los ejes, tocadas por el *frente* y *flanco* de la herramienta se cruzan).

[4] si se viola el requisito de paralelismo, el creador del sistema tiene que decir cómo distinguir entre sentido horario y antihorario.

3.5.2.4. Movimiento lineal coordinado

Para manejar una herramienta a lo largo de una ruta especifica, un centro de mecanizado debe coordinar el movimiento de varios ejes. Usamos el término *movimiento lineal coordinado* para describir la situación en la que, nominalmente, cada eje se mueve a velocidad constante y todos los ejes se mueven desde sus posiciones iniciales a sus posiciones finales al mismo tiempo. Si solo los ejes X, Y y Z (o uno o dos de ellos) se mueven, se produce movimiento en una línea recta, de ahí la palabra *lineal* en el término. En movimientos reales, a menudo no es posible mantener la velocidad constante por la aceleración o desaceleración al comienzo y/o al final del movimiento. Sin embargo, es factible controlar los ejes para que, en todo momento, cada eje haya completado la misma fracción del movimiento requerido que los otros ejes. Esto mueve la herramienta a lo largo de la misma ruta, y también llamamos a este tipo de movimiento *movimiento lineal coordinado*.

El movimiento lineal coordinado se puede realizar a la velocidad de avance, o a la velocidad rapida, o puede estar sincronizado con la rotación del husillo. Si los límites físicos en la velocidad del eje hacen que la tasa deseada sea inalcanzable, todos los ejes se ralentizan para mantener el camino deseado.

3.5.2.5. Velocidad de avance

La velocidad a la que se mueve el punto controlado es, nominalmente, la velocidad estable que puede establecer el usuario. En el intérprete, la tasa de alimentación se interpreta de la siguiente manera (a menos que los modos *alimentacion inversa al tiempo* o *alimentacion por revolución* se esten utilizando, en cuyo caso, consulte la Sección G93-G94-G95-Mode).

1. Si X,Y o Z se mueven, F está en unidades por minuto en el sistema cartesiano XYZ, y todos los demás ejes (ABCUVW) se mueven para arrancar y parar de manera coordinada.
2. De lo contrario, si U,V o W se mueven, F está en unidades por minuto en el sistema cartesiano UVW y todos los demás ejes (ABC) se mueven para arrancar y parar de manera coordinada.
3. De lo contrario, el movimiento es puro movimiento giratorio y la palabra F está en unidades de rotación en el sistema ABC *pseudo-cartesiano*.

3.5.2.6. Refrigerante

El refrigerante de inundación y el refrigerante de niebla pueden encenderse independientemente. El lenguaje RS274/NGC los apaga juntos con un solo codigo M. Ver Sección M7 M8 M9.

3.5.2.7. Dwell

Se puede ordenar que un centro de mecanizado haga dwell (es decir, mantenga todos los ejes inmóviles) durante una cantidad específica de tiempo. El uso más común de dwell es romper y despejar las virutas, por lo que el husillo suele girar durante un dwell. Independientemente del modo de control de ruta (ver la sección control de ruta) la máquina se detendrá exactamente al final del movimiento programado anterior, como si estuviera en modo de ruta exacta.

3.5.2.8. Unidades

Las unidades utilizadas para distancias a lo largo de los ejes X, Y y Z pueden medirse en milímetros o pulgadas. Las unidades para todas las demás cantidades involucradas en el control de la máquina no puede ser cambiadas. Diferentes cantidades usan diferentes unidades específicas. La velocidad del husillo se mide en revoluciones por minuto. Las posiciones de los ejes de rotación se miden en grados. La velocidad de alimentación se expresan en unidades de longitud actual por minuto, o grados por minuto, o unidades de longitud por revolución del husillo, como se describe en la Sección G93 G94 G95.

3.5.2.9. Posicion actual

El lugar donde en cualquier momento se encuentra el punto controlado se llama *posición actual*, y el controlador siempre conoce dónde está ese punto. Los números que representan la posición actual deben ajustarse si, en ausencia de cualquier movimiento del eje, ocurre alguno de estos eventos:

1. Se cambian las unidades de longitud.
2. El offset de la longitud de la herramienta ha cambiado.
3. Se modifican los offsets del sistema de coordenadas.

3.5.2.10. Plano seleccionado

Siempre hay un "plano seleccionado", que debe ser el plano XY, el YZ, o el XZ del centro de mecanizado. El eje Z es, por supuesto, perpendicular al plano XY, el eje X al plano YZ, y el eje Y al plano XZ.

3.5.2.11. Carrusel de herramientas

Se asigna cero o una herramienta a cada ranura en el carrusel de herramientas.

3.5.2.12. Cambio de herramienta

Se puede ordenar a un centro de mecanizado que cambie las herramientas.

3.5.2.13. Pallet Shuttle

Hasta dos palets pueden intercambiarse por comando.

3.5.2.14. Modo de control de ruta

El centro de mecanizado puede colocarse en cualquier modo de control de ruta entre estos tres; (1) modo de parada exacta, (2) modo de ruta exacta, o (3) modo continuo con tolerancia opcional. En el modo de parada exacta, la máquina se detiene brevemente al final de cada movimiento programado. En modo de ruta exacta, la máquina sigue la ruta programada lo más exactamente posible, ralentizandose o deteniendose, si es necesario en las esquinas agudas del camino. En modo continuo, las esquinas de la ruta pueden ser redondeadas ligeramente para que la velocidad de alimentación pueda mantenerse actualizada (pero no más que la tolerancia, si se ha especificado). Ver las secciones Sección 5.2.34,G61/G61.1>> y G64.

3.5.3. Interaccion del interprete con los interruptores

El intérprete interactúa con varios conmutadores. Esta sección describe las interacciones con más detalle. En ningún caso el intérprete sabe cuál es la configuración de cualquiera de estos interruptores.

3.5.3.1. Interruptores de porcentaje de alimentacion y velocidad

Los comandos RS274/NGC *M48* y *M49* del intérprete permiten o deshabilitan los controles de porcentaje de alimentación y velocidad. Para ciertos movimientos, como la salida al final de un hilo durante un ciclo de roscado, los interruptores se deshabilitan automáticamente.

LinuxCNC reacciona a la configuración de porcentaje de alimentación y velocidad cuando estos interruptores están habilitados.

Consulte la sección Interruptores M48-M49 para obtener más información.

3.5.3.2. Interruptor de borrado de bloque

Si el interruptor de borrado de bloque está activado, las líneas de código G que comienzan con una barra inclinada (el carácter de *borrar bloque*) no se interpretan. Si el interruptor está apagado, tales líneas si son interpretadas. Normalmente, este interruptor debe activarse antes de iniciar el programa NGC.

3.5.3.3. Interruptor de parada opcional de programa

Si este interruptor está activado y se encuentra un código M1, la ejecución del programa entra en pausa.

3.5.4. Tabla de herramientas

Para usar el intérprete se requiere una tabla de herramientas. El archivo dice que herramientas están en qué ranuras de un cambiador de herramientas y cuál es el tamaño y tipo de cada herramienta. El nombre de la tabla de herramientas se define en el archivo ini:

```
[EMCIO]

# archivo de tabla de herramientas
TOOL_TABLE = tooltable.tbl
```

El nombre de archivo predeterminado probablemente se parezca a lo anterior, pero es posible que prefiera darle a su máquina su propia tabla de herramientas, utilizando el mismo nombre en su archivo ini, pero siempre con extensión tbl. Por ejemplo:

```
TOOL_TABLE = acme_300.tbl
```

o

```
TOOL_TABLE = EMC-AXIS-SIM.tbl
```

Para obtener más información sobre los detalles del formato de la tabla de herramientas, vea la sección Formato de la tabla de herramientas.

3.5.5. Parametros

Bajo el punto de vista del lenguaje RS274/NGC, un centro de mecanizado mantiene una matriz de parámetros numéricos definida por el valor de sistema (RS274NGC_MAX_PARAMETERS). Muchos de ellos tienen usos específicos, especialmente en la definición de sistemas de coordenadas. La cantidad de parámetros numéricos puede aumentar a medida que el desarrollo agrege soporte para nuevos parámetros. La matriz persiste con el tiempo, incluso si el centro de mecanizado está apagado. LinuxCNC usa un archivo de parámetros para asegurar la persistencia y le da al intérprete la responsabilidad de mantener el archivo. El intérprete lee el archivo cuando se inicia y lo escribe cuando se cierra.

Todos los parámetros están disponibles para su uso en programas de código G.

El formato de un archivo de parámetros se muestra en la siguiente tabla. El archivo consiste en cualquier cantidad de líneas de encabezado, seguidas por una línea en blanco, seguidas por cualquier cantidad de líneas de datos. El intérprete omite las líneas de encabezado. Es importante que haya exactamente una línea en blanco (sin espacios ni tabuladores) antes de los datos. La línea de encabezado que se muestra en la siguiente tabla describe las columnas de datos, por lo que se sugiere (pero no es obligatorio) que esa línea siempre se incluya en el encabezamiento.

El intérprete solo lee las dos primeras columnas de la tabla. La tercera columna, "Comentario", no es leída por el intérprete.

Cada línea del archivo contiene el número de índice del parámetro en la primera columna y, en la segunda columna, el valor al que ese parámetro debe establecerse. El valor se representa como un numero flotante de doble precisión dentro del intérprete, pero el punto decimal no es obligatorio en el archivo. Todos los parámetros que se muestran en la siguiente tabla son parámetros requeridos y deben ser incluidos en cualquier archivo de parámetros, excepto cualquier parámetro que represente un valor de eje de rotación para un eje no utilizado, que puede omitirse. Se señalara un error si falta algún parámetro requerido. El archivo puede incluir cualquier otro parámetro, siempre que su número esté en el rango de 1 a 5400. Los números de los parámetros se deben organizar en orden ascendente; si no lo estan, se señalara un error. Cualquier parámetro incluido en el archivo leído por el intérprete se incluirá en el archivo que se escriba cuando se cierre. El archivo original se guarda como un archivo de copia de seguridad cuando se escribe el nuevo archivo. Los comentarios no se conservan cuando se escribe el archivo.

Formato de archivo de parámetros

Número de parámetro	Valor Parámetro	Comentario
5161	0.0	G28 Home X
5162	0.0	G28 Home Y

Vea la sección <<gcode:parameters,Parametros para más información.

3.6. Ejecutando LinuxCNC

3.6.1. Invocando LinuxCNC

Después de la instalación, LinuxCNC comienza como cualquier otro programa de Linux: ejecutelo desde una terminal mediante el comando *linuxcnc*, o selecciónelo en el menú Aplicaciones > CNC.

3.6.2. Selector de Configuración

Al iniciar LinuxCNC desde el menú CNC o desde la línea de comandos, sin especificar un archivo ini, se inicia el cuadro de diálogo Selector de Configuración.

El diálogo selector de configuración permite al usuario elegir una de las configuraciones existentes (Mis Configuraciones) o seleccionar una nueva (Configuraciones de ejemplo) para copiarla a su directorio de inicio. Las configuraciones copiadas aparecerá en Mis configuraciones en la próxima invocación del Selector de Configuración.

El selector de configuración ofrece una selección de configuraciones organizada asi:

- *Mis configuraciones* - Configuraciones de usuario ubicadas en ~/linuxcnc/configs
- *Configuraciones de ejemplo* - Estas configuraciones, cuando se seleccionan, se copian en ~/linuxcnc/configs. Una vez que copie una configuración, si usa el selector, selecciónela desde *Mis configuraciones*. Los nombres corresponden a los directorios bajo ../configs/
  - *sim* - Configuraciones que incluyen hardware simulado. Pueden ser usadas para probar o aprender cómo funciona LinuxCNC.
  - *by_interface* - Configuraciones organizadas por tipo de GUI.
  - *by_machine* - Configuraciones organizadas por tipo de máquina.
  - *apps* - aplicaciones que no requieren iniciar linuxcnc pero pueden ser útiles para pruebas o aplicaciones como PyVCP o <<cha:glade-vcp
  - *attic* - Configuraciones obsoletas o históricas.

Las configuraciones en sim suelen ser el punto de partida más útil para usuarios nuevos y están organizadas en torno a las siguientes guis soportadas:

- *axis* - Gui de teclado y mouse
- *gmoccapy* - Gui de Pantalla táctil
- *gscreen* - Gui de Pantalla táctil
- *pyvcp_demo* - Paneles de Control Virtuales Python
- *qtvcp_screens* - Guis diseñadas con Qt5 y Python
- *tklinuxcnc* - Gui de teclado y mouse (ya no se mantiene)
- *touchy* - Gui de Pantalla táctil

Un directorio de configuración de GUI puede contener subdirectorios con configuraciones que ilustran situaciones especiales o incrustación de otras aplicaciones.

Las configuraciones *by_interface* están organizadas en torno a interfaces hardware comunes compatibles como:

- general mechatronics
- mesa
- parport
- pico
- pluto
- servotogo
- vigilant
- vitalsystems

Estas configuraciones pueden requerir el uso de hardware relacionado como puntos de partida para un sistema.

Las configuraciones *by_machine* están organizadas alrededor de sistemas completos conocidos como:

- boss
- cooltool
- plasmac
- scortbot erIII
- sherline
- smithy
- tormach

Es posible que se requiera un sistema completo para usar estas configuraciones.

Los elementos de *aplicaciones* son típicamente utilidades que no requieren iniciar linuxcnc o demostraciones de aplicaciones que pueden usarse con linuxcnc:

- info: crea un archivo con información del sistema que puede ser útil para diagnóstico de problemas
- gladevcp - Ejemplo de aplicaciones de gladevcp.
- halrun - Inicia halrun en un terminal.
- latency: aplicaciones para investigar la latencia
  - latency-test: prueba estándar
  - latency-plot - grafico de barras
  - latency-histogram - histograma
- parport - Aplicaciones para probar el puerto paralelo.
- pyvcp - Ejemplo de aplicaciones pyvcp.
- xhc-hb04 - Aplicaciones para probar un MPG inalámbrico USB xhc-hb04

nota
En el directorio de aplicaciones, solo se ofrecen para copiar en el directorio de usuario las aplicaciones que se pueden modificar de forma útil por el usuario.

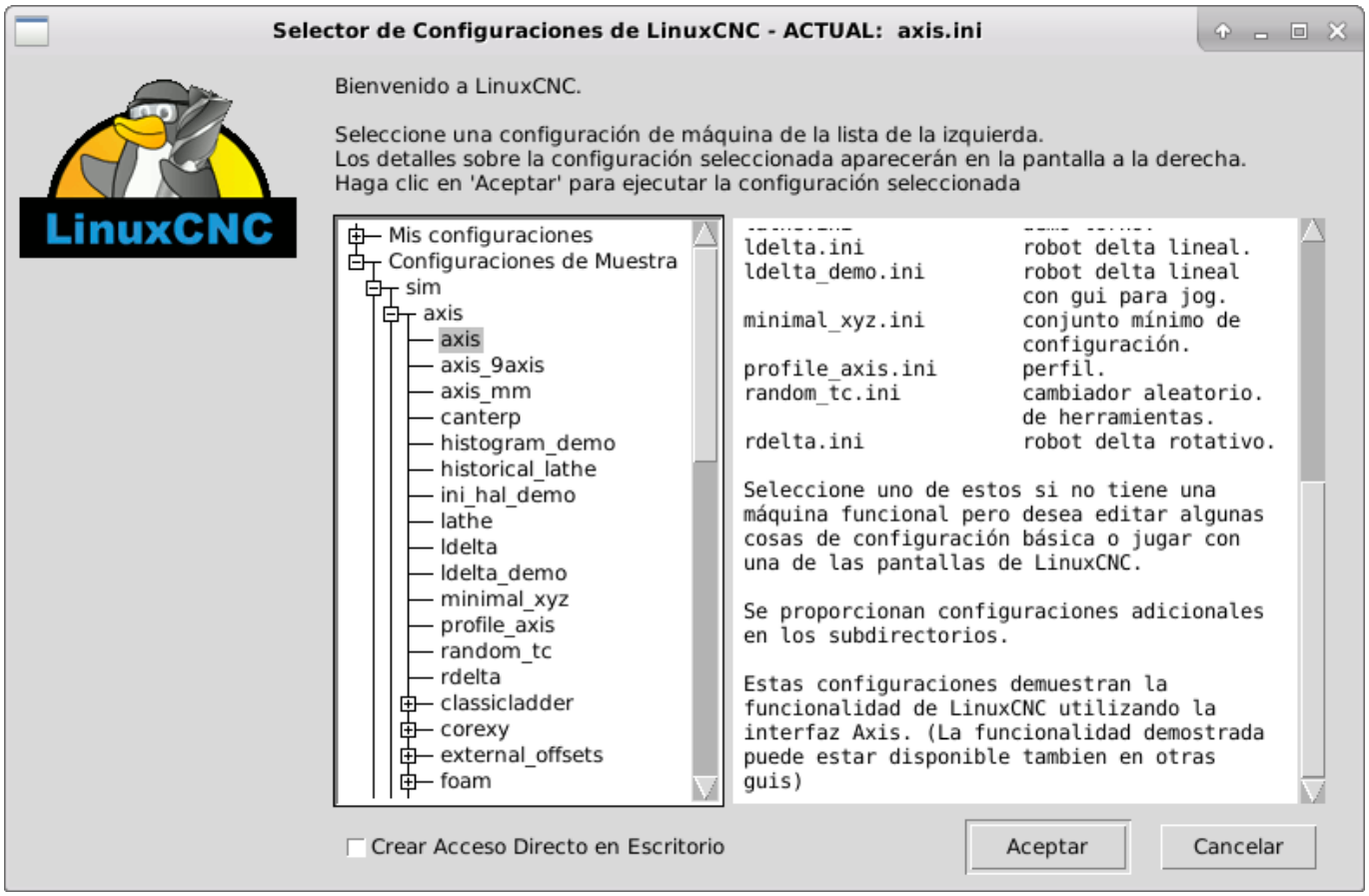


Figura 3.5: Selector de Configuración de LinuxCNC

Haga clic en cualquiera de las configuraciones enumeradas para mostrar información específica al respecto. Haga doble clic en una configuración o haga clic en Aceptar para comenzar la configuración.

Para agregar un icono en el escritorio que iniciara directamente una configuración sin mostrar la pantalla del Selector, seleccione *Crear Acceso Directo en Escritorio* y luego haga clic en *Aceptar*.

Cuando seleccione una configuración de la sección Configuraciones de Muestra, automáticamente se colocará una copia de esa configuración en el directorio ~/linuxcnc/configs.

3.6.3. Próximos pasos en la configuración

Después de encontrar la configuración de muestra que use el mismo interfaz de hardware que su máquina (o un simulador) y guardar una copia en su directorio de inicio, puede personalizarlo según los detalles de su máquina. Consulte el Manual del integrador para temas sobre la configuración.

3.6.4. Configuraciones del simulador

Todas las configuraciones enumeradas en Configuraciones de Muestra/sim están destinadas a ejecutarse en cualquier ordenador. No se requiere soporte de hardware específico y no es necesario tiempo real.

Estas configuraciones son útiles para estudiar capacidades u opciones individuales. Las configuraciones en sim están organizadas de acuerdo con la interfaz gráfica de usuario utilizada en la demostración. El directorio para Axis contiene la mayor cantidad de opciones y subdirectorios porque es la GUI más probada. Las capacidades demostradas con cualquier GUI específica pueden estar disponibles en otras GUIs también.

3.6.5. Recursos de configuración

El selector de configuración copia todos los archivos necesarios para una configuración a un nuevo subdirectorio de ~/linuxcnc/-configs. Cada directorio creado incluirá al menos un archivo ini (nombre_fichero.ini) que se usa para describir una configuración específica.

Los archivos de recursos dentro del directorio copiado incluyen típicamente uno o más archivos ini (nombre_fichero.ini) para configuraciones relacionadas y un archivo de tabla de herramientas (nombre_archivo_herramientas.tbl). Además, los recursos pueden incluir archivos hal (nombre_fichero.hal, nombre_fichero.tcl), un archivo README para describir el directorio, e información específica de configuración en un archivo de texto con nombre de una configuración específica (inifilename.txt). Estos dos últimos archivos se muestran cuando se utiliza el selector de configuración.

Las configuraciones de ejemplo suministradas pueden especificar archivos HAL en el archivo de configuración ini que no están presentes en el directorio copiado porque se encuentran en la biblioteca de sistema Hallib. Estos archivos se pueden copiar al directorio de configuración de usuario y ser alterados, según se requiera, por el usuario para modificaciones o pruebas. Puesto que el directorio de configuración del usuario es el primero donde se buscan archivos HAL, las modificaciones locales serán prevalentes.

El selector de configuración crea un enlace simbólico en el directorio de configuración de usuario (llamado hallib) que apunta a la biblioteca de sistema Halfile. Este enlace simplifica el copiado de un archivo de biblioteca. Por ejemplo, para copiar el archivo de biblioteca core_sim.hal para hacer modificaciones locales:

```
cd ~/linuxcnc/configs/nombre_de_configuracion
cp hallib/core_sim.hal core_sim.hal
```

3.7. Asistente de configuracion Stepconf

3.7.1. Introduccion

LinuxCNC es capaz de controlar una amplia variedad de tipos diferentes de maquinaria, utilizando diferentes interfaces de hardware.

Stepconf es un programa que genera archivos de configuracion para LinuxCNC para un tipo especifico de maquina CNC: aquellas que son controladas a traves de un *puerto paralelo estandar*, y utilizan las señales *Paso y Direccion*

Stepconf se instala automaticamente cuando se instala LinuxCNC; se encuentra en el menu CNC. El asistente genera un archivo en el directorio ~/linuxcnc/config en el cual guarda las opciones de cada configuracion que usted genere. Cuando se desea cambia algo, se necesita seleccionar el archivo cuyo nombre coincida con la configuracion que desea modificar. La extension del archivo es **.stepconf**.

El asistente Stepconf necesita una resolucion de pantalla de al menos 800 x 600 para que los botones de la parte baja de la pantalla sean visibles.

3.7.2. Pagina de entrada

Figura 3.6: Pagina inicial

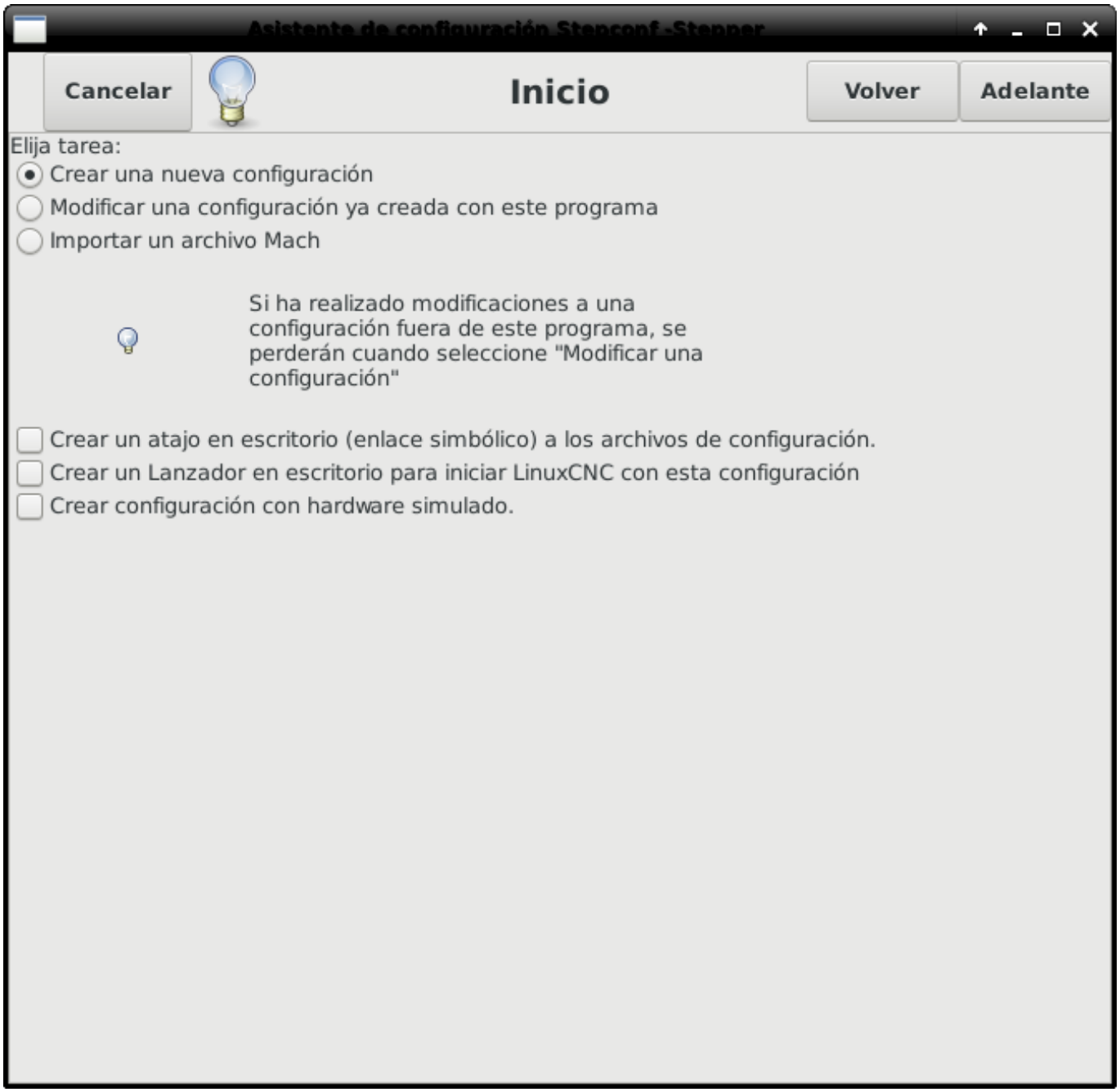


Figura 3.7: Pimera Pagina

Los tres primeros botones radio son autoexplicativos:

- *Crear una nueva configuración* - Crea una configuracion nueva.
- *Modificar una configuracion ya creada...* - Modifica una configuracion existente. Despues de seleccionar esta opcion, aparecera una pantalla de seleccion de archivo y usted debera seleccionar el archivo con extension .stepconf que desea modificar. Si usted relizo alguna modificacion previa a los archivos principales .hal o .ini, estas modificaciones se perderan. Las modificaciones en los archivos custom.hal y custom_postgui.hal no seran canbiadas por el asistente Stepconf. Stepconf resaltara la ultima configuracion que se construyo.
- *Importar un archivo Mach* - Despues de seleccionar esta opcion, aparecera una pantalla de seleccion de archivo. Seleccione un archivo de configuración de Mach; LinuxCNC intentará convertirlo en un archivo de configuración apropiado para si mismo.

Después de la importación, pase por las páginas de stepconf para confirmar/modificar las entradas. El archivo mach xml original no se cambiará.

Las siguientes opciones se registrarán en un archivo de preferencias para la próxima ejecución de Stepconf.

- *Crear Atajo en Escritorio* - Se generara en su escritorio un enlace a los archivos.
- *Crear un Lanzador en Escritorio* - Se generara un lanzador en su escritorio para iniciar la aplicacion.
- *Crear Hardware Simulado* - Permite crear configuraciones para pruebas, incluso sin tener el hardware apropiado.

3.7.3. Informacion Basica

Figura 3.8: Informacion Basica

- *Nombre de la maquina* - Seleccione un nombre para su maquina. Utilice solo letras mayusculas, minusculas, digitos, - y _.
- *Configuracion de Ejes* - Seleccione XYZ (Fresadora), XYZA (Fresadora de 4 ejes) o XZ (Torno).
- *Unidades Maquina* - Seleccione pulgadas o milimetros. Todas las preguntas posteriores (tales como el largo de los ejes, el paso de los tornillos, etc) deberan ser contestadas utilizando las unidades seleccionadas.
- *Tipo de Driver* - Si usted tiene uno de los controladores de motor a pasos listados en el menu desplegable, seleccionelo directamente. En cualquier otro caso, busque los 4 valores de tiempo necesarios. Utilize los manuales de su controlador y

rellene los campos. Si sus manuales le dan los datos en microsegundos multipliquelos por 1000 para nanosegundos. Por ejemplo, si el manual marca 4.5 us, escriba 4500 ns.

En la pagina wiki de LinuxCNC.org puede ser consultada una lista de controladores populares, asi como sus tiempos, en Temporizado de Drivers Stepper.

El acondicionamiento extra de señal o el aislamiento electrico, como cuando se usan optoacopladores y filtros RC en tarjetas breakboards, pueden requerir diferentes valores de tiempo a los indicados para su controlador. Puede ser el caso que se requiera agregar tiempo extra a los valores de temporizacion para compensar los filtros o aislamientos. La seccion de seleccion de configuracion tiene maquinas de la marca Sherline ya configuradas para su uso en caso de que posea una de estas.

- *Step Time* - Tiempo del pulso de paso "Encendido" en nanosegundos.
- *Step Space* - Tiempo minimo entre dos pulsos de paso en nanosegundos.
- *Direction Hold* - Tiempo que debe ser mantenido activo el pin de direccion despues de ordenar cambio de direccion, en nanosegundos.
- *Direction Setup* - Tiempo debe preceder a un cambio de direccion despues del ultimo pulso de paso en la direccion anterior.
- *Primer Parport* - Usualmente la direcion en hexadecimal del primer puerto paralelo es 0x378 (puerto no PCI).
- *Segundo Parport* - En caso de ser necesario especificar un puerto paralelo extra, introduca la direccion y el tipo. Para informacion de como encontrar la direccion de puertos paralelos PCI vea la seccion Port Address en el manual de integrador (trate con 0x278 o 0x3BC para puertos no PCI)
- *Maximo Jitter en Periodo Base* - Introduzca el resultado de la prueba de latencia. Para correr la prueba de latencia presione el boton "Test Jitter Periodo Base". Vea la seccion de la prueba de latencia para mas detalles.
- *Tasa Max de Pasos* - Stepconf calculara automaticamente la tasa maxima de pulsos de pasos basandose en las caracteristicas del controlador del motor y en el resultado de la prueba de latencia.
- *Periodo Base Mínimo* - Stepconf calculara automaticamente el periodo base minimo basandose en las caracteristicas del controlador del motor y el resultado de la prueba de latencia.

3.7.4. Prueba de latencia

Mientras se ejecute la prueba, usted debera *abusar* de la computadora. Mueva ventanas alrededor de la pantalla. Navegue en internet. Copie algunos archivos de gran tamano en diferentes partes del disco duro. Reproduzca musica. Corra algun programa OpenGl como glxgears. La idea es poner a la computadora en apuros mientras se ejecuta la prueba para poder tener una idea de cuales seran los peores casos de demanda a la computadora y sus tiempo de respuesta. Ejecute la prueba al menos unos cuantos minutos. Contra mas tiempo la ejecute, mas probable es que detecte casos especiales que solo suceden en intervalos poco frecuentes. Esta prueba es solo para la computadora; no se requiere que conecte los controladores de motores.

aviso
No ejecute LinuxCNC mientras realiza la prueba de latencia.

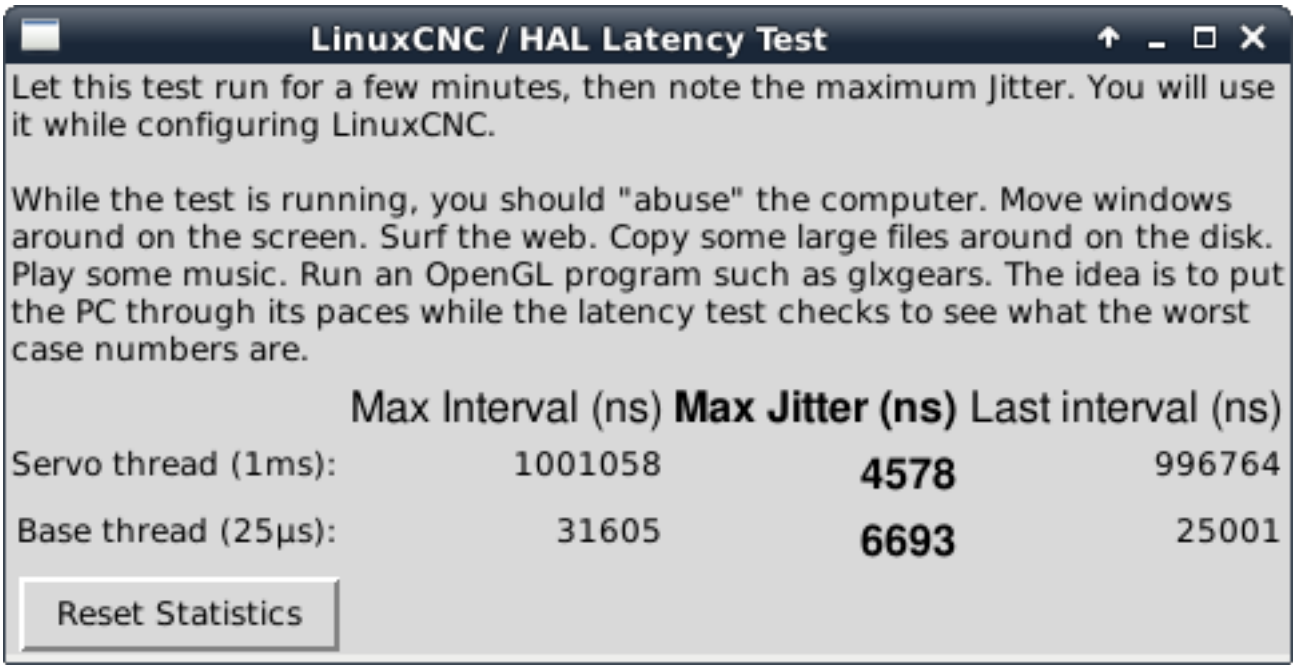


Figura 3.9: Prueba de Latencia

La latencia es el tiempo que le tomara a una PC concreta detener lo que esta haciendo y responder a una solicitud externa. En este caso, la solicitud es el *latido periodico* que sirve como referencia de tiempo para la genracion de los pulsos de paso. Cuanto menor sea la latencia, mas rapido se generaran los latidos, y mas rapidos y suaves seran los pulsos de paso.

La latencia es mucho mas importante que la velocidad de la CPU. La velocidad de la CPU no es el unico factor determinate en la latencia. La placa madre, tarjetas de video, puertos USB, problemas con SMI, y otras cosas pueden afectar la latencia.

Troubleshooting SMI Issues (LinuxCNC.org Wiki)

Encuentre aqui soluciones a algunos problemas de SMI y tiempo real comunes en Ubuntu

http://wiki.linuxcnc.org/cgi-bin/emcinfo.pl?FixingSMIIssues

El numero importante es el "Jitter Maximo". En el ejemplo, 9075 nanosegundos, o 9.075 microsegundos, es el maximo retraso. Guarde este numero, y escribalo en el recuadro Jitter Maximo del Periodo Base.

Si el maximo retraso se encuentra entre 15-20 microsegundos o menor (15000-20000 nanosegundos), la computadora deberia dar muy buenos resultados con la generacion de pulsos de pasos. Si la latencia maxima esta entre 30-50 microsegundos, se pueden seguir obteniendo buenos resultados, pero la tasa maxima de generacion de pulsos puede ser un poco decepcionante, especialmente si se usan micropasos o un tornillo con un paso muy fino. Si los numeros son 100us o mas (100 000 nanosegundos), la PC no es una buena candidata para la generacion de pulsos de paso por software. Numeros superiores a 1 milisegundo (1 000 000 nanosegundos) significan que la PC no es una buena candidata para ejecutar LinuxCNC, sin importar si se usa generacion de pulsos de paso por software o no.

3.7.5. Ajustes del puerto Paralelo

Figura 3.10: Pagina de ajuste del Puerto Paralelo

Para cada pin se debera seleccionar la señal de control que concuerde con la configuracion del puerto.

Active la casilla "invert" si la señal de control requiere ser invertida (0V para activo/Verdadero, 5v para inactivo/Falso)

- *Esquemas de pines predefinidos* - Se configuraran automaticamente los pines del 2 al 9 de acuerdo al estandar de las maquinas Sherline (Direccion en los pines 2, 4, 6, 8) o Xylotex (Direccion en los pines 3, 5, 7, 9).
- *Entradas y Salidas* - Si el pin no sera utilizado como entrada o salida, seleccionarlo como "Sin uso".

- *Señal de Paro Externo (E stop)* - Esta señal pude ser seleccionada en la casilla desplegable. Una cadena de señal de paro tipica utiliza solo contactos en serie normalmente cerrados.
- ’Posicion home y limites de seguridad (Homing & Limit Switches) - Estos pines pueden ser seleccionados para la mayoria de las configuraciones utilizando la casilla desplegable.
- *Bomba de Carga (Charge Pump)* - Si su controlador de motor requiere de una señal de bomba de carga, simplemente seleccione esta opcion de la lista desplegable y conecte la señal al pin seleccionado. La salida de la bomba de carga sera conectada a la tarea base por el programa Stepconf. La salida de bomba de carga sera aproximadamente 1/2 de la maxima tasa de generacion de pulsos de paso mostrados en la pagina de configuracion basica.

3.7.6. Configuracion del puerto paralelo 2

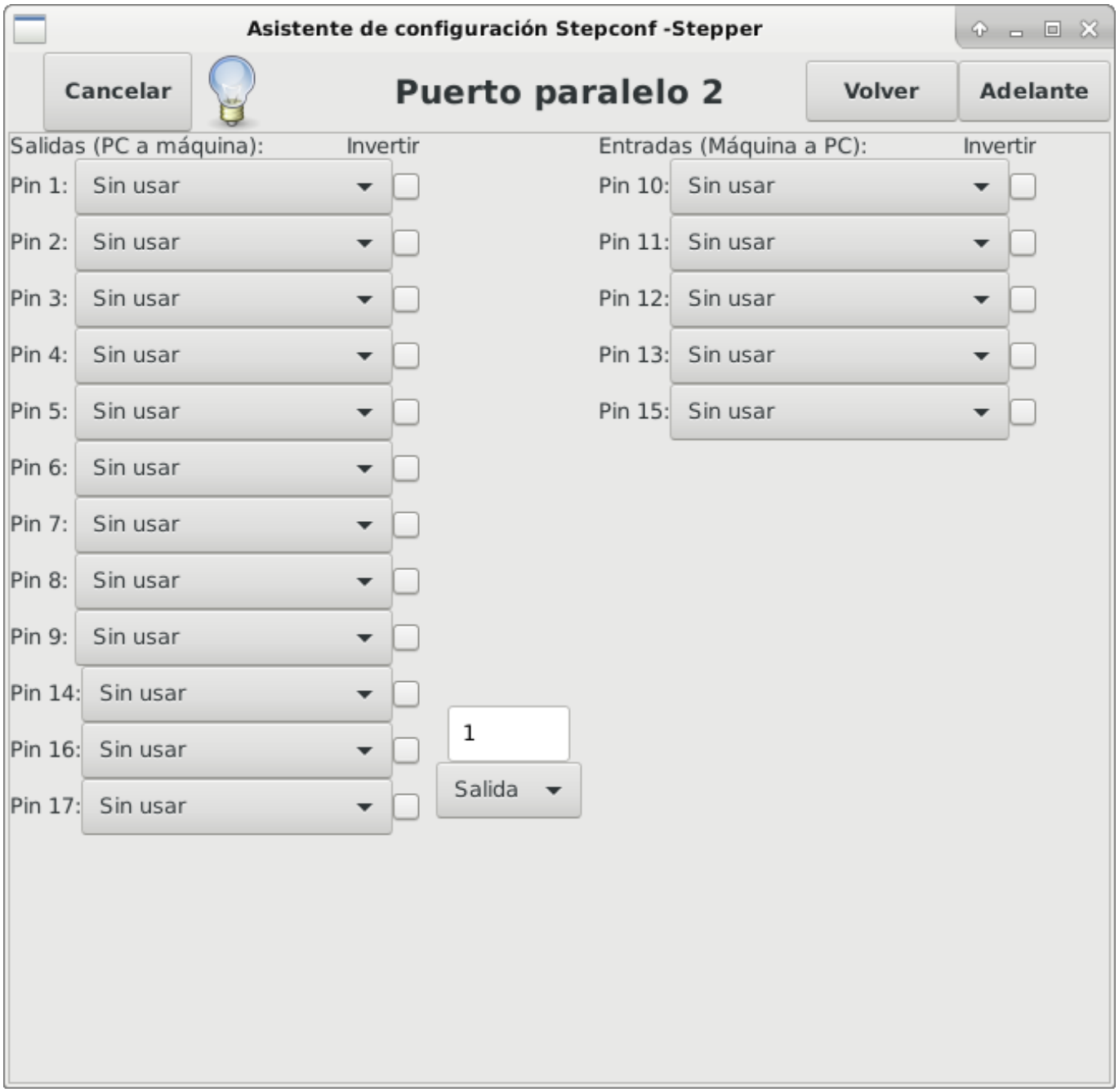


Figura 3.11: Página de configuración del puerto paralelo 2

El segundo puerto paralelo (si está seleccionado) puede configurarse y asignar sus pines en esta página.
No se pueden seleccionar señales de paso y dirección.
Puede seleccionarlo de entrada o de salida para maximizar el número de pines de entrada/salida que están disponibles.
Puede especificar la dirección como hexadecimal (a menudo 0x378) o como el número de puerto predeterminado de Linux (probablemente 1).

3.7.7. Opciones

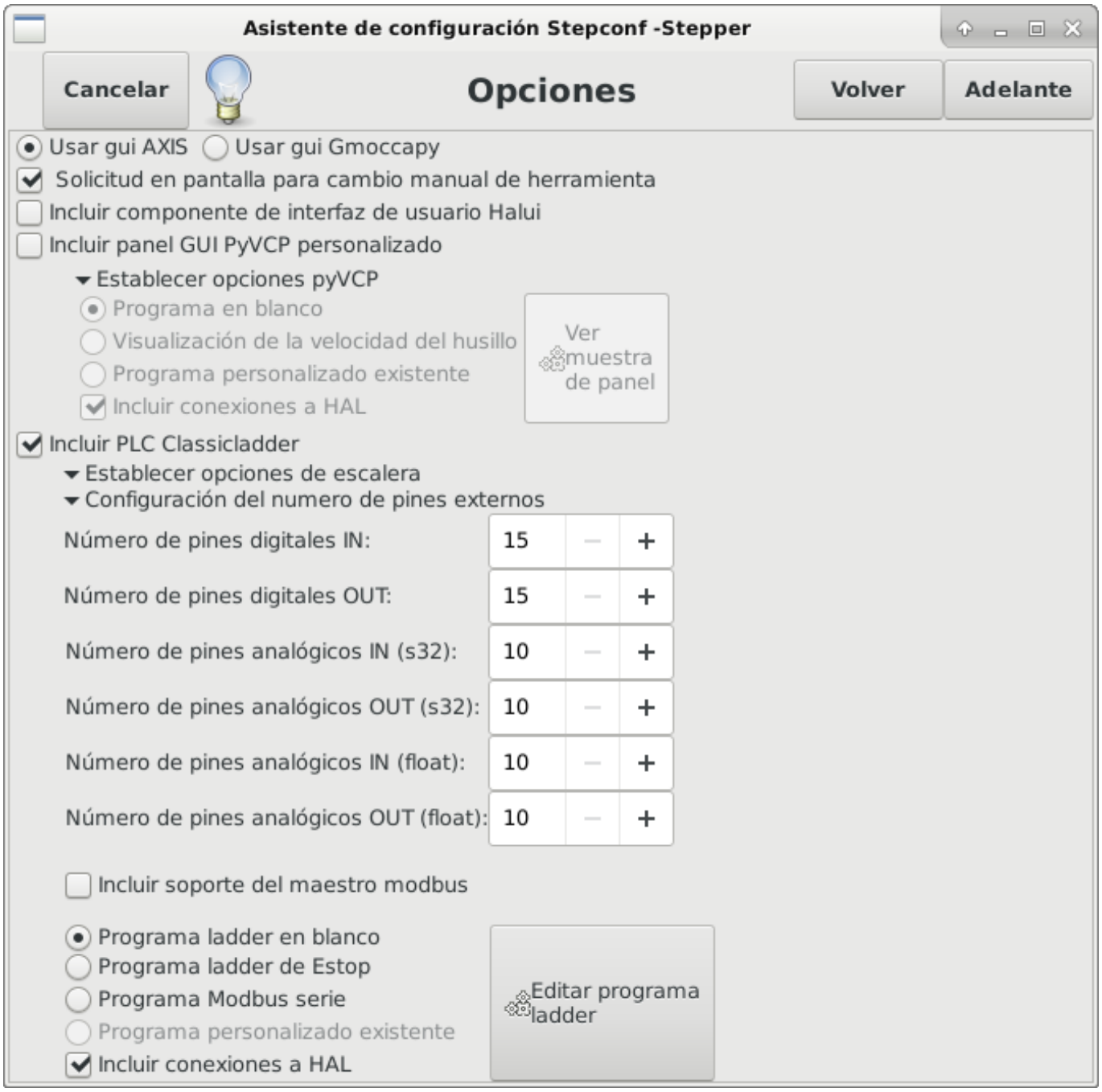


Figura 3.12: Configuracion avanzada

- *Incluir Halui*: esto agregará el componente de interfaz de usuario Halui. Ver el Capitulo HALUI para más información.
- *Incluir pyVCP*: esta opción agrega el archivo base del panel pyVCP o un archivo ejemplo para trabajar en el. Ver el Capítulo PyVCP para más información.
- *Incluir ClassicLadder PLC* - Esta opción agregará el PLC ClassicLadder (Controlador lógico programable). Ver el Capitulo Classicladder para más información.

- *Indicador en pantalla para cambio de herramienta* - Si esta casilla está marcada, LinuxCNC para y le pide que cambie la herramienta cuando se encuentre *M6*. Esta característica generalmente solo es útil si tiene herramientas predimensionadas.

3.7.8. Configuracion de los Ejes

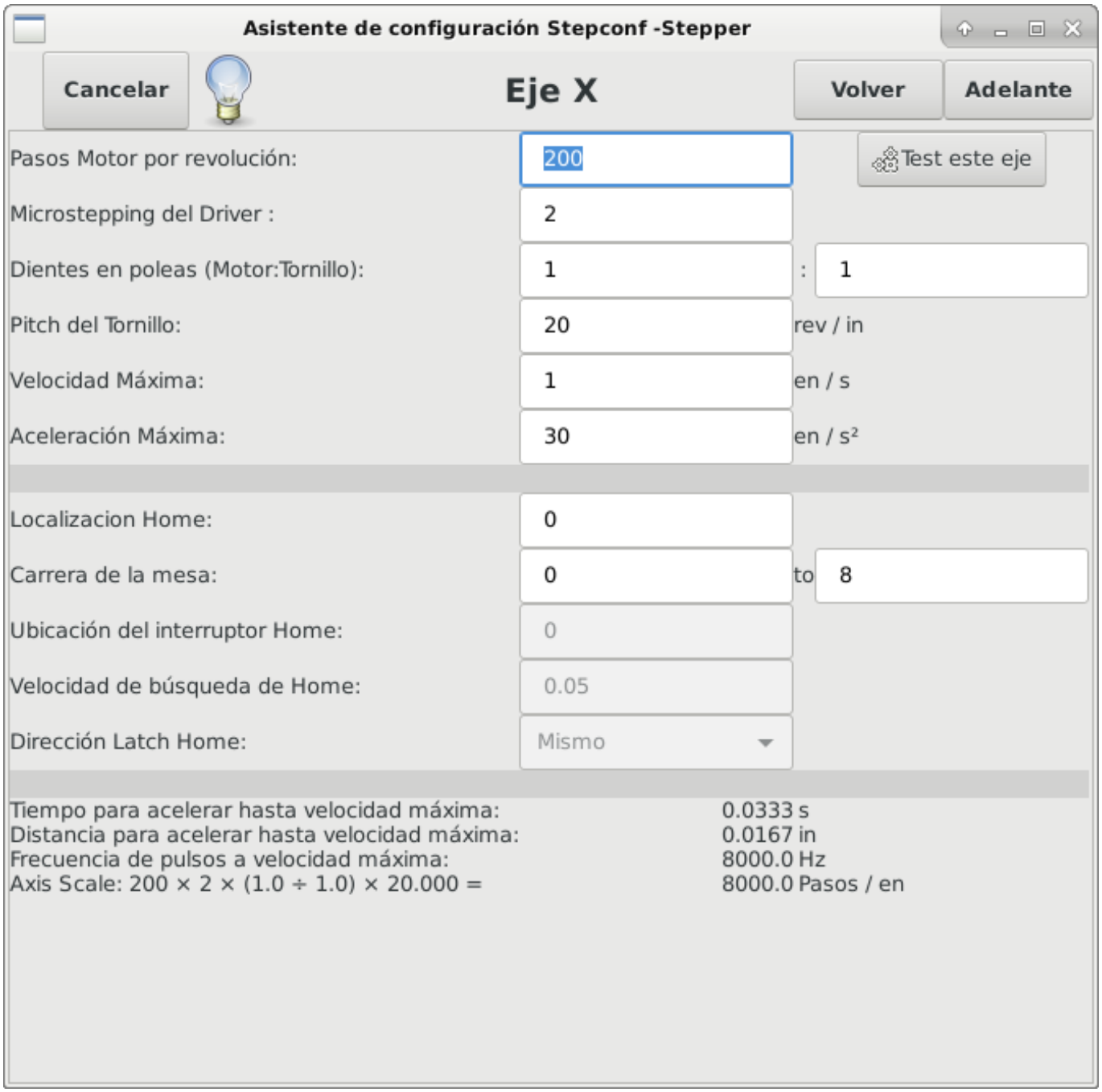


Figura 3.13: Pagina de configuracion de ejes

- *Pasos del motor por revolucion* (Motor Steps Per Revolution) - El numero de pasos completos por revolucion del motor. Si solo se tiene el dato de los grados por paso del motor (ejemplo 1.8 grados), se debe dividir 360 por el numero de grados por paso para encontrar el numero de pasos por revolucion.

- *Micro pasos* (Driver Microstepping) - El numero de micropasos producidos por el controlador por cada paso fisico completo del motor. Entre "2" para semipasos. Por ejemplo, si el controlador produce 1/10 de giro de un paso completo del motor por cada pulso de paso que recibe, escriba 10 en la casilla.
- *Relacion de Poleas* (Pulley Ratio) - Si su maquina tiene poleas o engranes entre el motor y el tornillo, escriba su relacion mecanica aqui. Si no tiene, escriba "1:1".
- *Paso del tornillo* (Leadscrew Pitch) - Entre aqui el paso del tornillo. Si se selecciono unidades "Inch", entre el numero de hilos por pulgada (por ejemplo, 8 para un tornillo de 8 TPI). Si se tiene un tornillo con varias entradas, se necesita saber cuantas vueltas se requieren para mover la tuerca una pulgada. Si se selecciono *mm* como unidades, entre el numero de milimetros que la tuerca se movera por revolucion (ejemplo, 2 para 2 mm/rev). Si la maquina se mueve en la direccion opuesta a la esperada, entre un valor negativo, o invierta el pin de direccion del eje.
- *Velocidad Maxima* (Maximum Velocity) - Entre la velocidad maxima del eje, en unidades por segundo.
- 'Aceleracion Maxima (Maximum Acceleration) - El valor correcto de esta casilla solo puede ser determinado por experimentacion. Vea <<sub:finding-maximum-velocity,Encontrar Velocidad Maxima para ajustar la velocidad, y <<sub:finding-maximum-acceleration,Encontrar Velocidad Maxima para ajustar la aceleracion.
- *Posicion Home* (Home Location) - Home es la posicion a la que la maquina se movera despues de completar el procedimiento de inicio del eje. Para maquinas sin interruptores de posicion home, esta es la posicion a la cual el operador debera mover la maquina antes de presionar el boton de inicializacion del eje (Home). Si se combinan interruptores home y de limite, se debera mover la maquina fuera del interruptor para inicializar el eje o se recibira un error de limite en el eje.
- *Carrera de la mesa* (Table Travel) - El rango de carrera que el codigo g no podra sobrepasar. La posicion de inicializacion del eje debe estar dentro del area de carrera. En particular, tener la posicion de inicializacion (Home) de un eje exactamente en un limite del area de carrera, producira una configuracion invalida.
- *Localizacion de los interruptores home* (Home Switch Location) - La posicion en la cual el interruptor home se activa o desactiva, relativa al origen maquina. Este apartado y los dos siguientes solo apareceran cuando se selecciona la existencia de interruptores home en la configuracion de los pines del puerto paralelo. Si se combinan los interruptores de limite y de home, la posicion del interruptor home no puede ser la misma que la posicion home o se producira un error de limite de articulacion.
- *Velocidad de busqueda de home* (Home Search Velocity) - Velocidad usada en la busqueda de los interruptores home. Si el interruptor se encuentra cercano al limite de carrera del eje, esta velocidad debe ser seleccionada de tal forma que el eje tenga suficiente tiempo para desacelerar hasta detenerse antes de llegar al limite fisico de la carrera. Si el interruptor se encuentra cerrado en un rango corto de carrera, (en lugar de estar cerrado desde el punto de disparo hasta un final de carrera), la velocidad debera ser seleccionada de tal forma que el eje pueda desacelerar hasta detenerse antes de que el interruptor se habra otra vez, y el procedimiento de homing debera comenzarse siempre desde el mismo lado del interruptor. Si la maquina se mueve en la direccion contraria al inicio del homing, cambie el signo del parametro **Home Search Velocity**.
- *Direccion de enclavamiento* (Home Latch Direction) - Seleccione "Igual" para que el interruptor sea liberado y posteriormente la maquina se acerque a el a muy baja velocidad. La segunda vez que el interruptor se cierre, definira la posicion home. Seleccione "Opuesto" para realizar la inializacion liberando lentamente el interruptor; cuando el interruptor se abra, se marcara la posiocion home.
- *Tiempo para acelerar hasta maxima velocidad* (Time to accelerate to max speed) - Tiempo calculado a partir de *Max Acceleration* y *Max Velocity*.
- *Distancia para acelerar hasta maxima velocidad* (Distance to accelerate to max speed) - Distancia para alcanzar maxima velocidad desde posicion de parado.
- *Tasa de pulsos a maxima velocidad* (Pulse rate at max speed) - Este dato se calcula en base a los valores anteriores. El valor maximo de la **Tasa** determina el *BASE_PERIOD*. Valores por encima de 20000Hz pueden producir tiempos de respuesta muy bajos o incluso bloqueos (La tasa maxima varia entre computadoras)
- *Escala del Eje* (Axis SCALE) - El numero que sera usado en el archivo ini en la seccion [SCALE]. Representa cuantos pasos se deben dar por unidad de usuario.
- *Probar este Eje* (Test this axis) - Esta opcion abre una ventana para permitir probar cada eje y puede ser utilizada despues de llenar toda la informacion referente a cada eje.

3.7.8.1. Probar este eje

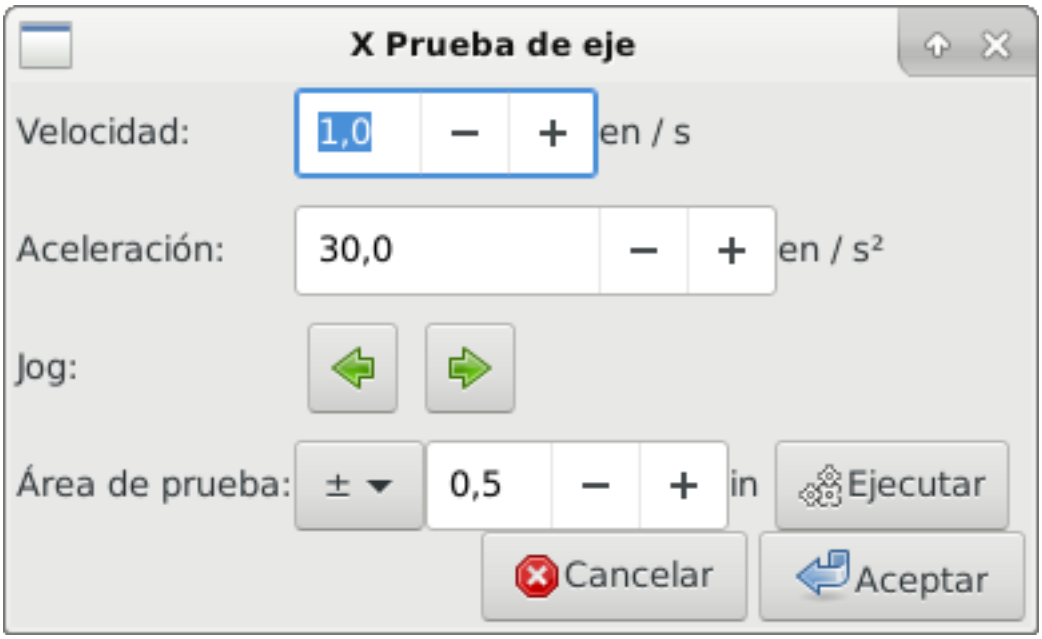


Figura 3.14: Probar este eje

Es un comprobador básico que solo emite señales de paso y dirección para probar diferentes valores de aceleración y velocidad.

importante
Para utilizar la prueba de este eje, debe habilitar manualmente el eje si se requiere. Si su controlador tiene una bomba de carga, tendrá que derivarla. Probar el eje no reacciona a las entradas del interruptor de límite. Usar con precaución.

Encontrar la velocidad máxima

Comience con una baja aceleración (por ejemplo, `2 pulgadas/s²` o `50 mm/s²`) y la velocidad que espera alcanzar. Usando los botones provistos, mueva el eje hasta cerca del centro de su carrera. Tenga cuidado, porque con un bajo valor de aceleración puede recorrerse una distancia sorprendentemente larga hasta que el eje desacelere completamente y pare.

Después de calcular la cantidad de carrera disponible, ingrese una distancia segura en el Área de prueba, teniendo en cuenta que después de un bloqueo, el motor puede comenzar a moverse en una dirección inesperada. Luego haga clic en Ejecutar. La máquina comenzará a avanzar y retroceder a lo largo de este eje. En esta prueba, es importante que la combinación de aceleración y área de prueba permita que la máquina alcance la velocidad seleccionada y la mantenga al menos una corta distancia: cuanto mayor sea la distancia, mejor será esta prueba. La fórmula `d=0.5 * v * v/a` da la distancia mínima requerida para alcanzar la velocidad especificada con la aceleración dada. Si es conveniente y seguro hacerlo, empuje la mesa contra la dirección del movimiento para simular las fuerzas de corte. Si la máquina se para, reduzca la velocidad y comience nuevamente la prueba.

Si la máquina no se paró, haga clic en el botón *Run* para parar. El eje vuelve ahora a la posición donde comenzó. Si la posición es incorrecta, el eje se estancó o perdió pasos durante la prueba. Reduzca la velocidad y comiencela otra vez.

Si la máquina no se mueve, se detiene o pierde pasos, incluso a baja velocidad, verifique lo siguiente:

- Corregir los tiempos de onda de paso
- Pinout correcto, incluyendo *Invert* en los pines de paso
- Cableado correcto y bien protegido

- Problemas físicos con el motor, acoplamiento del motor, husillo, etc.

Una vez que haya encontrado una velocidad a la que el eje no se detiene o pierde pasos durante este procedimiento de prueba, reducirlo en un 10% y usarlo como *Velocidad máxima* del eje.

Encontrar la máxima aceleración Con la velocidad máxima que encontro en el paso anterior, ingrese el valor de aceleración a probar. Usando el mismo procedimiento anterior, ajuste el valor de Aceleración hacia arriba o hacia abajo según sea necesario. En esta prueba, es importante que la combinación de aceleración y área de prueba permitan que la máquina alcance la velocidad seleccionada. Una vez que haya encontrado un valor en el que el eje no se detiene ni pierde pasos durante este procedimiento de prueba, reducirlo en un 10% y usarlo como Aceleración máxima del eje.

3.7.9. Configuracion del husillo

Figura 3.15: Página de configuración del husillo

Esta página solo aparece cuando se selecciona *Spindle PWM* en la página *Pin Portout Parallel* para una de las salidas.

3.7.9.1. Control de velocidad del eje

Si *Spindle PWM* aparece en el pinout, debe aportarse la siguiente información:

- *PWM Rate* - La *frecuencia portadora* de la señal PWM al husillo. Entrar *0* para el modo PDM, que es útil para generar un voltaje de control analógico. Consulte la documentación de su controlador de husillo para conocer el valor apropiado.
- *Speed 1 y 2, PWM 1 y 2*: el archivo de configuración generado utiliza una relación lineal simple para determinar el valor PWM para un valor RPM dado. Si los valores no se conocen, se pueden determinar. Para más información, ver determinación de la calibración del husillo.

3.7.9.2. Movimiento sincronizado con el husillo

Cuando las señales apropiadas de un encoder de husillo están conectadas a LinuxCNC a través de HAL, LinuxCNC admite el roscado en torno. Estas señales son:

- *Índice del husillo* - Es un pulso que ocurre *una vez por revolución* del husillo.
- *Fase A del husillo* - Este es un pulso que ocurre en múltiples ubicaciones, igualmente espaciadas, a medida que gira el husillo.
- *Fase B del husillo (opcional)* - Este es un segundo pulso, pero con un desplazamiento de la fase A del husillo. Las ventajas de usar tanto A como B son detección de dirección, mayor inmunidad al ruido y mayor resolución.

Si aparecen *Fase A de husillo* e *Índice de husillo* en el pinout, se debe ingresar la siguiente información:

- *Usar Spindle-At-Speed* - Con la retroalimentación del encoder se puede hacer que linuxcnc espere a que el husillo alcance la velocidad ordenada antes de que se mueva la alimentación. Seleccione esta opción y establezca la escala *close enough*.
- *Ganancia del filtro de pantalla de velocidad* - Configuración para ajustar la estabilidad de la visualización de la velocidad del husillo.
- *Ciclos por revolución* - El número de ciclos de la señal A del husillo durante una revolución. Esta opción solo está habilitada cuando una entrada se ha configurado como *Fase A del husillo*
- *Velocidad máxima en roscado* - La velocidad máxima del husillo utilizada en el roscado. Para un husillo de altas RPM o un encóder de husillo con alta resolución, es obligatorio un valor bajo de *BASE_PERIOD* .

3.7.9.3. Determinacion de la calibracion del husillo

Ingrese los siguientes valores en la página Configuración del husillo:

Velocidad 1:	0	PWM 1:	0
Velocidad 2:	1000	PWM 2:	1

Termine los pasos restantes del proceso de configuración, luego, inicie LinuxCNC con su configuración. Encienda la máquina y seleccione la pestaña MDI. Inicie el giro del husillo ingresando: *M3 S100*. Cambie la velocidad del husillo ingresando un número S diferente: *S800*. Los números válidos (en este momento) van de 1 a 1000.

Para dos números S diferentes, mida la velocidad real del eje en RPM. Registre los números S y las velocidades reales del eje. Ejecute Stepconf nuevamente. Para *Velocidad*, ingrese la velocidad medida, y para *PWM* ingrese el número S dividido entre 1000.

Ya que la mayoría de los controladores de husillo son ligeramente no lineales en sus curvas de respuesta, lo mejor es:

- Asegúrese de que las dos velocidades de calibración no estén demasiado juntas en RPM
- Asegúrese de que las dos velocidades de calibración estén en el rango de velocidades que típicamente usará durante el fresado

Por ejemplo, si su husillo va de 0 RPM a 8000 RPM, pero generalmente usa velocidades de 400 RPM (10 %) a 4000 RPM (100 %), encuentre los valores de PWM que dan 1600 RPM (40 %) y 2800 RPM (70 %).

3.7.10. Configuracion de la maquina completa

Haga clic en *Aplicar* para escribir los archivos de configuración. Más tarde, puede volver a ejecutar este programa y ajustar la configuración que ingresó antes.

3.7.11. Recorrido de eje y home

Para cada eje, hay un rango limitado de recorrido. El final físico del recorrido se llama *parada dura o hard*.

Antes de la "parada dura" hay un "interruptor de límite". Si se encuentra el interruptor de límite durante la operación normal, LinuxCNC apaga el amplificador de motor. La distancia entre la "parada dura" y el "interruptor de límite" debe ser lo suficientemente larga como para permitir que un motor sin alimentacion se detenga.

Antes del *interruptor de límite* hay un *límite suave o soft*. Este es un límite impuesto en el software después de home. Si un comando MDI o un programa de código g superara el límite soft, no se ejecutará. Si un desplazamiento pasa el límite suave, se detiene en el límite suave.

El *interruptor home* se puede colocar en cualquier lugar dentro del recorrido (entre paradas duras). Siempre que el hardware externo no desactive los amplificadores de motor cuando se alcanza el interruptor de límite, uno de los interruptores de límite puede ser utilizado como un interruptor home.

La *posición cero* es la ubicación en el eje que es 0 en el sistema de coordenadas de la máquina. Por lo general, la "posición cero" estará dentro de los "límites suaves". En los tornos, el modo de velocidad de superficie constante requiere que *X = 0* en la máquina corresponda al centro de rotación del husillo cuando no está activo el offset de la herramienta.

La *posición home* es la ubicación dentro del recorrido a la que el eje sera movido al final de la secuencia de home. Este valor debe estar dentro de los "límites suaves". En particular, la *posición home* nunca debe ser exactamente igual a un *límite suave*.

3.7.11.1. Operando sin interruptores de limite

Una máquina puede ser operada sin interruptores de límite. En este caso, solo los límites suaves impiden que la máquina alcance la parada dura. Los límites suaves solo funcionan después de que la máquina ha sido puesta a home.

3.7.11.2. Operando sin Switches Home

Una máquina puede operarse sin interruptores home. Si la máquina tiene interruptores de límite, pero no hay interruptores home, lo mejor es utilizar un interruptor de límite como el interruptor home (Por ejemplo, elija *Límite mínimo + Home X* en el pinout). Si la máquina no tiene interruptores, o los interruptores de límite no pueden ser utilizados como interruptores de inicio por otra razón, entonces la máquina debe ponerse *a ojo* en home o usando marcas de coincidencia. Homing a ojo no es tan repetible como el home por interruptores, pero aún permite que los límites suaves sean de utilidad.

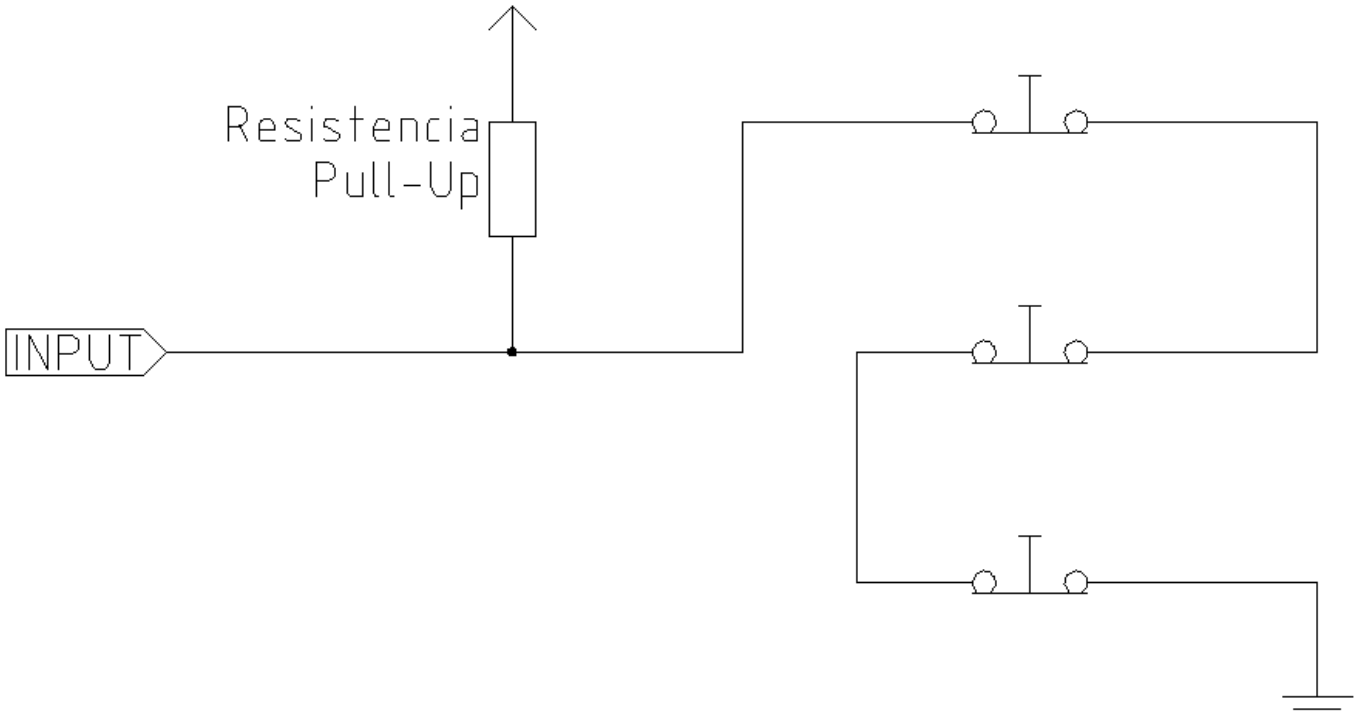
3.7.11.3. Opciones de cableado del interruptor de home y de limite

El cableado ideal para interruptores externos sería de una entrada por interruptor. Sin embargo, el puerto paralelo de PC solo ofrece un total de 5 entradas, mientras que hay hasta 9 interruptores en una máquina de 3 ejes. En cambio, múltiples interruptores pueden conectarse entre sí en varias formas para que se requiera un menor número de entradas.

Las siguientes figuras muestran la idea general de cablear múltiples interruptores a un solo pin de entrada. En cada caso, cuando se activa un interruptor, el valor visto en ENTRADA va de lógica ALTA a BAJA. Sin embargo, LinuxCNC espera un valor VERDADERO cuando se cierra un interruptor, por lo que el correspondiente cuadro *Invert* debe verificarse en la página de configuración del pinout. El resistor pull-up que se muestra en los diagramas mantiene la entrada alta hasta que se realice la conexión a tierra y la entrada pasa a baja. Sin resistencia, la entrada puede flotar entre encendido y apagado cuando el circuito está abierto. Normalmente, para un puerto paralelo, puede usar resistencias de 47k.

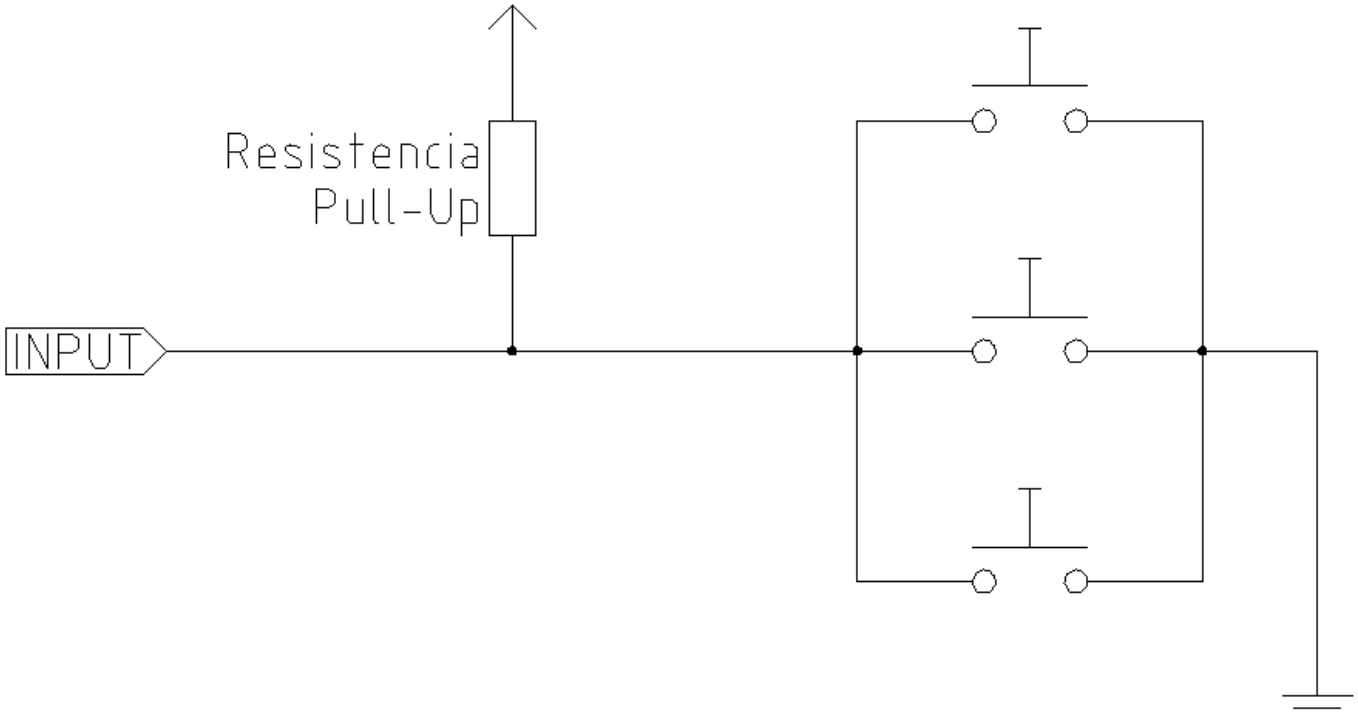
1. Interruptores normalmente cerrados

Cableado de interruptores N/C en serie (diagrama simplificado)



1. Interruptores normalmente abiertos

Cableado de interruptores de N/O en paralelo (diagrama simplificado)



Las siguientes combinaciones de interruptores están permitidas en Stepconf:

- Combinar los interruptores de home para todos los ejes
- Combinar los interruptores de límite para todos los ejes
- Combinar ambos interruptores de límite para un eje
- Combinar ambos interruptores de límite y el interruptor de home para un eje
- Combinar un interruptor de límite y el interruptor de home para un eje

3.8. Asistente de configuracion Mesa

PNCconf está hecho para ayudar a construir configuraciones que utilizan productos específicos Mesa *Anything I/O*.

Puede configurar sistemas servo de bucle cerrado o sistemas de hardware paso a paso. Utiliza un enfoque de *asistente* similar a Stepconf (utilizado para sistemas accionados por pasos software, por puerto paralelo).

PNCconf aún se encuentra en una etapa de desarrollo (Beta), por lo que hay algunos errores y carece de algunas características. Informe de los errores y sugerencias a la página del foro de LinuxCNC o a la lista de correo.

Hay dos líneas accion cuando se utiliza PNCconf:

Una es usar PNCconf para configurar siempre su sistema; si decide cambiar algunas opciones, vuelva a cargar PNCconf y permita que configure las nuevas opciones. Esta actitud funciona bien si su máquina es bastante estándar y puede usar archivos personalizados para agregar características no estándar. PNCconf intenta trabajar con usted en este aspecto.

La otra es usar PNCconf para construir una configuración que esté cerca de lo deseado y luego editar a mano para adaptarlo a sus necesidades. Esta sería la elección si necesita modificaciones extensas más allá del alcance de PNCconf o solo quiere jugar con el o aprender LinuxCNC

Navege por las páginas del asistente con los botones de avanzar, retroceder y cancelar. Hay también un botón de ayuda que brinda información sobre las páginas, diagramas y una página de salida.

SUGERENCIA: la página de ayuda de PNCconf debe tener la información más actualizada e informaciónde detalles adicional.

3.8.1. Instrucciones punto por punto

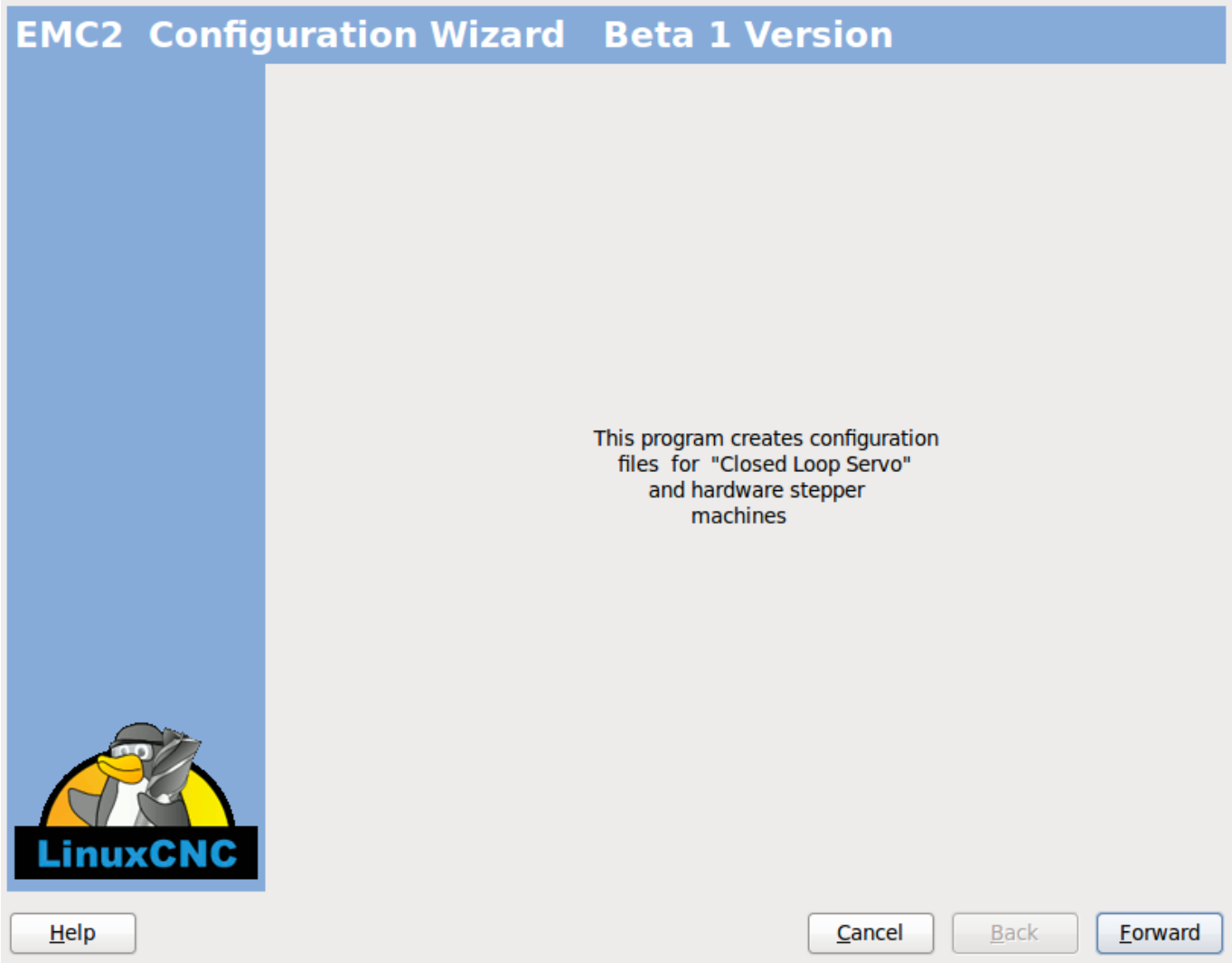


Figura 3.16: Ventana Splash PnCConf

3.8.2. Crear o editar

Se permite seleccionar una configuración previamente guardada o crear una nueva. Si elige *Modificar una configuración* y luego presiona siguiente, se mostrara un cuadro de selección de archivos. Pncconf preselecciona su último archivo guardado. Elija la configuración que usted desea editar. Si realizó algún cambio en los archivos principales hal o ini **Pncconf sobre-escribira** esos archivos y los cambios se perderán. Algunos archivos no serán sobre-escritos y Pncconf colocara una nota en esos archivos. También le permite seleccionar las opciones de atajo de escritorio/lanzador. Un atajo de escritorio colocará un icono de carpeta en el escritorio que apunta a sus nuevos archivos de configuración. De lo contrario, tendría que buscar en su carpeta home bajo linuxcnc/configs.

Un lanzador de escritorio agregará un icono al escritorio para iniciar su configuración directamente. También puede iniciarlo desde el menú principal utilizando el Selector de Configuración *LinuxCNC* encontrado en el menú CNC y seleccionando su nombre de configuración.

3.8.3. Informacion basica de la maquina

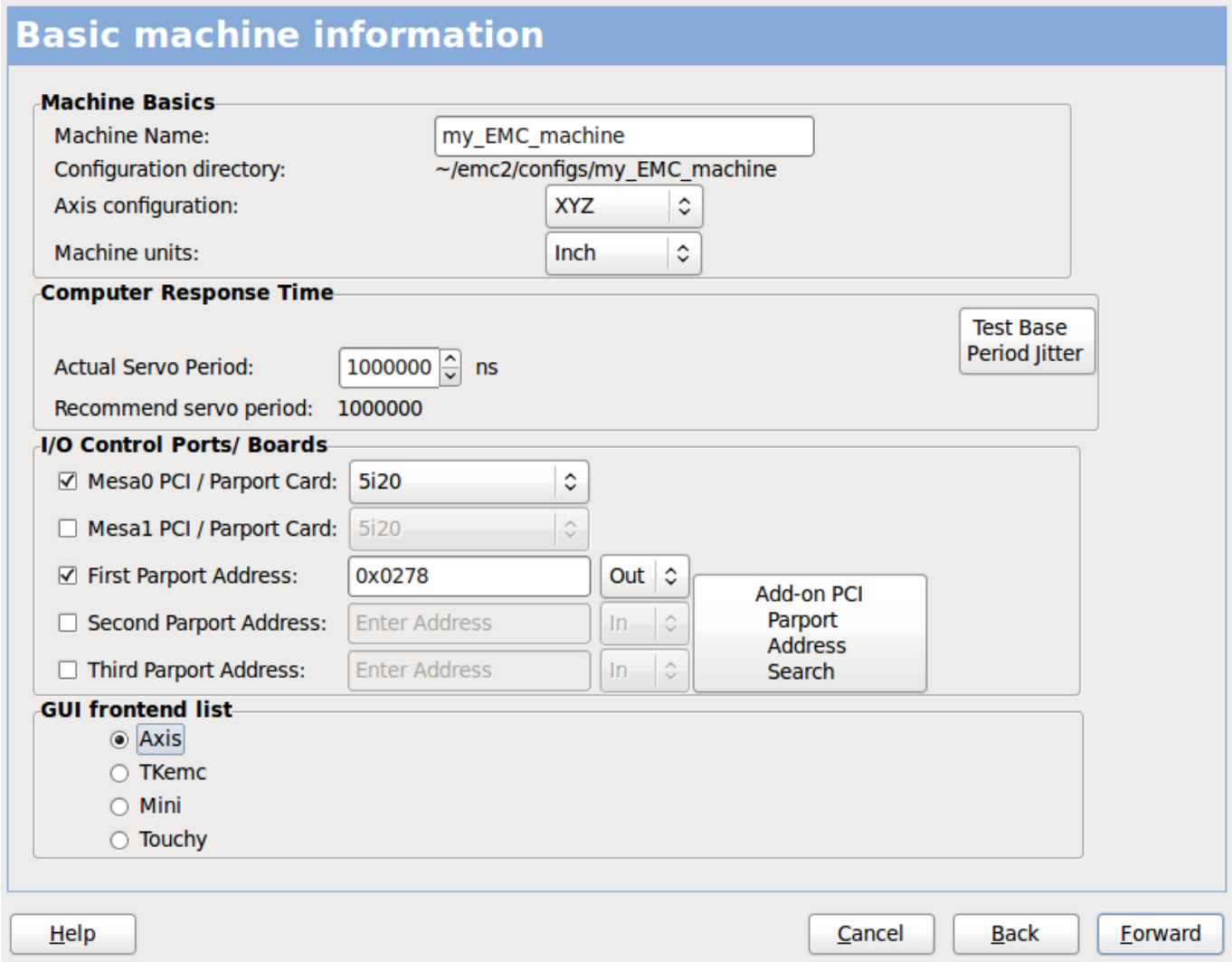


Figura 3.17: PnCConf Basico

Conceptos básicos de la máquina

Si usa un nombre con espacios, PNCconf reemplazará los espacios con guiones bajos (como regla general, a Linux no le gustan los espacios en los nombres). Elija una configuración de ejes; esto selecciona qué tipo de máquina está construyendo y qué ejes están disponibles. El selector de unidades de máquina permite la entrada de datos de unidades métricas o imperiales en las siguientes páginas.

SUGERENCIA: los valores predeterminados no se convierten cuando se usa métrica, así que asegúrese de que sean valores correctos.

Tiempo de respuesta del equipo

El período servo configura el latido del corazón del sistema. La latencia se refiere a la cantidad de tiempo de computadora que puede ser más largo que ese período. Como un ferrocarril, LinuxCNC requiere todo en una línea de tiempo muy estrecha y consistente o pasaran cosas malas. LinuxCNC requiere y usa un sistema operativo *en tiempo real*, que solo significa que tiene un tiempo de respuesta de baja latencia (retraso) cuando LinuxCNC requiere sus cálculos y, cuando los esta haciendo, no puede ser interrumpido por solicitudes de menor prioridad (como la entrada del usuario a botones de pantalla, dibujo, etc).

Probar la latencia es muy importante y una cosa clave a verificar en primer lugar. Afortunadamente, usando las tarjetas Mesa para hacer el trabajo que requiere el tiempo de respuesta más rápido (conteo de encoders y generación de PWM) podemos soportar mucha más latencia, y entonces podemos usar el puerto paralelo para otras cosas. La prueba estándar en LinuxCNC es verificar la latencia del período BASE (aunque no estemos utilizando un período base). Si se presiona el botón de prueba de jitter de periodo base, se inicia la ventana de la prueba de latencia (también puede cargar esto directamente desde el panel de aplicaciones/cnc). La prueba debe ejecutarse durante unos minutos, cuantos más, mejor. Considerar 15 minutos mínimo, o una noche entera aún mejor. En este momento use la computadora para cargar cosas, usar la red, usar USB, etc. Queremos saber el peor caso de latencia y para averiguar si alguna actividad particular perjudica nuestra latencia. Necesitamos mirar el jitter de período base. Cualquier cosa por debajo de 20000 es excelente, incluso podrías hacer pasos software rápidos con la máquina. De 20000 a 50000 sigue siendo bueno para pasos software y está bien para nosotros. De 50000 a 100000 ya no es tan bueno, pero aún se puede usar con tarjetas hardware que se encargen de las cosas de respuesta rápida. Cualquier cosa por debajo de 100000 es útil para nosotros. Si la latencia es decepcionante o si tiene un mal jitter periódicamente, todavía es posible mejorarlo.

SUGERENCIA: hay una lista de equipos compilada por los usuarios con la latencia obtenida en la wiki de LinuxCNC: http://wiki.linuxcn cgi-bin/wiki.pl?Latency-Test Considere agregar su información a la lista. También en esa página hay enlaces a información sobre cómo solucionar algunos problemas de latencia.

Una vez conformes con la latencia se debe elegir un período servo. En la mayoría de los casos, un período servo de 1000000 ns esta bien (eso da una tasa de cálculo del servo de 1 kHz - 1000 cálculos por segundo). Si está construyendo un servosistema de lazo cerrado que controla el par (corriente) en lugar de velocidad (voltaje), una tasa más rápida sería mejor; algo así como 200000 (velocidad de cálculo de 5 kHz). El problema con la velocidad del servo es que deja menos tiempo disponible para que la computadora haga otras cosas además de cálculos de LinuxCNC. Normalmente, la pantalla (GUI) se vuelve menos receptiva. Usted debe decidir sobre el equilibrio. Tenga en cuenta que si sintoniza su servosistema de lazo cerrado y luego cambia el período del servo, probablemente necesite volver a sintonizarlo.

Puertos/tarjetas de control de E/S

PNCconf es capaz de configurar máquinas que tienen hasta dos placas Mesa y tres puertos paralelos. Los puertos paralelos solo se pueden usar para E/S de baja velocidad (velocidad servo).

Mesa

Debe elegir al menos una placa Mesa ya que PNCconf no configurará los puertos paralelos para contar encoders o pasos de salida o PWM. Las tarjetas Mesa disponibles en el cuadro de selección se basan en lo que PNCconf encuentra en el firmware del sistema. Hay opciones para agregar firmware personalizado y/o *blacklist* (ignorar) algun firmware o tarjetas usando un archivo de preferencias. Si no se encuentra firmware, PNCconf mostrará una advertencia y usará un ejemplo interno de firmware - no se podrán realizar pruebas. Un punto a tener en cuenta es que si se eligen dos tarjetas PCI Mesa, actualmente no hay manera de predecir qué tarjeta es 0 y cual es 1 - se debe probar - mover las tarjetas podría cambiar su orden. Si se configuran dos tarjetas, ambas deben instalarse para que las pruebas funcionen.

Puerto paralelo

Se pueden usar hasta 3 puertos paralelos (denominados parports) como E/S simple. Usted debe establecer la dirección del parport. Puede ingresar el puerto paralelo de Linux segun su sistema de numeración de puertos (0, 1 o 2) o ingresar la dirección real. La dirección de un parport en placa base es a menudo 0x0278 o 0x0378 (escrito en hexadecimal) pero puede ser encontrado en la página de BIOS. La página del BIOS se encuentra cuando usted arranca su computadora. Debe presionar una tecla para entrar en ella (como F2). En la página BIOS puede encontrar la dirección del puerto paralelo y configurarlo en el modo SPP, EPP, etc. en algunas computadoras esta información se muestra durante unos segundos durante el inicio. Para tarjetas de puerto paralelo PCI, la dirección se puede encontrar presionando el boton *buscar dirección de parport*. Aparece la página de salida de ayuda con una lista de todos los dispositivos PCI que se pueden encontrar. Debe haber una referencia a un dispositivos puerto paralelo con una lista de direcciones. Una de esas direcciones debería funcionar. No todos los puertos paralelos PCI funcionan correctamente. Cualquiera de los tipos se puede seleccionar como *in* (máximo cantidad de pines de entrada) o *out* (cantidad máxima de pines de salida)

Lista de interfaz de usuario GUI

Esto especifica las pantallas gráficas que usará LinuxCNC. Cada una tiene unas opciones diferentes.

AXIS

- totalmente compatible con tornos.

- es el front-end más desarrollado y utilizado
- está diseñado para ser utilizado con mouse y teclado
- está basado en tkinter, por lo que integra PYVCP (control virtual basado en python) de forma natural.
- tiene una ventana gráfica 3D.
- permite VCP integrado en el lateral o en la pestaña central

TOUCHY

- Touchy fue diseñado para ser utilizado con una pantalla táctil, con los mínimos interruptores físicos y un volante MPG.
- Requiere botones para ciclo de inicio, aborto y señales de un solo paso
- También requiere que se seleccione jogging MPG de eje compartido.
- está basado en GTK por lo que integra GLADE VCP (paneles de control virtual) de forma natural.
- permite paneles VCP integrados en la pestaña central
- no tiene ventana gráfica
- el aspecto se puede cambiar con temas personalizados

TkLinuxCNC

- pantalla azul de alto contraste
- ventana de gráficos separada
- sin integración de VCP

3.8.4. Configuracion externa

Esta página le permite seleccionar controles externos como jogging o mando manual de velocidades.

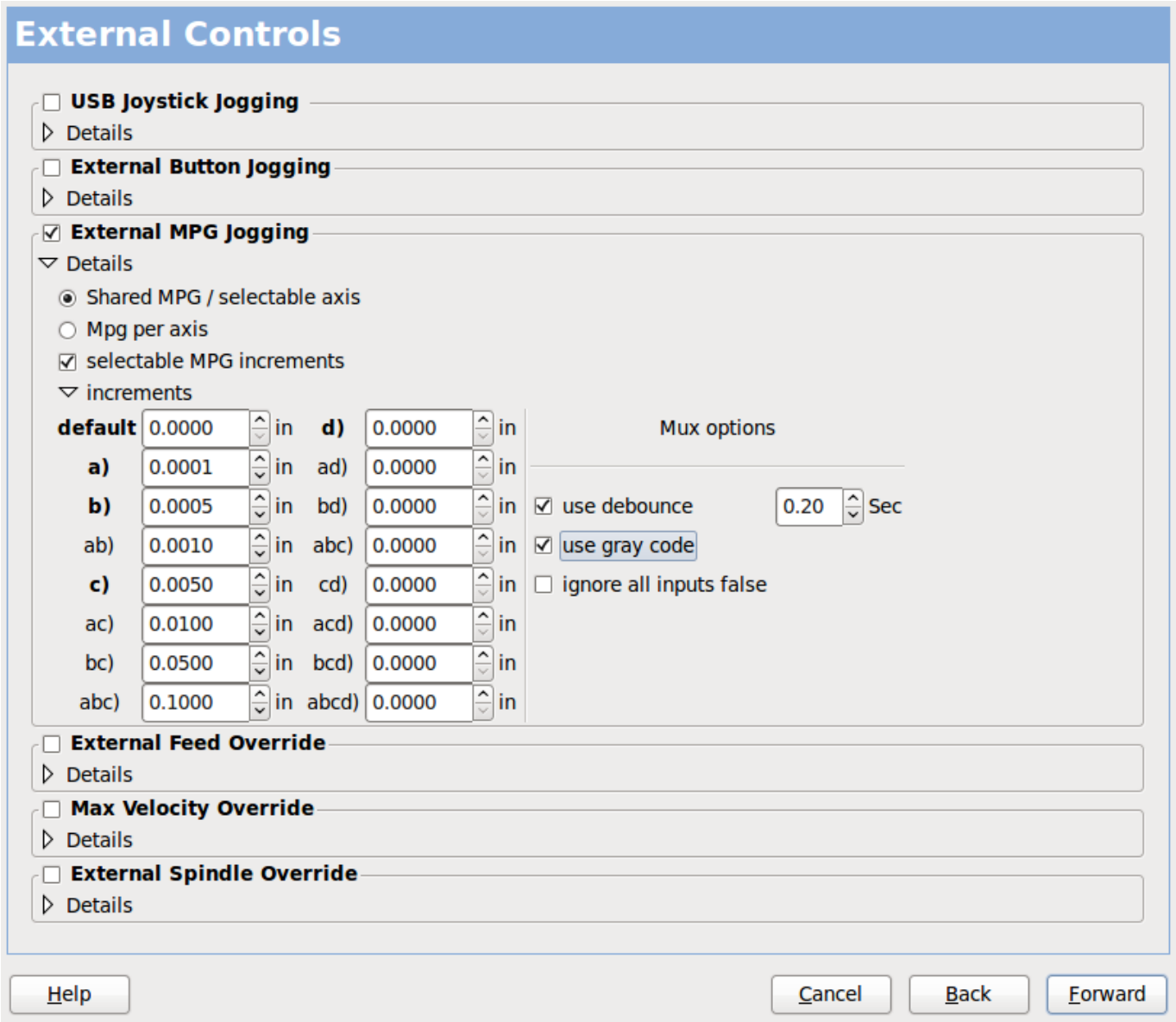


Figura 3.18: GUI Externo

Si selecciona un Joystick para jogging, lo necesitará siempre conectado para que LinuxCNC lo cargue. Para usar los sticks analógicos para un jogging útil probablemente necesite agregar algún código HAL personalizado. El jogging MPG requiere un generador de impulsos conectado a un contador de encoder MESA. Los controles de mando manual pueden usar un generador de impulsos (MPG) o interruptores (como un dial giratorio). Los botones externos se pueden usar con los interruptores de un joystick OEM.

Joystick para jogging

Requiere instalar una *regla de dispositivo* (devive rule) personalizada en el sistema. Este es un archivo que LinuxCNC usa para conectarse a la lista de dispositivos de LINUX. PNCconf le ayudará a hacer ese archivo.

Buscar regla de dispositivo buscará reglas en el sistema, puede usar esto para encontrar el nombre de los dispositivos que ya ha construido con PNCconf.

Agregar una regla de dispositivo le permitirá configurar un nuevo dispositivo siguiendo las indicaciones. Necesitará que su dispositivo este disponible.

prueba de dispositivo le permite cargar un dispositivo, ver los nombres de sus pines y verificar su funciones con halmeter

El jogging con joystick usa componentes HALUI y hal_input.

Botones externos
permite el jog de eje con botones simples a una velocidad de jog específica. Probablemente lo mejor para jogging rápido.

MPG Jogging
Le permite usar un generador manual de impulsos para mover ejes de la máquina.

Los MPG a menudo se encuentran en máquinas de grado comercial. La salida de pulsos en cuadratura se pueden contar con un contador de encoder MESA. PNCconf permite un MPG por eje o un MPG compartido con todos los ejes. Permite la selección de velocidades de jogging usando interruptores o una sola velocidad.

La opción de incrementos seleccionables usa el componente mux16. Este componente tiene opciones como debounce y codigo Gray para ayudar a filtrar la entrada del interruptor.

Ajuste manual
PNCconf permite mando manual de velocidades de avance y/o velocidad del husillo con un generador de pulsos (MPG) o interruptores (por ejemplo, rotativos).

3.8.5. Configuracion de GUI

Aquí puede establecer valores predeterminados para las pantallas de visualización, agregar paneles de control virtual (VCP), y establecer algunas opciones de LinuxCNC ..

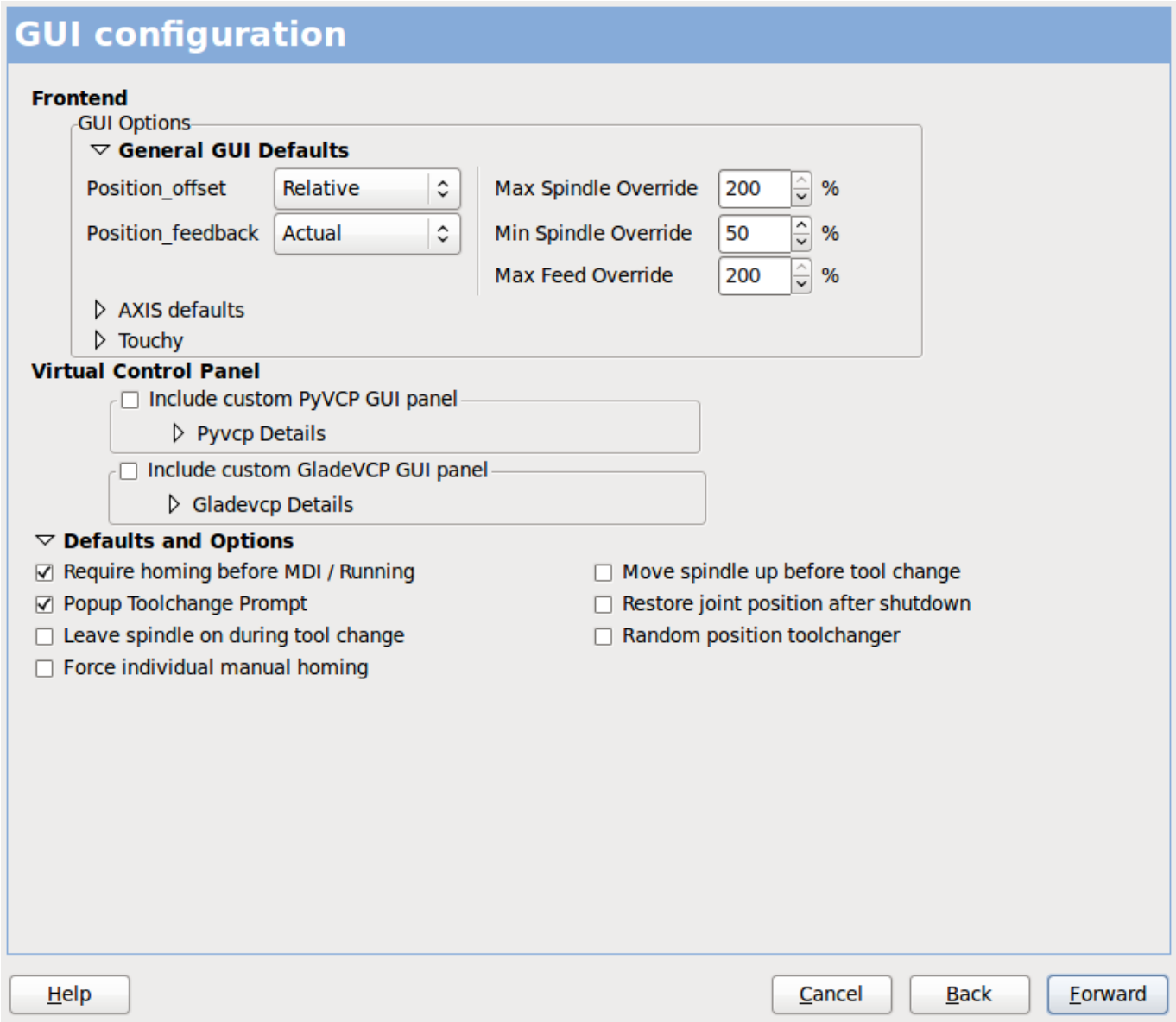


Figura 3.19: Configuración de GUI

Opciones de GUIs

Las opciones predeterminadas permiten elegir los valores predeterminados generales para cualquier pantalla de visualización.

Los valores predeterminados de AXIS son opciones específicas de AXIS. Si elige las opciones de tamaño, posición o forzar maximizar, PNCconf le preguntará si es correcto sobrescribir el archivo de preferencias (.axisrc). A menos que haya agregado comandos manualmente a este archivo, sera correcto permitirlo. La posición y forzar máximizar se pueden usar para mover AXIS a un segundo monitor si el sistema es capaz.

Los valores predeterminados de Touchy son opciones específicas de Touchy. La mayoría de las opciones de Touchy pueden ser cambiadas mientras Touchy se está ejecutando usando la página de preferencias. Touchy usa GTK para dibujar su pantalla, y soporta temas. Temas controla el aspecto básico y la *sensación* de un programa. Puede descargar temas de la red o editarlos usted mismo. Hay una lista de los temas actuales en la computadora entre los que puede elegir. Para ayudar a que parte del texto se destacara, PNCconf le permite anular los valores predeterminados de los temas. Las opciones de posición y forzar máximizar se pueden usar para mover Touchy a un segundo monitor si el sistema es capaz.

Opciones de VCP

Los paneles de control virtuales permiten agregar controles y pantallas personalizadas. AXIS y Touchy pueden integrar estos controles dentro de la pantalla en posiciones designadas. Hay dos tipos de VCP: PyVCP que usa *Tkinter* para dibujar la pantalla y GLADE VCP que usa *GTK* para dibuja la pantalla.

PyVCP

El archivo XML de las pantallas PyVCP solo se puede construir a mano. PyVCPs encajan naturalmente con AXIS ya que ambos usan TKinter.

Los pines HAL se crean para que el usuario se conecte dentro de su archivo HAL personalizado. Hay un panel de visualización de husillo de ejemplo para que el usuario lo use tal como está o lo use como base. Puede seleccionar un archivo en blanco en el que luego puede agregar los controles *widgets* o seleccionar un ejemplo de visualización del husillo que mostrará su velocidad e indicará si está a la velocidad requerida.

PNCconf conectará los pines HAL de visualización del husillo adecuados para usted. Si está utilizando AXIS, entonces el panel se integrará en el lado derecho. Si no utiliza AXIS, el panel se separará de la pantalla de interfaz.

Puede usar las opciones de geometría para ajustar el tamaño y mover el panel, por ejemplo para moverlo a una segunda pantalla si el sistema es capaz. Si presiona el botón *Display sample panel*, se respetarán las opciones de tamaño y ubicación.

GLADE VCP

GLADE VCPs encaja naturalmente dentro de la pantalla TOUCHY ya que ambos usan GTK para dibujar, pero al cambiar el tema de GLADE VCP se puede hacer que combine bastante bien con AXIS (pruebe Redmond).

Utilize un editor gráfico para construir sus archivos XML. Los pines HAL se crean para que el usuario se conecte, dentro de su archivo HAL personalizado.

GLADE VCP también permite una interacción de programación mucho más sofisticada (y complicada) que PNCconf actualmente no aprovecha (ver GLADE VCP en el manual).

PNCconf tiene paneles de muestra para que el usuario los utilice tal como están o compilados. Con GLADE VCP, PNCconf le permitirá seleccionar diferentes opciones en su muestra.

En "Opciones de muestra", seleccione cuáles le gustaría. Los botones cero usan comandos HALUI que puede editar más adelante en la sección HALUI.

Auto Z touch-off también requiere el programa touch-off de classic ladder y una entrada de sonda seleccionada. Requiere una placa conductora de toque y una herramienta conductora puesta a tierra. Para obtener una idea de cómo funciona, consulte:

http://wiki.linuxcnc.org/cgi-bin/wiki.pl?ClassicLadderExamples#Single_button_probe_touchoff

En *Opciones de visualización*, el tamaño, la posición y forzar máximizar se pueden usar en un panel *autónomo* para cosas tales como colocar la pantalla en un segundo monitor si el sistema es capaz

Puede seleccionar un tema GTK que establezca la apariencia básica del panel. Por lo general, deseara que esto coincida con la pantalla de la interfaz. Estas opciones se usarán si presiona el botón *Mostrar muestra*. Con GLADE VCP dependiendo de la pantalla frontal, puede seleccionar dónde se mostrará el panel.

Puede forzarlo a que sea independiente o, con AXIS, puede estar en el centro o en el lado derecho. Con Touchy puede estar en el centro.

Valores predeterminados y opciones

- Requiere homing antes de MDI/Running
  - Si desea poder mover la máquina antes del homing, desmarque esta casilla de verificación.
- Indicación emergente de herramienta
  - Para cambios de herramienta, elija entre un aviso en pantalla o exportación de nombres de señal estándar para un archivo Hal de cambiador de herramientas personalizado proporcionado por el usuario.
- Dejar el husillo encendido durante el cambio de herramienta:
  - Utilizado para tornos
- Forzar homing manual individual
- Mover el husillo hacia arriba antes de cambiar la herramienta

- Restaurar la posición de la articulación después del cierre
  - Utilizado para máquinas con cinemáticas no triviales
- Cambiadores de herramienta de posición aleatoria
  - Se usa para cambiadores de herramientas que no devuelven la herramienta a la misma ranura. Necesitará agregar código HAL personalizado para admitir cambiadores de herramientas.

3.8.6. Configuracion de tarjetas Mesa

Las páginas de configuración de Mesa le permiten utilizar diferentes firmwares. En la página básica seleccione una tarjeta Mesa, elija el firmware disponible y seleccione qué y cuántos componentes están disponibles.

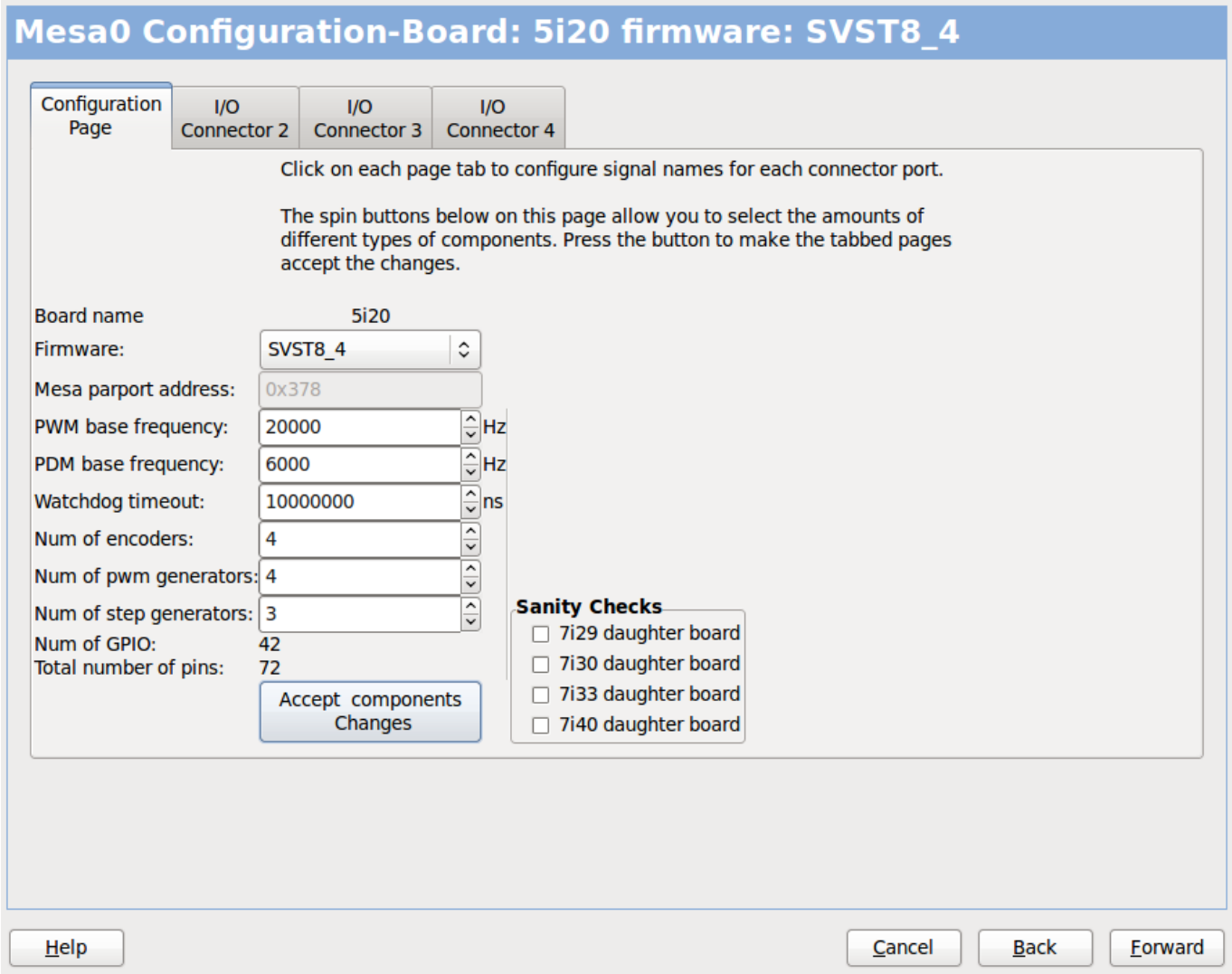


Figura 3.20: Configuración Mesa

La dirección de parport se usa solo con la tarjeta parport de Mesa, la 7i43. Los parport en placa base generalmente usan 0x278 o 0x378, aunque debería poder encontrar la dirección desde la página de BIOS. La 7i43 requiere que el puerto paralelo use el modo EPP, de nuevo establecido en la página de BIOS. Si usa un puerto paralelo PCI, la dirección puede ser buscada utilizando el botón de búsqueda en la página básica.

nota
Muchas tarjetas PCI no son compatibles con el protocolo EPP correcto.

La frecuencia base PDM PWM y 3PWM establece el equilibrio entre rizado y linealidad. Si usa tarjetas hijas Mesa, los documentos para la placa deben dar recomendaciones.

importante
Es importante seguir esto para evitar daños y obtener el mejor rendimiento.

```
La 7i33 requiere PDM y una frecuencia base PDM de 6 mHz
La 7i29 requiere PWM y una frecuencia base PWM de 20 Khz
La 7i30 requiere PWM y una frecuencia base PWM de 20 Khz
La 7i40 requiere PWM y una frecuencia base PWM de 50 Khz
La 7i48 requiere UDM y una frecuencia base PWM de 24 Khz
```

El tiempo de espera de Watchdog se usa para establecer cuánto tiempo esperará la placa MESA antes de matar las salidas si la comunicación se interrumpe desde la computadora. Por favor, recuerde que Mesa usa salidas "activas bajas" lo que significa que cuando el pin de salida está activado, sera bajo (aproximadamente 0 voltios) y si la salida es alta (aproximadamente 5 voltios), está apagado. Asegúrese de que su equipo es seguro cuando esté apagado (watchdog activado).

Puede elegir la cantidad de componentes disponibles anulando la selección de los no utilizados. No todos los tipos de componentes están disponibles con todos los firmware.

Elegir por debajo de la cantidad máxima de componentes permite ganar más pines GPIO. Si usa tarjetas hijas, tenga en cuenta que no debe deseleccionar los pines que usa la tarjeta. Por ejemplo, algunos firmware admiten dos tarjetas 7i33; si solo tiene una puede anular la selección de suficientes componentes para utilizar el conector que admite la segunda 7i33. Los componentes son deseleccionados numéricamente por el número más alto primero y siguiendo sin saltar números. Si lo hace, los componentes estaran no donde los quiere, y entonces debe usar un firmware diferente. El firmware dicta dónde, qué y las cantidades máximas de los componentes. Es posible un firmware personalizado. Pregunte al contactar a los desarrolladores de LinuxCNC y Mesa. Usar firmware personalizado en PNCconf requiere procedimientos especiales y no siempre es posible, aunque se intenta hacer que PNCconf sea lo más flexible posible.

Después de elegir todas estas opciones, presione el botón *Aceptar cambios de componentes* y PNCconf actualizará las páginas de configuración de E/S. Solo se mostrarán las pestañas de E/S para los conectores disponibles, dependiendo de la placa Mesa.

3.8.7. Configuracion de E/S Mesa

Las pestañas se utilizan para configurar los pines de entrada y salida de las placas Mesa. PNCconf le permite crear nombres de señal personalizados para usar en archivos HAL personalizados.

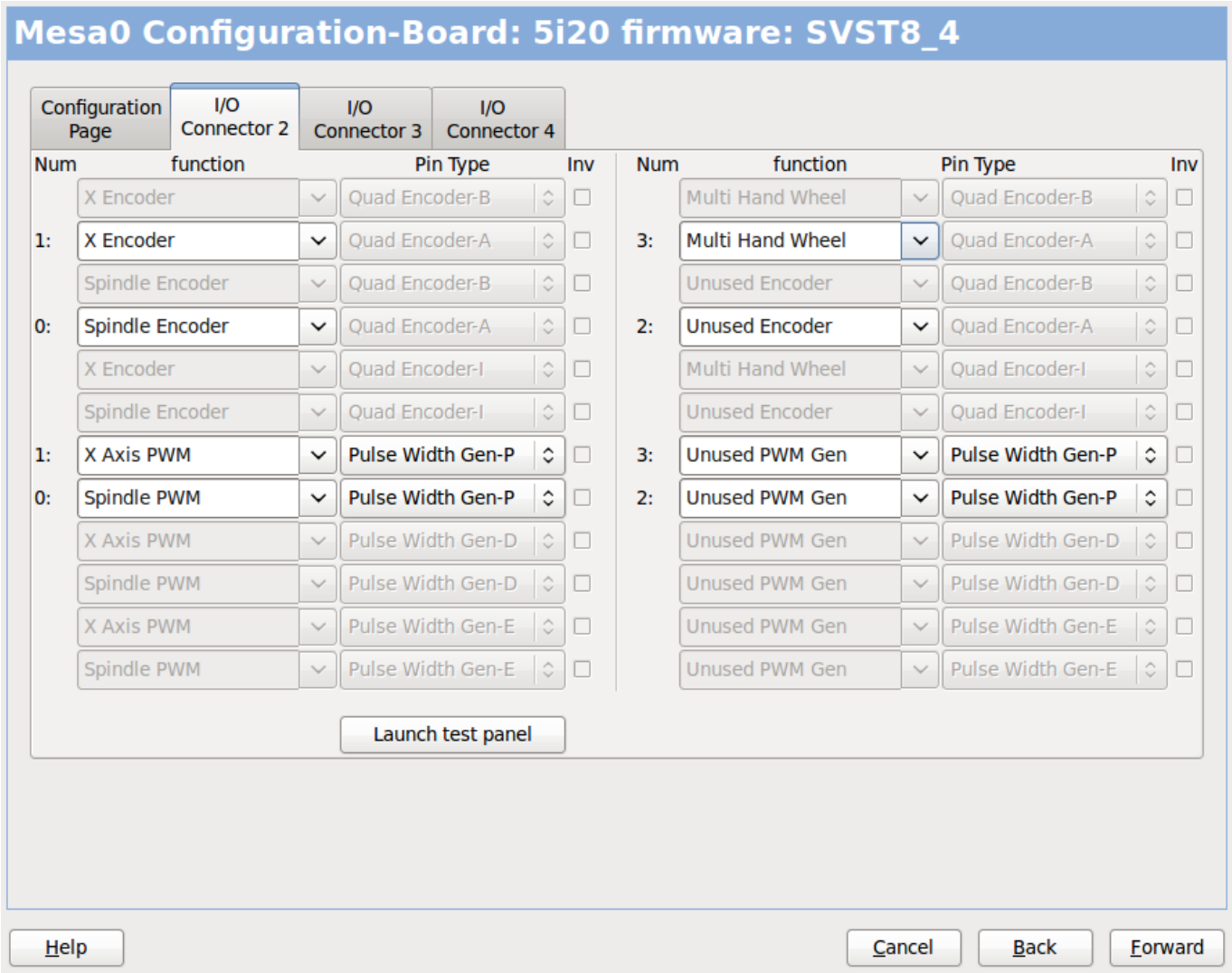


Figura 3.21: Mesa I/O C2

En esta pestaña con este firmware, los componentes están configurados para una tarjeta hija 7i33, generalmente utilizada con servos de lazo cerrado. Tenga en cuenta que los números de componente de los contadores de encoder y los controladores PWM no están en orden numérico. Siguen los requisitos de la tarjeta hija.

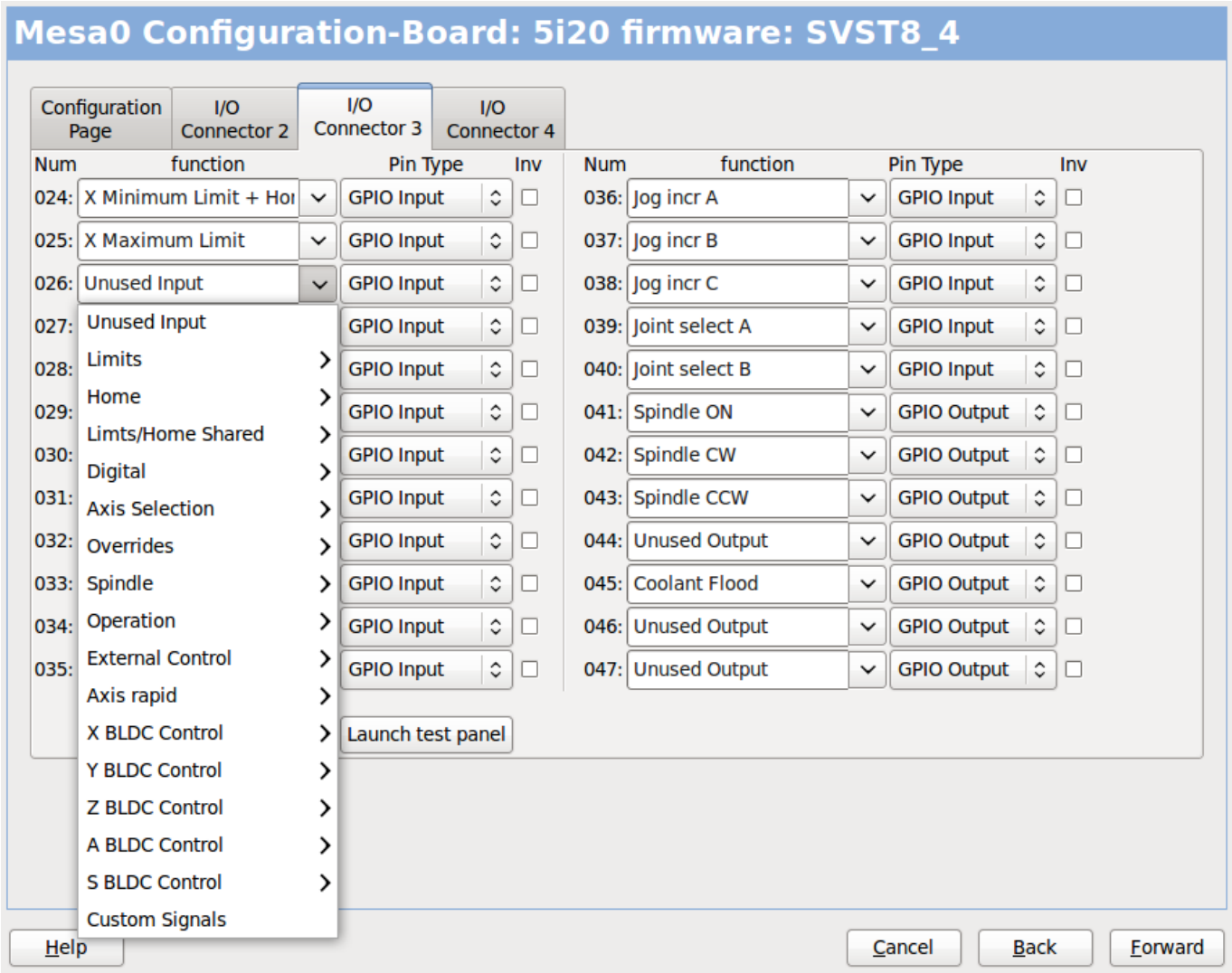


Figura 3.22: Mesa I/O C3

En esta pestaña, todos los pines son GPIO. Tenga en cuenta los números de 3 dígitos; coincidirán con el número de pin HAL. Los pines GPIO se pueden seleccionar como entrada o salida y se pueden invertir.

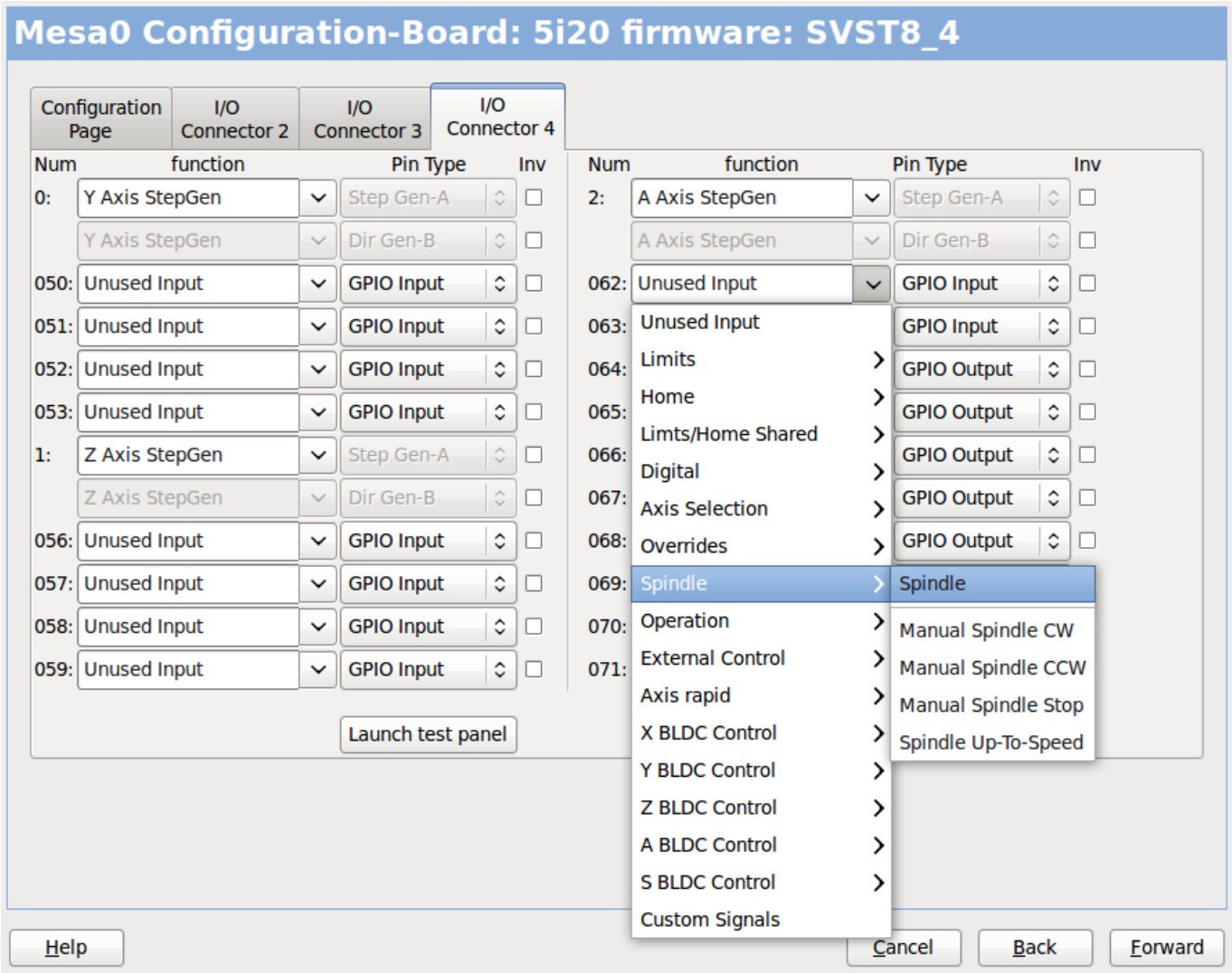
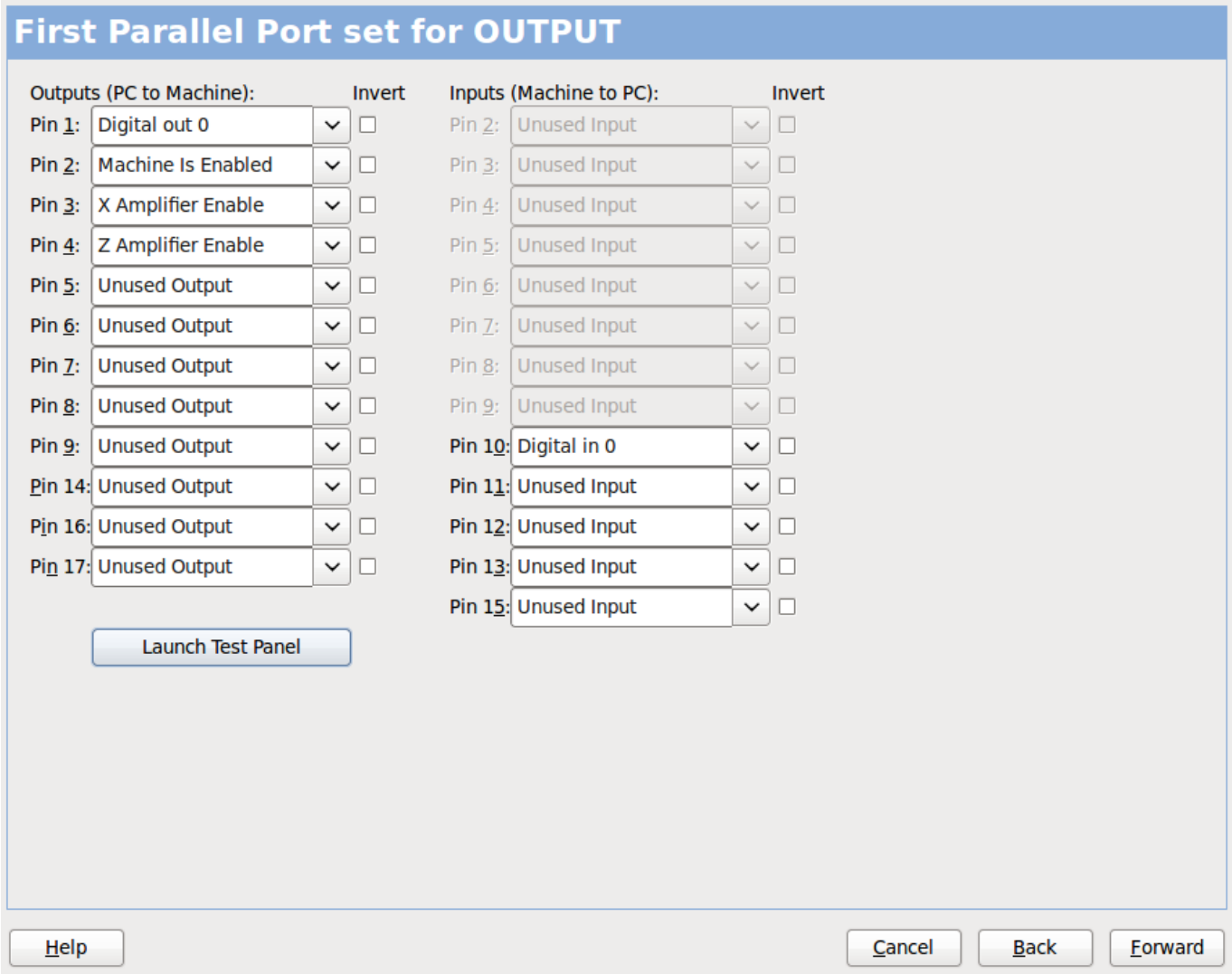


Figura 3.23: Mesa I/O C4

En esta pestaña hay una mezcla de generadores de pasos y GPIO. Los generadores de paso y los pines de dirección se pueden invertir. Tenga en cuenta que invertir un pin Step Gen-A (el pin de salida de paso) cambia el tiempo del paso. Debería coincidir con lo que espera su controlador

3.8.8. Configuracion de Puerto Paralelo



El puerto paralelo se puede usar para E/S simple, similar a los pines GPIO de Mesa.

3.8.9. Configuración de eje

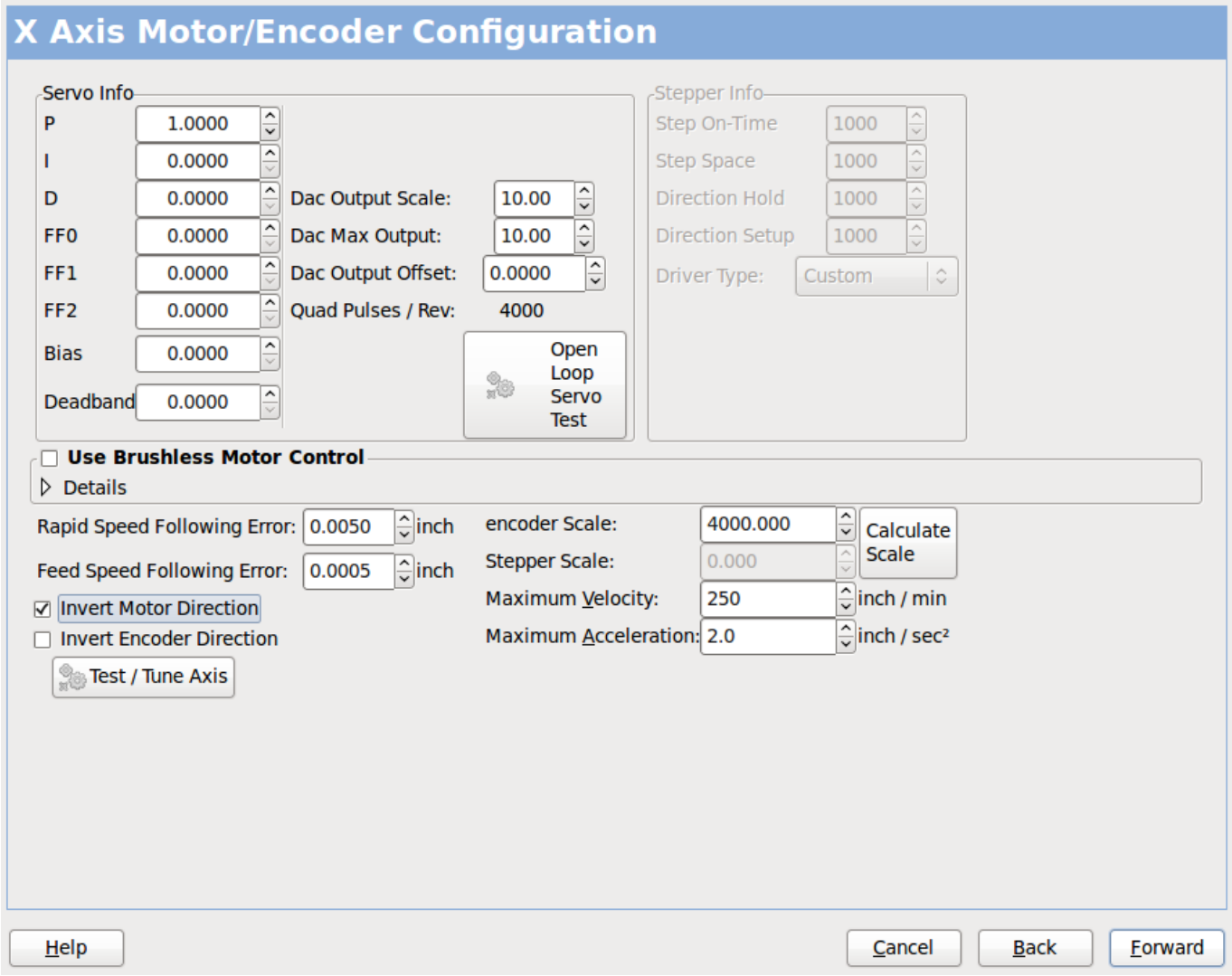


Figura 3.24: Ajuste del drive del eje

Esta página permite configurar y probar la combinación de motor y/o encoder. Si usa un servomotor, hay disponible una prueba de lazo abierto. Si usa un paso a paso, hay disponible una prueba de afinación.

Prueba de lazo abierto

La prueba de lazo abierto es importante ya que confirma la dirección del motor y encoder. El motor debe mover el eje en la dirección positiva cuando se presiona el botón positivo y también el encoder debe contar en positivo. El movimiento del eje debe seguir el estandar del manual de maquinaria [5] o la pantalla gráfica AXIS no tendrá mucho sentido. Esperemos que la página de ayuda y los diagramas le ayudan a resolver esto. Tenga en cuenta que las direcciones de los ejes se basan en movimiento de la HERRAMIENTA, no en movimiento de la mesa. No hay rampa de aceleración con lazo abierto; pruebe comenzando con números de DAC bajos. Al mover el eje una distancia conocida, se puede confirmar la escala del encoder. El encoder debe contar incluso sin el amplificador habilitado dependiendo de cómo se suministra energía al mismo.

ADVERTENCIA: Si el motor y el codificador no están de acuerdo con la dirección de conteo, entonces el el servo se descontrolara cuando use el control PID.

[5] "nomenclatura de los ejes" en el capítulo "Control Numérico" en "Machinery's Handbook" publicado por Industrial Press.

Dado que en este momento no se puede probar la configuración PID en PNCconf, la configuración sera real cuando reedite/ingrese una configuración PID probada.

Escalado DAC, salida máxima y offset se utilizan para adaptar la salida DAC.

Compute DAC

Estos dos valores son los factores de escala y compensación para la salida del eje al amplificadores de motor. El segundo valor (compensación) se resta del cálculo de salida (en voltios), y dividido por el primer valor (factor de escala), antes de ser escrito a los convertidores D/A. Las unidades en el valor de la escala están en voltios verdaderos por voltios de salida DAC. Las unidades en el valor de compensación están en voltios. Estos pueden ser utilizado para linearizar un DAC.

Específicamente, al escribir salidas, LinuxCNC primero convierte el deseado salida en unidades cuasi-SI a valores de actuador crudos, por ejemplo, voltios para un amplificador DAC. Esta escala se ve así: El valor de la escala se puede obtener analíticamente haciendo un análisis de unidad, es decir, las unidades son [unidades SI de salida] / [unidades de actuador]. Por ejemplo, en una máquina con un amplificador de modo de velocidad tal que 1 voltio da como resultado una velocidad de 250 mm / seg, tenga en cuenta que las unidades del desplazamiento están en la máquina unidades, por ejemplo, mm / seg, y se restan de las lecturas del sensor. los el valor de este desplazamiento se obtiene al encontrar el valor de su salida que rinde 0.0 para la salida del actuador. Si el DAC está linealizado, este desplazamiento es normalmente 0.0.

La escala y el offset se pueden usar también para linealizar el DAC, lo que da como resultado valores que reflejan los efectos combinados de la ganancia del amplificador, la no linealidad del DAC, unidades DAC, etc. Para ello, siga este procedimiento:

- Construya una tabla de calibración para la salida, indicando al DAC el voltaje deseado y midiendo el resultado:

Mediciones de tensión de salida

| ========== | * Raw * | **Mesurado** | -10 | * -9.93 * | -9 | * -8.83 * | 0 | * -0.96 * | 1 | * -0.03 * | 9 | * 9.87 * | 10 | * 10.07 * | ==========

- Haz un ajuste lineal de mínimos cuadrados para obtener los coeficientes a, b de modo que meas = a * raw + b
- Tenga en cuenta que queremos una salida bruta tal que nuestro resultado medido sea Idéntico a la salida ordenada. Esto significa
  - cmd = a * raw + b
  - crudo = (cmd-b) / a
- Como resultado, se pueden usar los coeficientes ayb del ajuste lineal como la escala y el offset para el controlador directamente.

MAX SALIDA: El valor máximo para la salida de la compensación PID que se escribe en el Amplificador motor, en voltios. El valor de salida calculado se fija a este límite. El límite se aplica antes de escalar a unidades de salida sin procesar. Se aplica el valor. simétricamente tanto para el lado positivo como para el negativo.

- Prueba de sintonía * La prueba de afinación desafortunadamente solo funciona con sistemas basados en pasos. Otra vez Confirmar que las direcciones en el eje son correctas. Luego prueba el sistema ejecutando el eje de ida y vuelta, si la aceleración o la velocidad máxima es demasiado alta, perder pasos Mientras trota, tenga en cuenta que puede tomar un tiempo para un eje con baja aceleración para detener. Los interruptores de límite no funcionan durante esta prueba. Tú Puede establecer un tiempo de pausa para cada final del movimiento de prueba. Esto te permitiría configure y lea un indicador de cuadrante para ver si está perdiendo pasos.
- Cronometraje * La sincronización paso a paso debe adaptarse a los requisitos del controlador de pasos. Pncconf proporciona algunos tiempos predeterminados del controlador o permite configuraciones de tiempo personalizadas
  1. Consulte http://wiki.linuxcnc.org/cgi-bin/wiki.pl?Stepper_Drive_Timing para algunos números de tiempo más conocidos (siéntase libre de agregar los que haya descubierto). Si en caso de duda, utilice números grandes como 5000, esto solo limitará la velocidad máxima.
- Control de motor sin escobillas * Estas opciones se utilizan para permitir un control de bajo nivel de motores sin escobillas utilizando Firmas especiales y pizarras hijas. También permite la conversión de sensores HALL. de un fabricante a otro. Sólo está parcialmente apoyado y lo hará requiere uno para terminar las conexiones HAL. Póngase en contacto con la lista de correo o foro para más ayuda.

Cálculo de la Escala de Axis

image
images / pncconf-scale-calc.png [alt = "Cálculo de la escala del eje"]

La configuración de la escala se puede ingresar directamente o se puede usar la "escala de cálculo" botón para ayudar. Use las casillas de verificación para seleccionar los cálculos apropiados. Nota que los *dientes de polea* requieren el número de dientes no la relación de engranaje. Gusano a su vez La relación es justo lo contrario, requiere la relación de transmisión. Si estas feliz con el escala presionar aplicar de lo contrario presionar cancelar e ingresar la escala directamente.

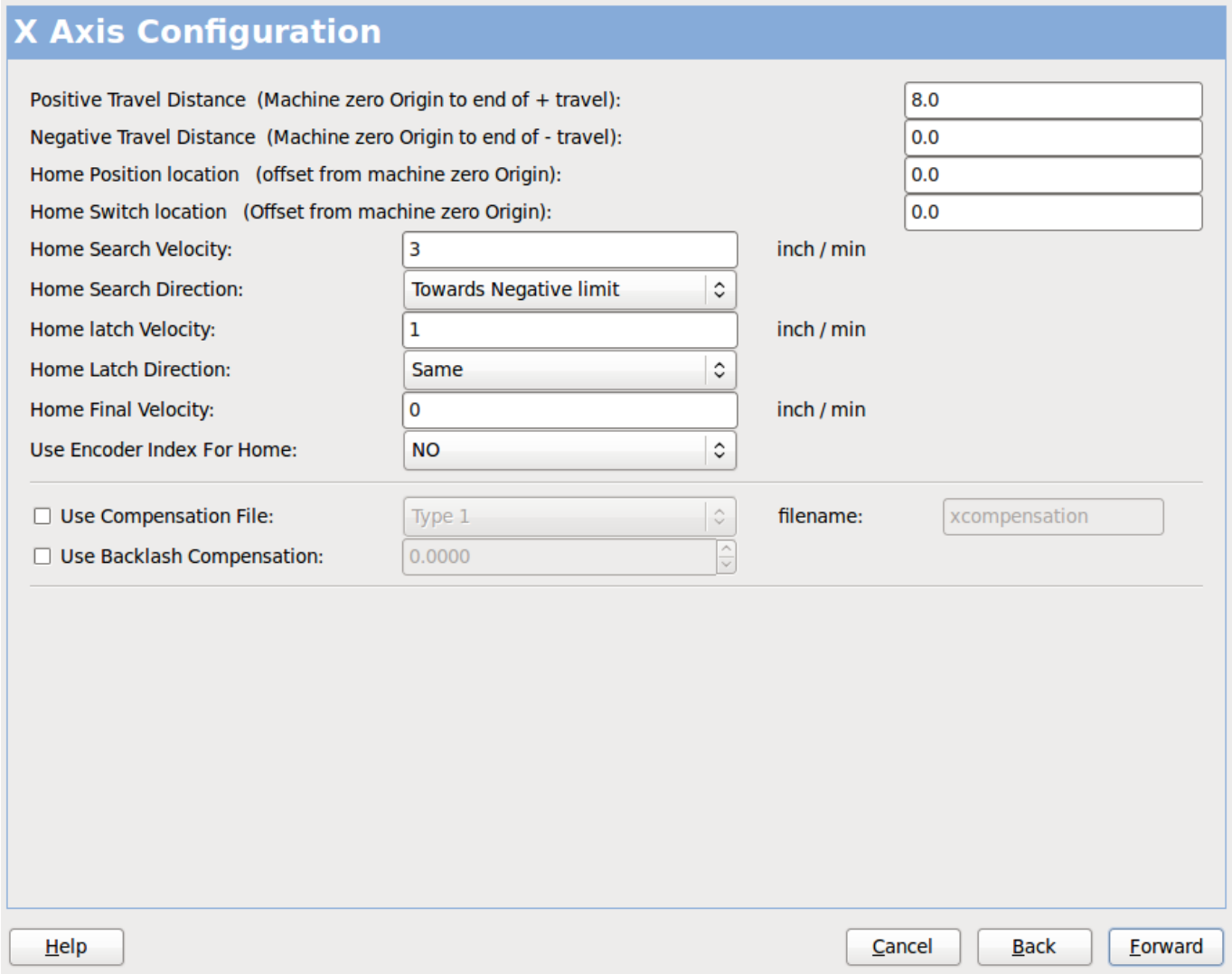


Figura 3.25: Axis de configuración

Consulte también la pestaña del diagrama para ver dos ejemplos de Inicio y finales de carrera. Estos son dos ejemplos de Muchas formas diferentes de establecer homing y límites.

IMPORTANTE: Es muy importante comenzar con el eje moviéndose hacia la derecha. ¡La dirección o, de lo contrario, llegar a casa bien es muy difícil!

Recuerda direcciones positivas y negativas. Consulte la HERRAMIENTA, no la tabla según el manual de Maquinistas.

EN UN TÍPICO MOLINO DE RODILLA O CAMA

- cuando la TABLA se mueve hacia fuera, es la dirección positiva de Y
- cuando la TABLA se mueve a la izquierda, esa es la dirección X positiva
- cuando la TABLA se mueve hacia abajo, esa es la dirección Z positiva
- cuando la CABEZA se mueve hacia arriba, esa es la dirección Z positiva

EN UN TORNO TIPICO

- cuando la HERRAMIENTA se mueve hacia la derecha, lejos del mandril
- Esa es la dirección positiva de Z
- cuando la HERRAMIENTA se mueve hacia el operador
- Esa es la dirección X positiva. Algunos tornos tienen X
- opuesto (p. ej., herramienta en la parte posterior), que funcionará bien pero
- La pantalla gráfica AXIS no se puede hacer para reflejar esto.

Cuando se utilizan mandos de retorno y / o limitadores LinuxCNC espera que las señales HAL sean verdaderas cuando El interruptor está siendo presionado / disparado. Si la señal es incorrecta para un interruptor de límite entonces LinuxCNC pensará que la máquina está al final del límite todo el tiempo. Si la lógica de búsqueda del interruptor de casa es incorrecta LinuxCNC parecerá a casa en la dirección equivocada. Lo que realmente está haciendo es intentar retroceder El interruptor de la casa.

Decidir sobre la ubicación del interruptor de límite.

Los interruptores de límite son la copia de seguridad de los límites de software en el caso algo eléctrico va mal por ejemplo. Servo Runaway. Los interruptores de límite deben colocarse de manera que la máquina no Golpea el extremo físico del movimiento del eje. Recuerda el eje pasará por el punto de contacto si se mueve rápido. Finales de carrera Debe estar *bajo activo* en la máquina. p.ej. el poder corre a través los interruptores todo el tiempo - se dispara una pérdida de potencia (interruptor abierto). Si bien uno podría conectarlos a la otra forma, esto es a prueba de fallas. Es posible que deba invertirse para que la señal HAL en LinuxCNC en *alto activo* - una VERDADERA significa que el interruptor se disparó. Cuando iniciando LinuxCNC si recibe una advertencia de límite y el eje NO está accionando el interruptor, invirtiendo la señal es probablemente el solución. (use HALMETER para verificar la señal HAL correspondiente p.ej. joint.0.pos-lim-sw-in interruptor de límite positivo del eje X)

Decida la ubicación del interruptor de la casa.

Si está utilizando interruptores de límite, también puede utilizar uno como interruptor de la casa. Un interruptor de inicio separado es útil si tiene un largo eje que en uso es generalmente un largo camino desde los finales de carrera o Mover el eje hacia los extremos presenta problemas de interferencia. con material. por ejemplo, un eje largo en un torno hace que sea difícil llegar a los límites sin tener que La herramienta golpea el eje, por lo que un interruptor de inicio separado más cerca de la medio puede ser mejor Si tiene un codificador con índice, el interruptor de inicio actúa como un Por supuesto el hogar y el índice será la ubicación real de la casa.

Decidir sobre la posición de la máquina ORIGEN.

ORIGEN DE LA MÁQUINA es lo que utiliza LinuxCNC para hacer referencia a todas las coordenadas del usuario sistemas desde. Se me ocurre una pequeña razón por la que tendría que estar en cualquier lugar. Sólo hay unos pocos códigos G que pueden acceder al Sistema COORDINADO A MÁQUINA (G53, G30 y G28) Si utiliza la opción de cambio de herramienta en G30, tener el origen en la herramienta Cambiar de posición puede ser conveniente. Por convención, puede ser más fácil Para tener el ORIGEN en el interruptor de la casa.

Decidir sobre la (final) POSICIÓN DEL HOGAR.

esto solo coloca el carro en una posición consistente y conveniente después de que LinuxCNC descubre dónde está ORIGEN.

Medir / calcular las distancias de desplazamiento del eje positivo / negativo.

Mueve el eje al origen. Marca una referencia en el móvil. Deslice y el soporte no móvil (para que estén en línea) se mueva. La máquina hasta el final de los límites. Medir entre las marcas que es una. de las distancias de viaje. Mueva la mesa al otro extremo del recorrido. Medir las marcas de nuevo. Esa es la otra distancia de viaje. Si el origen está en uno de los límites entonces la distancia de viaje será cero.

(máquina) ORIGEN
El Origen es el punto cero de la MÁQUINA. (no el punto cero en el que colocó el cortador / material en). LinuxCNC usa este punto para referenciar todo lo demás desde. Debe estar dentro de los límites del software. LinuxCNC usa la ubicación del interruptor de inicio para calcular la posición de origen (cuando se utilizan interruptores de casa o debe configurarse manualmente si no se utilizan los interruptores de inicio.

Distancia de viaje
Esta es la distancia máxima que el eje puede viajar en cada direccion Esto puede o no se puede medir directamente desde el origen hasta el interruptor de límite. Lo positivo y distancias de viaje negativas deben sumar a la Distancia total de viaje.

DISTANCIA POSITIVA DE VIAJE
Esta es la distancia desde la cual viaja el Eje. El origen a la distancia de viaje positivo o El viaje total menos el viaje negativo. distancia. Lo pondrías a cero si el El origen se posiciona en el límite positivo. Siempre será cero o un número positivo.

DISTANCIA DE VIAJE NEGATIVO
Esta es la distancia desde la cual viaja el Eje. El origen a la distancia de viaje negativa. o el viaje total menos el viaje positivo distancia. Lo pondrías a cero si el El origen se posiciona en el límite negativo. Esto siempre será cero o un número negativo. Si te olvidas de hacer este PNCconf negativo. Lo haré internamente.

(Final) POSICIÓN DE CASA
Esta es la posición que tendrá la secuencia de inicio. terminar en Se hace referencia desde el origen. por lo que puede ser negativo o positivo dependiendo de En qué lado del Origen se encuentra. Cuando en la posición de inicio (final) si debes moverte en la dirección positiva para llegar al Origen, entonces el número será negativo.

UBICACIÓN DEL INTERRUPTOR DE HOGAR
Esta es la distancia desde el interruptor de la casa a el origen. Podría ser negativo o positivo. Dependiendo de qué lado del Origen es situado. Cuando en la ubicación de cambio de casa si debes moverte en la dirección positiva para llegar al Origen, entonces el número será negativo. Si configura esto a cero entonces el El origen estará en la ubicación del límite. interruptor (más la distancia para encontrar el índice si se usa)

Home Search Velocity
Curso de velocidad de búsqueda en el hogar en unidades por minuto.

Inicio Buscar Dirección
Establece la dirección de búsqueda del interruptor de inicio ya sea negativo (es decir, hacia el final de carrera negativo) o positivo (es decir, hacia un final de carrera positivo)

Inicio Latch Velocity
Fina velocidad de búsqueda de casa en unidades por minuto.

Inicio Final Velocity
Velocidad utilizada desde la posición de cierre hasta la posición inicial (final) en unidades por minuto. Se establece en 0 para la velocidad rápida máxima

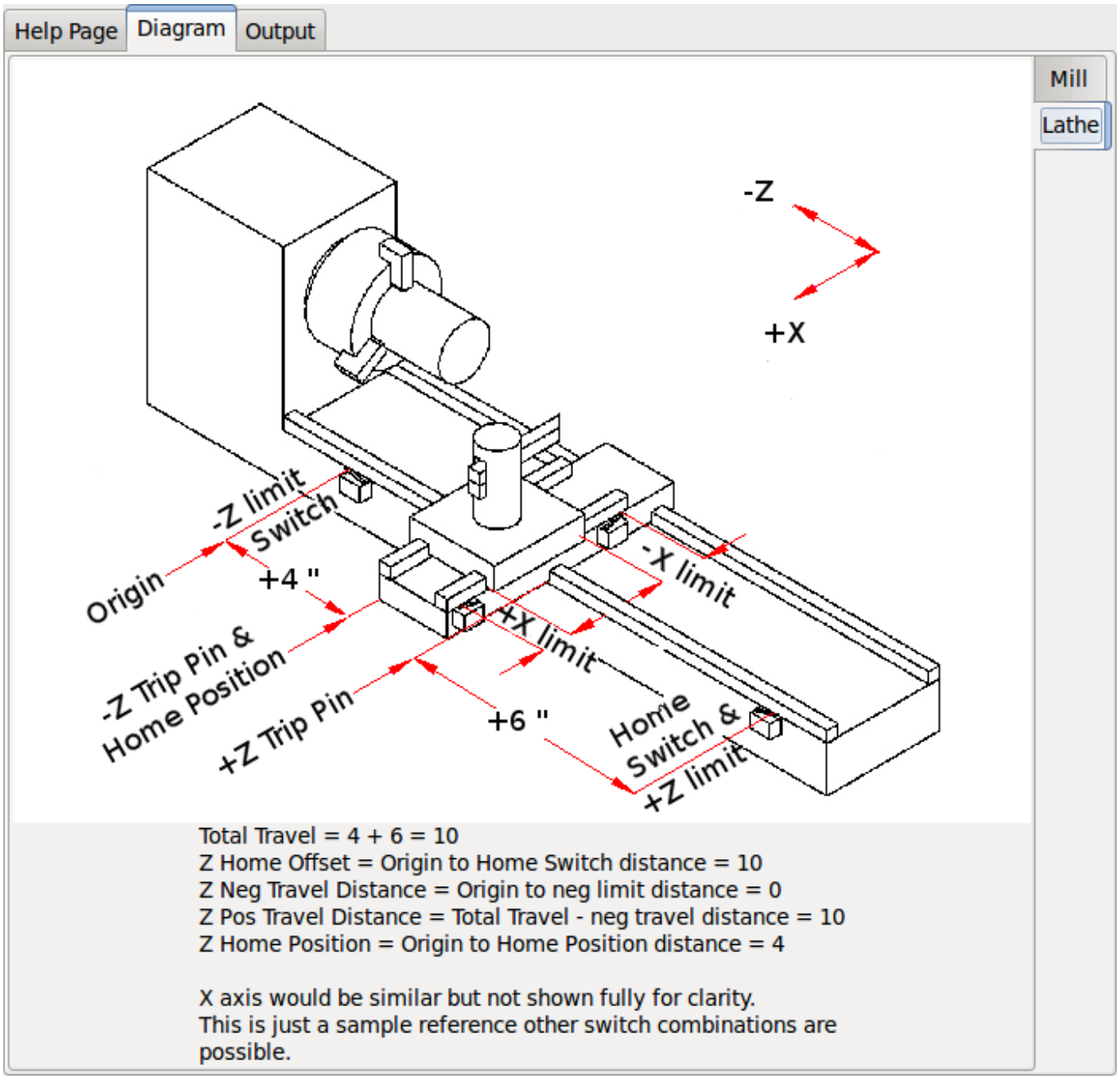
Dirección del pestillo de la casa
Permite ajustar la dirección del pestillo a la misma. o al contrario de la dirección de búsqueda.

Utilice el índice del codificador para el hogar
LinuxCNC buscará un pulso de índice de codificador mientras esté en La etapa de cierre de homing.

Utilizar archivo de compensación
Permite especificar un nombre de archivo Comp y el tipo. Permite una compensación sofisticada. Ver << sec: sección del eje, Sección AXIS >> del Capítulo INI.

Utilice la compensación de contragolpe
Permite el ajuste de la compensación de retroceso simple. Poder No se puede utilizar con el archivo de compensación. Ver << sec: sección del eje, Sección AXIS >> del Capítulo INI.

Diagrama de ayuda .AXIS



Los diagramas deberían ayudar a demostrar un ejemplo de interruptores de límite y Direcciones de movimiento del eje estándar. En este ejemplo, el eje Z era dos interruptores de límite, el interruptor positivo se comparte como un interruptor de la casa. El ORIGEN DE LA MÁQUINA (punto cero) se encuentra en el límite negativo. El borde izquierdo del carro es el pasador de disparo negativo y la derecha el pin de viaje positivo. Deseamos que la POSICIÓN FINAL DEL HOGAR esté a 4 pulgadas del ORIGEN en el lado positivo. Si el carro se moviera al límite positivo, mediríamos 10 pulgadas entre el límite negativo y el pin de disparo negativo.

3.8.10. Configuración del eje

Si selecciona señales de eje, esta página está disponible para configurar el eje. controlar.

CONSEJO: Muchas de las opciones en esta página no se mostrarán a menos que la opción correcta sea Seleccionado en páginas anteriores!

CONFIGURACIÓN DEL HUSILLO

image
images / pncconf-spindle-config.png [alt = "Configuración del eje"]

Esta página es similar a la página de configuración del motor del eje.

Hay algunas diferencias:

- A menos que uno haya elegido un eje accionado por pasos, no hay aceleración o limitación de velocidad.
- No hay soporte para cambios de engranajes o rangos.
- Si seleccionó una opción de visualización de husillo VCP, entonces la escala de husillo a velocidad y Se pueden mostrar las configuraciones del filtro.
- Spindle-at-speed permite a LinuxCNC esperar hasta que el husillo esté a la velocidad solicitada Antes de mover el eje. Esto es particularmente útil en tornos con constante Alimentación superficial y grandes cambios de diámetro de velocidad. Requiere cualquiera de los codificadores. retroalimentación o una señal digital de velocidad de giro típicamente conectada a un VFD conducir.
- Si utiliza retroalimentación de codificador, puede seleccionar un ajuste de escala de velocidad de huso que Especifica qué tan cerca debe estar la velocidad real de la velocidad solicitada para ser considerado a la velocidad.
- Si se utiliza retroalimentación de codificador, la visualización de velocidad VCP puede ser errática - la La configuración del filtro se puede utilizar para suavizar la pantalla. La escala del codificador debe ser configurado para el codificador cuenta / engranaje utilizado.
- Si está utilizando una sola entrada para un codificador de husillo, debe agregar la línea: setp hm2_7i43.0.encoder.00.counter-mode 1 (cambiando el nombre de la placa y el número de codificador a sus requisitos) en una costumbre Archivo HAL. Para más información, consulte la << sec: hm2-encoder, Sección de codificadores >> en Hostmot2 Información sobre el modo de contador.

3.8.11. Opciones avanzadas

Esto permite configurar los comandos HALUI y cargar la escalera clásica y la muestra. programas de escalera Si seleccionó las opciones de GLADE VCP, como para el eje de puesta a cero, habrá comandos que muestran. Consulte la sección << cha: hal-user-interface, HALUI Chapter >> para obtener más información sobre el uso personalizado Halcmds. Hay varias opciones de programa de escalera. El programa Estop permite que un interruptor ESTOP externo o la interfaz gráfica de usuario lancen un estop. También tiene una señal de bomba de lubricación temporizada. El Z-touch-off automático es con una placa de touch-off, el botón de apagado GLADE VCP y comandos especiales HALUI para establecer el origen del usuario actual en cero y rápido claro. El programa modbus serie es básicamente un programa de plantilla en blanco que configura Escalera clasica para modbus serie. Ver el << cha: classicladder, Classicladder Chapter >> en el manual.

OPCIONES AVANZADAS

image
images / pncconf-advanced.png [alt = "Opciones avanzadas"]

3.8.12. componentes HAL

En esta página puede agregar componentes HAL adicionales que pueda necesitar para personalizarlos. Archivos HAL. De esta manera, uno no debería tener que editar manualmente el archivo HAL principal, mientras aún está permitiendo los componentes necesarios para el usuario.

HAL COMPONENTES

image
images / pncconf-hal.png [alt = "HAL Components"]

La primera selección es componentes que pncconf utiliza internamente. Puede configurar pncconf para cargar instancias adicionales de los componentes para su archivo HAL personalizado.

Seleccione el número de instancias que necesitará su archivo personalizado, pncconf agregará Lo que necesita después de ellos.

Esto significa que si necesita 2 y pncconf necesita 1 pncconf cargará 3 instancias y utilizará el último.

Comandos de componentes personalizados
Esta selección le permitirá cargar componentes HAL que pncconf no usa. Agregue el comando loadrt o loadusr, bajo el encabezado *comando de carga* Agregue el comando addf bajo el encabezado *Comando Thread*. Los componentes se agregarán al hilo entre la lectura de entradas y la escritura. de salidas, en el orden en que se escriben en el *comando de hilo*.

3.8.13. Uso avanzado de PNCconf

PNCconf hace todo lo posible para permitir una personalización flexible por parte del usuario. PNCconf es compatible con nombres de señal personalizados, carga personalizada de componentes, Archivos HAL personalizados y firmware personalizado.

También hay nombres de señales que PNCconf siempre proporciona independientemente de las opciones seleccionado, para archivos HAL personalizados del usuario Pensando que la mayoría de las personalizaciones deberían funcionar independientemente de si posteriormente seleccionas Diferentes opciones en PNCConf.

Eventualmente, si las personalizaciones están fuera del alcance del marco de trabajo de PNCconf puede usar PNCconf para construir una configuración base o usar uno de los ejemplos de LinuxCNC Configuraciones y solo edita a mano lo que quieras.

Nombres de señales personalizadas
Si desea conectar un componente a algo en un archivo HAL personalizado, escriba un Nombre de la señal única en el cuadro de entrada de combo. Ciertos componentes agregarán terminaciones a su nombre de señal personalizado:

Los codificadores agregarán <nombre personalizado> +:

- posición
- cuenta
- velocidad
- habilitar índice
- Reiniciar

Steppers añadir:

- habilitar
- cuenta
- posicion-cmd
- posición-fb
- velocidad-fb

PWM añadir:

- habilitar
- valor

Los pines GPIO solo tendrán conectado el nombre de la señal ingresada

De esta manera, uno puede conectarse a estas señales en los archivos HAL personalizados y aún así Tienes la opción de moverlos más tarde.

Nombres de señales personalizadas
La página de Componentes de Hal se puede usar para cargar los componentes que necesita un usuario para personalizacion

Cargando Custom Firmware
PNCconf busca el firmware en el sistema y luego busca el archivo XML que Se puede convertir a lo que entiende. Estos archivos XML solo se suministran para Firmware lanzado oficialmente por el equipo de LinuxCNC. Para utilizar un firmware personalizado debe convertirlo en una matriz que PNCconf entienda y agregar su ruta de archivo al archivo de preferencias de PNCconf. Por defecto esta ruta busca en el escritorio una carpeta llamada custom_firmware y un archivo llamado firmware.py.

El archivo de preferencias oculto está en el archivo de inicio del usuario, es llamado .pncconf-preferences y requiere uno para seleccionar *mostrar archivos ocultos* para ver y editarlo. El contenido de este archivo se puede ver cuando carga por primera vez PNCconf - presione el botón de ayuda y mire la página de salida.

Pregunte en la lista de correo o foro de LinuxCNC para obtener información sobre la conversión de firmware personalizado. No todo el firmware se puede utilizar con PNCconf.

Archivos HAL personalizados
Hay cuatro archivos personalizados que puede usar para agregar comandos HAL a:

- custom.hal es para comandos HAL que no tienen que ejecutarse después de la interfaz GUI cargas Se ejecuta después del archivo de configuración HAL.
- custom_postgui.hal es para comandos que deben ejecutarse después de que se cargue AXIS o Standalone PYVCP muestra cargas.
- custom_gvcp.hal es para comandos que deben ejecutarse después de cargar glade VCP.
- shutdown.hal es para que los comandos se ejecuten cuando LinuxCNC se apaga de manera controlada.

3.9. Linux FAQ

Estos son algunos comandos básicos y técnicas para nuevos usuarios de linux. Información más completa se puede encontrar en la web o mediante las páginas del manual con el comando man.

3.9.1. Login automatico

Al instalar linuxCNC con el CD de Ubuntu por defecto se tiene que iniciar sesión cada vez que encienda el ordenador. Para activar autentificacion automática vaya a *System* > *Administration* > *Login Window*. Si se trata de una instalación nueva, la ventana de inicio de sesión puede tardar unos segundo en aparecer. Usted tiene que tener la contraseña que utilizo para la instalación para acceder a la ventana de configuracion de inicio. En la pestaña seguridad marque habilitar acceso automático y elija un nombre de usuario de la lista (seleecione su nombre de usuario).

N.T. Debian Stretch usa por defecto el entorno de escritorio Xfce, con el gestor de pantallas lightDM. Para obtener acceso automatico con Stretch:

- En un terminal, use el comando:

```
$ /usr/sbin/lightdm --show-config
```

- Anote el path absoluto del archivo de configuracion lightdm.conf.
- Edite ese archivo con un editor de texto puro (gedit, nano, etc), como root.
- Busque y descomente las lineas:

```
#autologin-user=
#autologin-user-timeout=0
```

- Haga autologin-user=su_nombre_usuario
- Guarde y reinicie.

3.9.2. Inicio automatico de LinuxCNC

Para tener un inicio automático de linuxCNC con su configuración después de encender el equipo vaya a *System* > *Preferences* > *Sessions* > *Startup Applications*, seleccione la opcion agregar nuevo. Vaya a su carpeta de configuración y seleccione el archivo .ini. Cuando el cuadro de dialogo selector de archivos se cierra, añadir linuxcnc y un espacio al frente de la ruta a su archivo .ini.

Ejemplo:

```
linuxcnc /home/mill/linuxcnc/config/mill/mill.ini
```

N.T. En Debian Stretch, vaya a *Aplicaciones* > *Configuracion* > *Administrador de Configuracion* > *Sesion e Inicio*. En la pestaña *Autoarranque de aplicaciones*, use el boton *Añadir*. De un nombre, una descripcion y una orden similar al ejemplo anterior. En el proximo reinicio, LinuxCNC arrancara automaticamente.

3.9.3. Terminal

Hay que hacer muchas cosas desde la terminal, como verificar el búfer de mensajes del núcleo con *dmesg*. Ubuntu y Linux Mint tienen un atajo de teclado Ctrl + Alt + t. N.T. Debia Stretch no tiene definido ningun atajo de teclado. Se puede crear facilmente con el *Administrador de Configuracion*.

La mayoría de los administradores de archivos modernos admiten el boton derecho para abrir un terminal; solo asegúrese de hacer clic derecho en un área en blanco o en un directorio.

La mayoría de los sistemas operativos tienen el terminal como elemento de menú, generalmente en *Accesorios*.

3.9.4. Paginas de manual

Las páginas del manual son generadas automáticamente en la mayoría de los casos. Las páginas del manual estan generalmente disponibles para la mayoría de los programas y comandos de linux.

Para abrir una página del manual utilice una terminal. La terminal se puede abrir usando la ruta *Applications* > *Accessories* > *Terminal*. Por ejemplo si quiere encontrar algo acerca del comando de busqueda "find" tecle:

```
man find
```

Utilice las teclas *página-arriba y 'pagina-abajo* para ver las páginas del manual y la tecla Q para salir de la visualizacion.

3.9.5. Lista de modulos

A veces, para soluciónar algun problema, usted necesita obtener una lista de los módulos que se encuentran cargados. En una ventana de terminal teclee:

```
lsmod
```

Si desea enviar la salida de lsmod a un archivo de texto teclee:

```
lsmod > mymod.txt
```

El archivo de texto resultante, *mymod.txt*, se colocara en el directorio home si usted no cambio de directorio cuando abrio la terminal y será nombrado mymod.txt, o como usted lo haya nombrado.

3.9.6. Edicion de archivos de root

Al abrir el explorador de archivos y ver que el propietario del archivo es el usuario root, se tienen que hacer algunos pasos adicionales para modificar ese archivo. La edición de algunos archivos puede tener malos resultados. Tenga cuidado al editar los archivos de root; generalmente usted puede ver y abrir los archivos root en modo de *solo lectura*.

3.9.6.1. Con la linea de comandos

En una ventana de terminal teclee.

```
sudo gedit
```

Abrir el archivo con el menu File > Open > Edit, y proceda a editar.

3.9.6.2. Usando la interface grafica

```
.Haga clic derecho sobre el escritorio y seleccione 'Crear lanzador'
.Escriba un nombre como 'editar sudo'
.Escriba 'gksudo "gnome-open %u"' como el comando y guarde el lanzador
 en su escritorio.
.Arrastré el archivo a su lanzador para abrir y editar.
```

3.9.6.3. Acceso tipo super usuario

En Ubuntu (o Debian) puede convertirse en super usuario tecleando "sudo -i" en una terminal. Debera escribir su contraseña. Tenga cuidado porque usted puede dañar su instalacion si no sabe lo que esta haciendo.

3.9.7. Comandos en la terminal

3.9.7.1. Directorio de trabajo

Para encontrar la ruta del directorio actual de trabajo en la terminal, teclee:

```
pwd
```

3.9.7.2. Cambiar de directorios

Para subir un nivel en la terminal teclee:

```
cd ..
```

Para subir dos niveles en la terminal teclee:

```
cd ../..
```

Para desplazarse hacia abajo hacia el subdirectorio linuxcnc/configs en la terminal teclee:

```
cd linuxcnc/configs
```

3.9.7.3. Listado de los archivos en un directorio

Para ver una lista de todos los archivos y subdirecciones en la terminal teclee:

```
dir
```

ó

```
ls
```

3.9.7.4. Encontrar un archivo

El comando *find* puede ser un poco confuso para un usuario nuevo de linux. La sintaxis básica es la siguiente:

```
find directorio-inicio parametros acciones
```

Por ejemplo para encontrar todos los archivos .ini en el directorio de linuxcnc primero tiene que usar el comando pwd para ver el directorio. + abra una ventana de terminal y escriba.

```
pwd
```

y pwd podría devolver el siguiente resultado:

```
/home/joe
```

Con esta información se pondrá el comando conjunto de esta manera:

```
find /home/joe/linuxcnc -name \*.ini -print
```

Aqui, -name es el nombre del archivo que se busca y -print hace que se muestre el resultado en la ventana de terminal. El nombre *.ini indica *devolver todos los archivos que tienen la extensión .ini*. La diagonal se requiere para escapar los metacaracteres de la consola. Si desea mas informacion al respecto, vea las paginas man de *find*.

3.9.7.5. Busqueda de texto

```
grep -irl 'buscar' *
```

Este comando encuentra todos los archivos que contienen el texto *buscar* en el directorio actual y todos los subdirectorios por debajo de este, sin tener en cuenta el uso de mayusculas. La -i es para ignorar mayusculas, la -r es recursivo (incluir todos los subdirectorios en la búsqueda) y la opcion -l retornara una lista de los nombres de archivo. Si no se usa -l tambien se obtendra el texto donde fue encontrada la ocurrencia de lo buscado dentro de *buscar*. El * es un comodín para buscar todos los archivos.

3.9.7.6. Mensaje de arranque

Para ver los mensajes de arranque usar "dmesg" en la ventana de comandos. Para guardar los mensajes de arranque en un archivo use el operador de redirección, de esta manera:

```
dmesg > bootmsg.txt
```

El contenido de este archivo puede ser copiado y pegado en línea para compartir con la gente que le este intentando ayudar a diagnosticar un problema.

Para borrar el buffer de mensajes, teclee:

```
sudo dmesg -c
```

Esto puede ser útil justo antes del arranque de LinuxCNC, por lo que solo habra un registro de información relacionada con el lanzamiento actual de LinuxCNC.

Para encontrar la direccion de un puerto paralelo integrado use grep para filtrar la información producida por dmesg.

Después del arranque abrir una terminal y escribir:

```
dmesg|grep parport
```

3.9.8. Items convenientes

3.9.8.1. Iniciador de terminal

Si quiere añadir un iniciador de terminal en la barra del panel en la parte superior de la pantalla normalmente puede hacer clic derecho en el panel en la parte superior de la pantalla y seleccionar "añadir al panel". Seleccione lanzador de aplicación personalizado y agregar. Dele un nombre y use el comando gnome-terminal en la caja de comando.

3.9.9. Problemas de Hardware

3.9.9.1. Informacion Hardware

Para encontrar que hardware está conectado a la placa base, en una ventana de terminal teclee.

```
lspci -v
```

3.9.9.2. Resolucion del monitor

Durante la instalación, ubuntu intentara detectar la configuración del monitor. Si esto no funciona el sistema se instalara con un máximo de resolución 800x600.

Instrucciones para arreglar esto, se encuentra aquí:

https://help.ubuntu.com/community/FixVideoResolutionHowto

3.9.10. Paths

Paths relativos Los phats relativos se basan en el directorio de arranque que contiene el archivo ini. Usar paths relativos puede facilitar la relocalizacion de archivos de configuracion pero requiere una buena comprensión de los especificadores de path linux.

```
./f0        es lo mismo que f0, e.g., un archivo llamado f0 en el directorio
../f1       se refiere a un archivo llamado f1 en el directorio padre
../../f2    se refiere a un archivo llamado f2 en el directorio padre del padre
../../../f3 etc.
```

3.10. Informacion para Usuarios de Torno

3.10.1. Modo Torno

Si su máquina CNC es un torno, hay algunos cambios específicos que probablemente querrá hacer a su archivo ini para obtener los mejores resultados de LinuxCNC.

Si está utilizando la gui AXIS, haga que Axis muestre sus herramientas de torno correctamente. Consulte la sección configuración INI para obtener más detalles.

Para configurar AXIS para el modo torno:

```
[DISPLAY]

# Informar a Axis que nuestra máquina es un torno.
LATHE = TRUE
```

El modo torno en Axis no establece su plano predeterminado en G18 (XZ). Debe programarlo en el preámbulo de cada archivo gcode o (mejor) agrégar a su archivo ini lo siguiente:

```
[RS274NGC]

# códigos g modales iniciales del intérprete
RS274NGC_STARTUP_CODE = G18 G20 G90
```

Si está usando Gmoccapy entonces vea la sección Torno de Gmoccapy.

3.10.2. Tabla de herramientas de torno

La "Tabla de herramientas" es un archivo de texto que contiene información sobre cada herramienta. El archivo se encuentra en el mismo directorio que su configuración y de forma predeterminada se denomina "tool.tbl". Las herramientas pueden estar en un cambiador de herramientas o ser cambiadas manualmente. El archivo puede editarse con un editor de texto o actualizarse con G10 L1, L10, L11. También hay un editor de tablas de herramientas incorporado en la gui Axis. El número máximo de entradas en la tabla de herramientas es 56. El número máximo de herramienta y de ranura es 99999.

Las versiones anteriores de LinuxCNC tenían dos formatos de tablas de herramientas diferentes para fresadoras y tornos, pero desde la versión 2.4.x, se utiliza un formato de tabla de herramientas para todas las máquinas. Simplemente ignore las partes de la tabla de herramientas que no pertenecen a su máquina,o que no necesite usar. Para obtener más información sobre los detalles del formato de tabla de herramientas, vea la sección Tabla de Herramientas.

3.10.3. Orientacion de la herramienta de torno

La siguiente figura muestra las orientaciones de la herramienta de torno con el ángulo de la línea central (l.c.) de cada orientación e información sobre FRONTANGLE y BACKANGLE. FRONTANGLE y BACKANGLE están en el sentido de las agujas del reloj comenzando en una línea paralela a Z+.

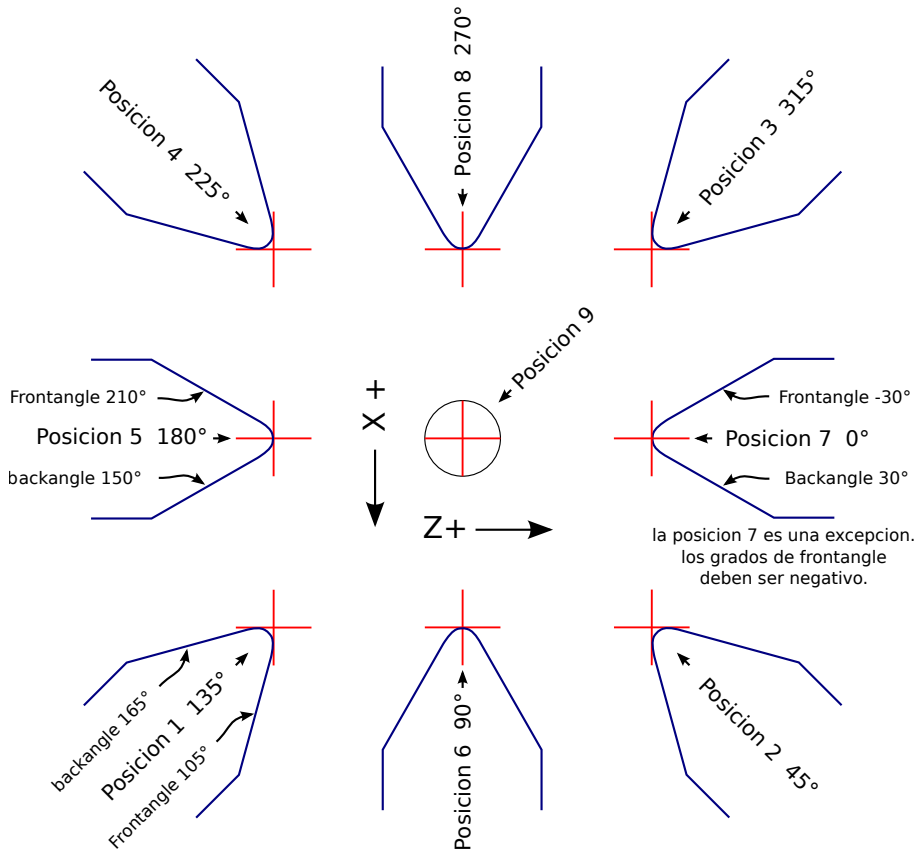
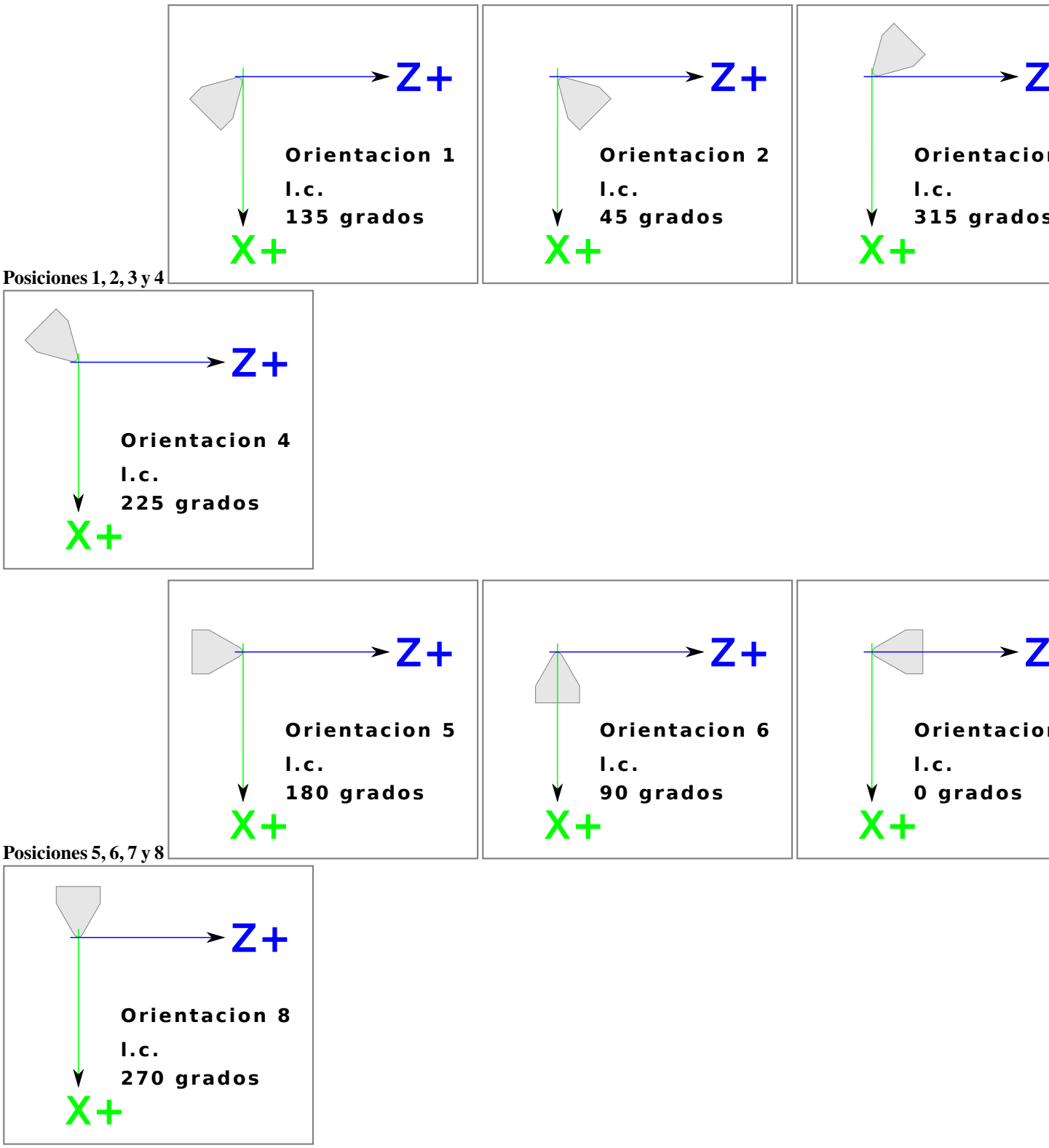


Figura 3.26: Orientaciones de la herramienta del torno

En AXIS, la siguientes figura muestra cómo se ven las posiciones de las herramientas, tal como se ingresaron en la tabla de herramientas.



Posiciones 1, 2, 3 y 4

Posiciones 5, 6, 7 y 8

3.10.4. Tool Touch Off

Cuando AXIS se ejecuta en modo de torno, puede configurar X y Z en la tabla de herramientas mediante la ventana Touch Off. Si tiene una torreta de herramientas normalmente tendra seleccionado *Touch off to fixture* al configurar su torreta. Cuando configura

Z cero del material tendra seleccionado *Touch off to material*. Para obtener más información sobre los códigos G utilizados para las herramientas, consulte M6, Tn, y G43. Para obtener más información sobre las opciones touch off de la herramienta en Axis, consulte Tool Touch Off.

3.10.4.1. X Touch Off

El offset del eje X para cada herramienta es normalmente un desplazamiento desde la línea central del husillo.

Un método es tomar su herramienta de torneado normal y preparar algunas piezas a un diámetro conocido. Usando la ventana Touch Off, ingrese el diámetro medido (o radio si está en modo radio) para esa herramienta. Luego, use liquido de marcado o un rotulador para cubrir la pieza. Lleve cada herramienta hasta que toque solo el tinte y establezca su offset X con touch off en el diámetro de la pieza utilizada. Asegúrese de que las herramientas en los cuadrantes de las esquinas tengan el radio de la punta establecido correctamente en la tabla de herramientas para que el punto de control sea correcto. Tool touch off automaticamente agrega un G43 por lo que la herramienta actual es el offset actual.

Una sesión típica podría ser:

1. homing de cada eje si no lo están.
2. Configurar la herramienta actual con *Tn M6 G43*, donde *n* es el número de la herramienta.
3. Seleccionar el eje X en la ventana de Control Manual.
4. Mover la X a una posición conocida o realizar un corte de prueba y medir el diámetro.
5. Seleccionar Touch Off y seleccionar Tabla de herramientas, luego ingrese la posición o el diámetro.
6. Siga la misma secuencia para corregir el eje Z.

Nota: si está en el modo Radio, debe ingresar el radio, no el diámetro.

3.10.4.2. Z Touch Off

Los offsets del eje Z pueden ser un poco confusos al principio porque hay dos elementos para el desplazamiento Z. Existe el desplazamiento de la tabla de herramientas y el desplazamiento de las coordenadas de la máquina. Primero veremos los offsets de la tabla de herramientas. Un método es usar un punto fijo en su torno y establecer el desplazamiento Z para todas las herramientas desde este punto. Algunos usan la nariz del husillo o la cara del mandril. Esto le da la posibilidad de cambiar a una nueva herramienta y establece su offset Z sin tener que reiniciar todas las herramientas.

Una sesión típica podría ser:

1. homing de cada eje si no lo están.
2. Asegurar de que no haya offsets en vigor para el sistema de coordenadas actual.
3. Configurar la herramienta actual con *Tn M6 G43*, donde *n* es el número de la herramienta.
4. Seleccionar el eje Z en la ventana de Control manual.
5. Acerque la herramienta a la superficie de control. Usando un cilindro mueva el Z lejos de la superficie de control hasta que el cilindro pase justo entre la herramienta y la superficie de control.
6. Seleccione Touch Off y seleccione la Tabla de herramientas y establezca la posición en 0.0.
7. Repita para cada herramienta usando el mismo cilindro.

Ahora todas las herramientas están desplazadas a la misma distancia de una posición estándar. Si cambia una herramienta como una broca, repita lo anterior y ahora estara sincronizado con el resto de las herramientas para el desplazamiento Z. Algunas herramientas pueden requerir un poco mas para determinar el punto de control desde el punto touch off. Por ejemplo, si tiene una herramienta de tronzado de 0.125" de ancho y toca con el lado izquierdo, pero quiere que el derecho sea el Z0, ingrese 0.125" en la ventana touch off.

3.10.4.3. El offset Z de la maquina

Una vez que todas las herramientas tienen el offset Z ingresado en la tabla de herramientas, puede utilizar cualquier herramienta para configurar el offset de la máquina utilizando el sistema de coordenadas de máquina.

Una sesión típica podría ser:

1. homing de cada eje si no lo estan.
2. Configurar la herramienta actual con "Tn M6", donde "n" es el número de la herramienta.
3. Emitir un G43 de modo que el offset de la herramienta actual esté vigente.
4. Lleve la herramienta a la pieza de trabajo y configure el offset Z de la máquina.

Si olvida configurar el G43 para la herramienta actual al configurar el sistema de coordenadas de la máquina compensado, no obtendrá lo que espera, ya que el offset de la herramienta se agregará al offset actual cuando la herramienta se utilize en su programa.

3.10.5. Movimiento sincronizado del husillo

El movimiento sincronizado del husillo requiere un codificador de cuadratura conectado al husillo con un pulso índice por revolución. Ver la página man de motion y el Ejemplo de Control de Husillo para más información.

Roscado El ciclo de roscado G76 se utiliza para hilos internos y externos. Para obtener más información, consulte la sección G76.

Velocidad constante de superficie CSS (Constant Surface Speed) utiliza el origen de la máquina X modificado por el offset X de la herramienta para calcular la velocidad del husillo en RPM. CSS hará un seguimiento de los cambios en los offsets de herramienta. El origen maquina X debe ser donde la herramienta de referencia (la que tiene cero desplazamiento) está en el centro de rotación. Para obtener más información, consulte la sección G96.

Alimentacion por revolución La alimentación por revolución moverá el eje Z en la cantidad de F por cada revolución. Esto no es para roscar; use G76 para para roscado. Para obtener más información, consulte la sección G95.

3.10.6. Arcos

Calcular arcos puede ser desafiante si no se considera el modo de radio y diámetro en tornos y la orientacion del sistema de coordenadas de la máquina. Lo siguiente se aplica a los arcos de formato centro. En un torno, debe incluir G18 en su preámbulo ya que el valor predeterminado es G17, incluso si está en modo torno, en la interfaz de usuario Axis. Los arcos en el plano G18 XZ utilizan los desplazamientos I (eje X) y K (eje Z).

3.10.6.1. Diseno de arcos en torno

Un torno típico tiene el husillo a la izquierda del operador y la herramienta en el lado del operador desde línea central del husillo. Esto normalmente se configura con el eje Y imaginario (+) apuntando al suelo.

Lo siguiente será cierto en este tipo de configuración:

- El eje Z(+) apunta hacia la derecha, alejandose del husillo.
- El eje X(+) apunta hacia el operador, y cuando está en el lado del operador, los valores de X son positivos.

Algunos tornos con herramientas en la parte posterior tienen el eje Y imaginario (+) apuntando hacia arriba.

Las direcciones de arco G2/G3 se basan en el eje alrededor del cual giran. En el caso de los tornos, se trata del eje Y imaginario. Si el eje Y (+) señala hacia abajo, hay que mirar hacia arriba para que aparezca el arco que va en la dirección correcta. Así que mirando desde (no hacia) arriba, invierta G2/G3 para que el arco aparezca en la dirección correcta.

3.10.6.2. Modo de radio y diametro

Al calcular arcos en modo radio solo tiene que recordar la dirección de rotación que se aplica a su torno.

Al calcular arcos en el modo de diámetro, X es el diámetro y el offset X (I) es el radio incluso si está en el modo de diámetro G7.

3.10.7. Ruta de la herramienta

3.10.7.1. Punto de control

El punto de control de la herramienta sigue el camino programado. El punto de control está en la intersección de las líneas paralelas a X y Z, tangentes al diámetro de la punta de la herramienta, tal como define los ejes X y Z un touch off para esa herramienta. Al tornear o refrentar piezas con caras rectas, el camino de corte y el filo de la herramienta siguen el mismo recorrido. Al tornear radios y ángulos, el filo de la herramienta no seguirá la ruta programada a menos que esté activada la *compensacion del cortador*. En la siguientes figuras se puede ver cómo el punto de control no sigue el filo de la herramienta, como se podría suponer.

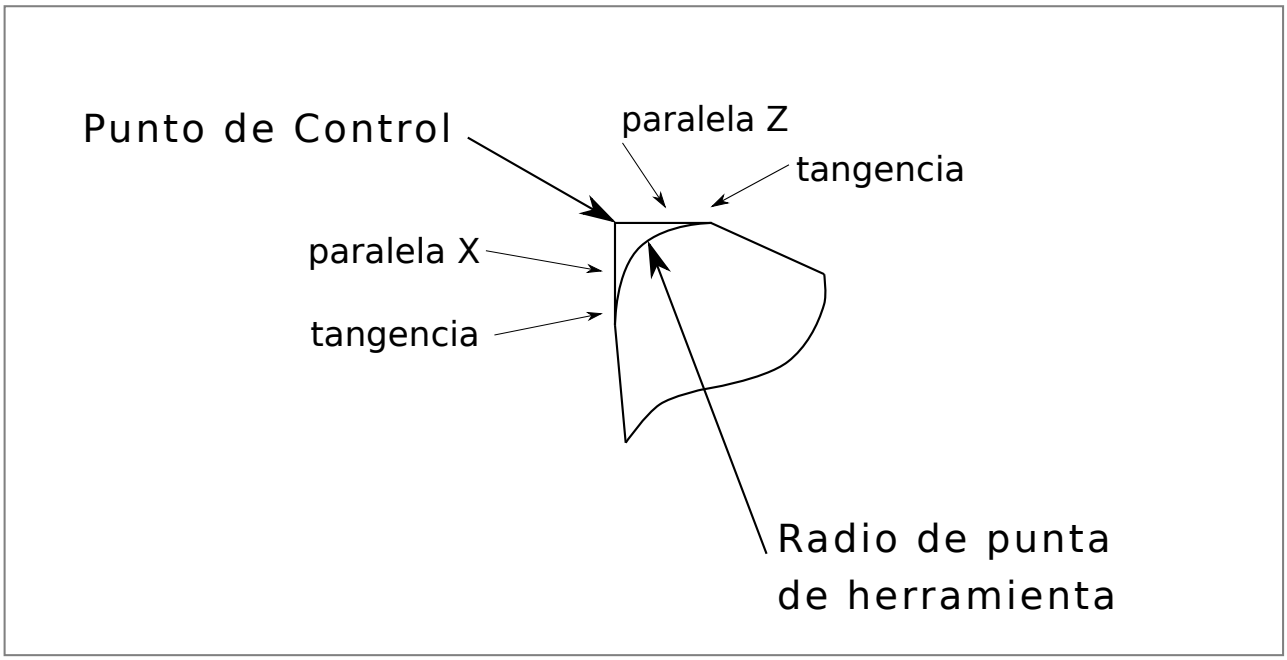


Figura 3.27: Punto de control

3.10.7.2. Corte de angulos sin Compensacion

Ahora imagine que programamos una rampa sin compensacion. El camino programado se muestra en la siguiente figura. Como se puede ver, la ruta programada y la ruta de corte deseada son una y la misma, siempre que nos estemos moviendo en la dirección X o Z solamente.

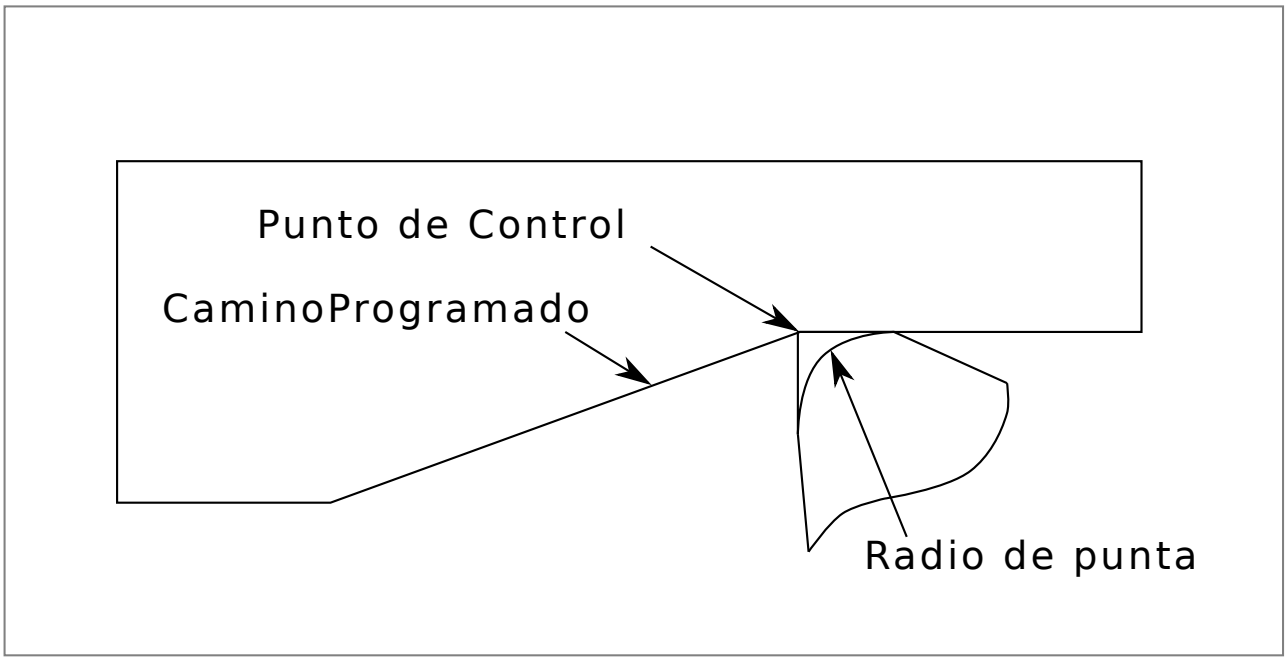


Figura 3.28: Rampa de entrada

Ahora, a medida que el punto de control avanza a lo largo del camino programado, el borde del cortador real no sigue ese camino como se muestra en la siguiente figura. Hay dos maneras de resolver esto; compensacion del cortador o ajustar la ruta programada para compensar el radio de la punta.

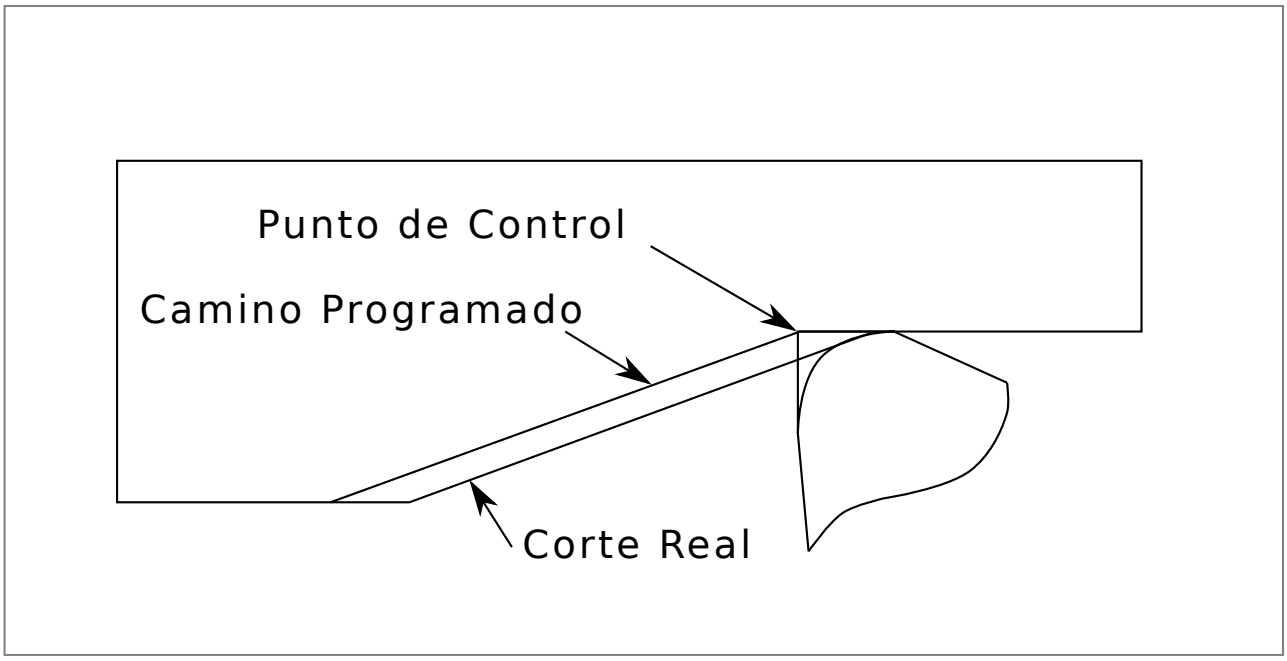


Figura 3.29: Rampa de ruta

El ejemplo anterior se podria ajustar la ruta programada a la ruta real deseada moviendo la ruta programada hacia la izquierda del radio de la punta de la herramienta.

3.10.7.3. Cortando un radio

En este ejemplo examinaremos lo que sucede durante el corte de un radio. sin compensacion. En la siguiente figura se ve la herramienta torneando el diametro exterior de la pieza. El punto de control de la herramienta sigue el camino programado y la herramienta toca el diametro.

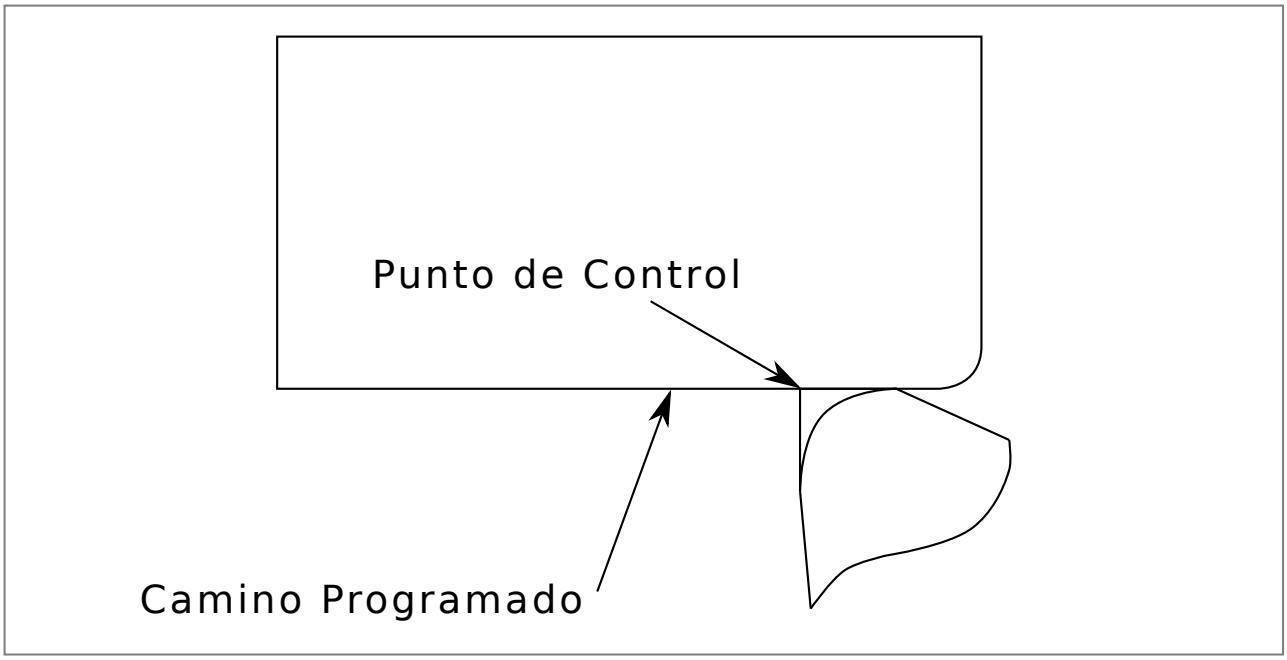


Figura 3.30: Torneado exterior

En la siguiente figura puede ver que a medida que la herramienta se acerca al final de la pieza, el punto de control todavía sigue el camino pero la punta de la herramienta ha dejado la pieza y está al aire. También puede ver que aunque ha sido programado un radio, la pieza terminará con una esquina cuadrada.

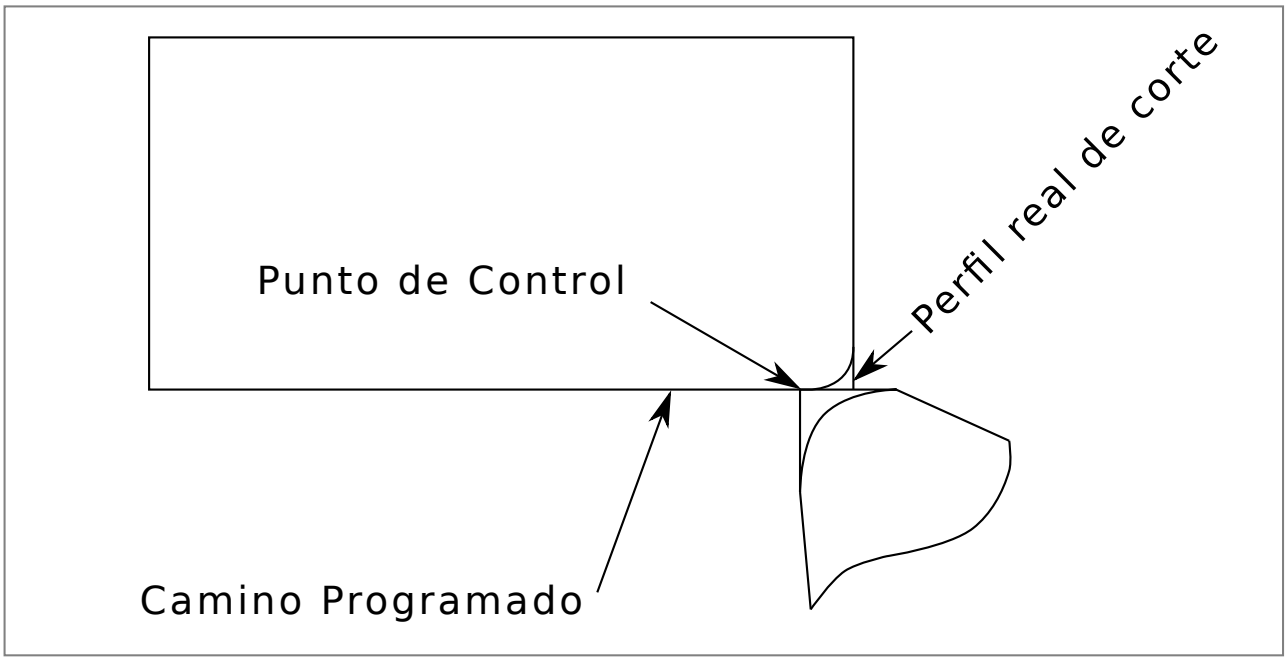


Figura 3.31: Corte del radio

Ahora puede ver como el punto de control sigue el radio programado La punta de la herramienta ha dejado la pieza y ahora está al aire.

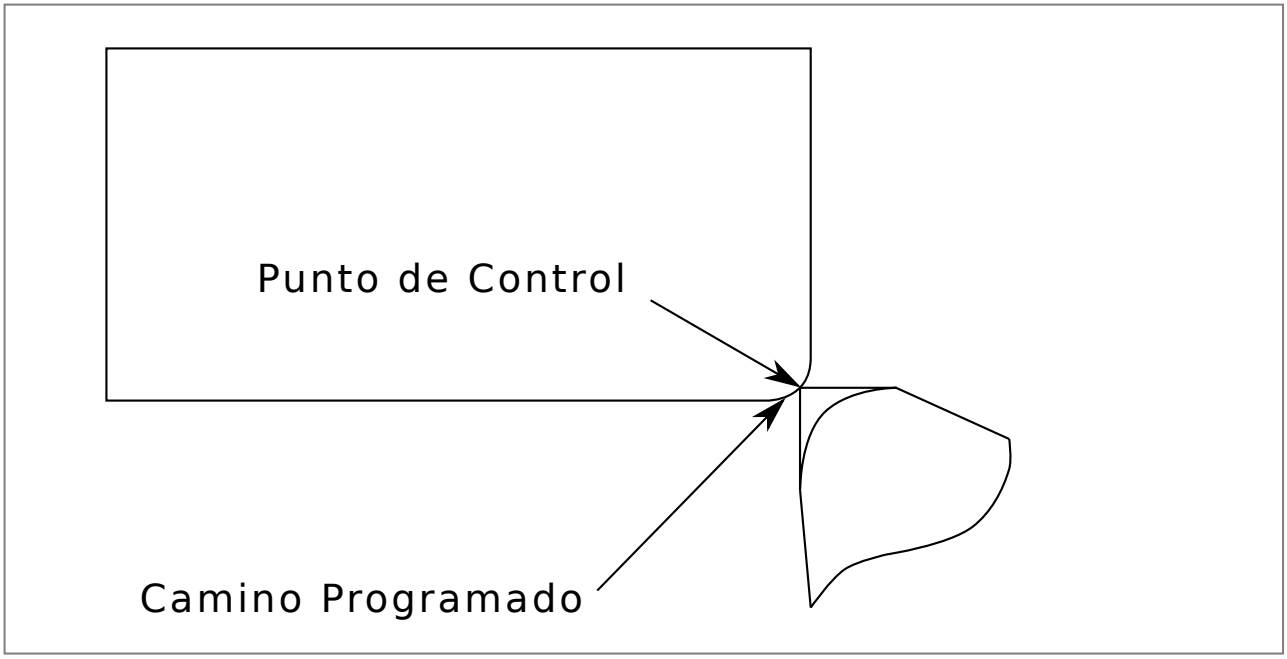


Figura 3.32: Corte del radio

En la figura final podemos ver que la punta de la herramienta terminará de cortar la cara pero deja una esquina cuadrada en lugar de un radio. Notese también que si se programa el corte para que finalice en el centro de la pieza, una pequeña cantidad de material no se cortara debido al radio de la herramienta. Para terminar una cara cortada en el centro de una parte, tiene que programar la herramienta para ir más allá del centro; como minimo, el radio de la punta de la herramienta.

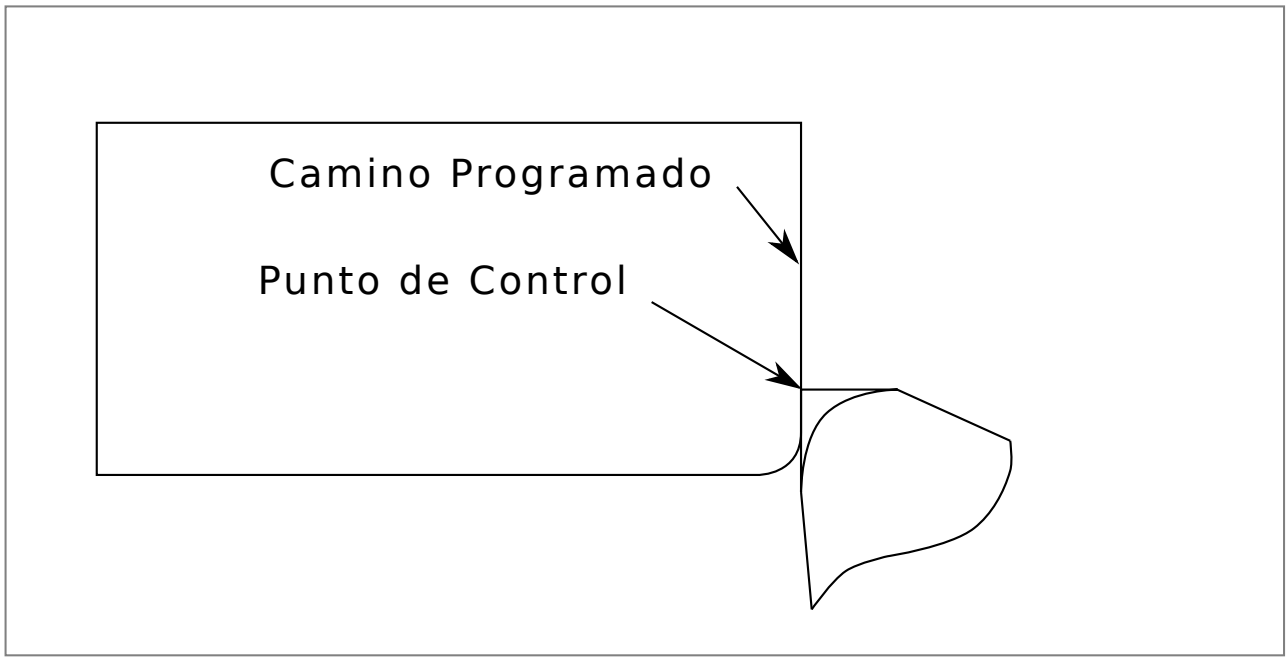


Figura 3.33: Corte de la cara

3.10.7.4. Usando Compensacion del Cortador

Cuando utilice una herramienta de corte en un torno, piense en el radio de la punta de la herramienta como el radio de un cortador cilindrico (fresa). Cuando se usa una herramienta de corte, el camino debe ser suficiente grande para una herramienta cilindrica que no interfiriera en la línea siguiente. Cuando se cortan líneas rectas en el torno es posible que no desee utilizar compensacion del cortador. Por ejemplo, mandrinar un agujero con una barra demasiado ajustada puede que no le permita suficiente espacio para realizar el movimiento de salida. El movimiento de entrada en un arco con compensacion es importante para obtener los resultados correctos.

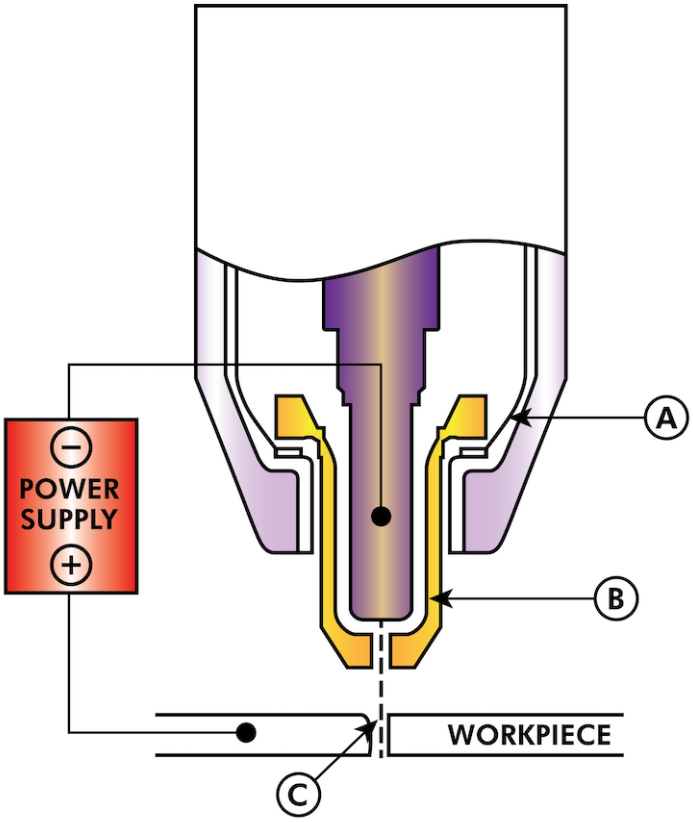
3.11. Basicos del Corte por Plasma para Usuarios de LinuxCNC

3.11.1. ¿Qué es el plasma?

El plasma es un cuarto estado de la materia, un gas que se ha calentado a una temperatura extremadamente alta e ionizado para que se vuelva eléctricamente conductor. Los procesos de corte y acanalado por arco de plasma utilizan este plasma para transferir un arco eléctrico a la pieza de trabajo. El metal a cortar o quitar se derrite por el calor del arco y se sopla. Si bien el objetivo del corte con arco de plasma es la separación del material, el acanalado con arco de plasma se utiliza para eliminar metales a una profundidad y ancho controlados.

Las antorchas de plasma son similares en diseño a las bujías de automocion. Consisten en secciones negativas y positivas separadas por un aislante central. Dentro de la antorcha, el arco piloto comienza en el espacio entre el electrodo cargado negativamente y la punta cargada positivamente. Una vez que el arco piloto ha ionizado el gas, la columna de plasma sobrecalentada fluye a través del pequeño orificio en la punta de la antorcha, que se enfoca en el metal a cortar.

En una antorcha de corte por plasma, un gas frío ingresa a la Zona B, donde un arco piloto entre el electrodo y la punta de la antorcha calienta e ioniza el gas. El arco de corte principal se transfiere luego a la pieza de trabajo a través de la columna de gas de plasma en la Zona C. Al forzar el gas de plasma y el arco eléctrico a través de un pequeño orificio, la antorcha suministra una alta concentración de calor a un área pequeña. El arco de plasma rígido y restringido se muestra en la Zona C. Se usa polaridad directa de corriente continua (CC) para el corte por plasma, como se muestra en la ilustración. La zona A canaliza un gas secundario que enfría la antorcha. Este gas también ayuda al gas de plasma de alta velocidad a expulsar el metal fundido del corte permitiendo un corte rápido y sin escoria.



TYPICAL TORCH HEAD DETAIL

3.11.2. Inicialización de arco

Hay dos métodos principales para la inicialización del arco para cortadores de plasma que están diseñados para funcionar con CNC. Si bien se utilizan otros métodos en algunas máquinas (como el inicio por contacto, donde se requiere el contacto físico de la antorcha con el material), no son adecuados para aplicaciones CNC.

3.11.2.1. Inicio por alta frecuencia

Este tipo de inicio es ampliamente empleado. Aunque es una tecnología más antigua, funciona bien y se inicia rápidamente. Pero, debido a la alta frecuencia y al alto voltaje que se requiere para ionizar el aire, tiene algunos inconvenientes. A menudo interfiere con los circuitos electrónicos circundantes e incluso puede dañar los componentes. También se necesita un circuito especial para crear el arco piloto. Los modelos de bajo costo no tendrán un arco piloto, y requieren tocar el consumible para comenzar el trabajo. El empleo de un circuito de HF también puede aumentar los problemas de mantenimiento, ya que generalmente hay puntos ajustables que deben limpiarse y reajustarse de vez en cuando.

3.11.2.2. Inicio por Blowback

Este tipo de arranque utiliza la presión de aire suministrada a la cortadora para forzar un pequeño pistón o cartucho dentro del cabezal de la antorcha hacia atrás para crear una pequeña chispa entre la superficie interior del consumible, ionizando el aire y creando una pequeña llama de plasma. Esto también crea un "arco piloto" que proporciona una llama de plasma que permanece encendida, tanto si está en contacto con el metal como si no. Este es un muy buen tipo de inicio que ahora utilizan varios fabricantes. Su ventaja es que requiere algo menos de circuiteria, es bastante confiable y genera mucho menos ruido eléctrico.

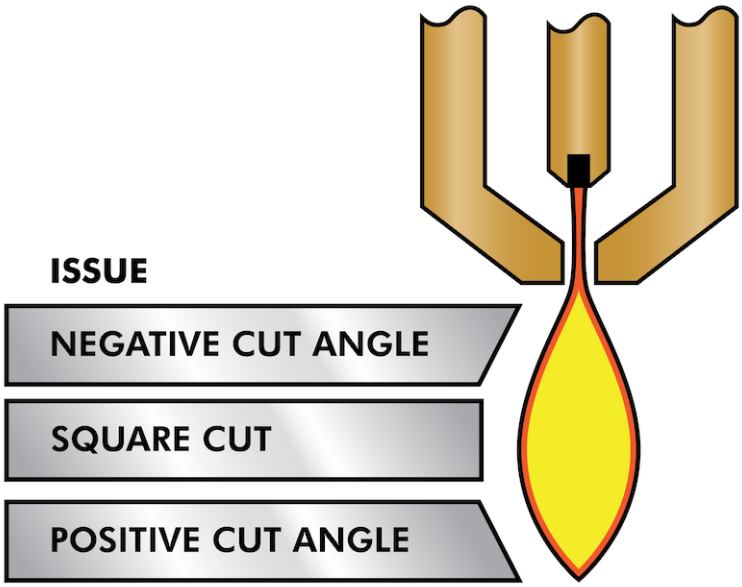
Para los sistemas CNC de plasma de aire de nivel basico, se prefiere el estilo blowback para minimizar la interferencia eléctrica con la electrónica y las PC estándar, pero el arranque de alta frecuencia sigue siendo el más importante en máquinas grandes de 200 amperios o más. Estos requieren PC's y dispositivos electrónicos de nivel industrial e incluso los fabricantes comerciales han tenido problemas con fallos porque no han tenido en cuenta el ruido eléctrico en sus diseños.

3.11.3. Plasma y CNC

Las operaciones de plasma en máquinas CNC son bastante especiales en comparación con el fresado o el torneado y es un proceso un tanto huérfano. El calentamiento desigual del material por el arco de plasma hará que la plancha se pandee y se doble. La mayoría de las planchas de metal no salen de fabrica en un estado muy uniforme o plano. Las planchas gruesas (más de 30 mm) pueden estar fuera de plano de 50 a 100 mm. La mayoría de las operaciones de Código G de una maquina CNC comenzarán a partir de una referencia conocida o una pieza de stock que tiene un tamaño y forma conocidos, y el Código G se escribe para desbastar el exceso y finalmente cortar la parte terminada. Con el plasma, el estado desconocido de la plancha hace que sea imposible generar un código G que satisfaga estas variaciones en el material.

Un arco de plasma tiene forma ovalada y la altura de corte debe controlarse para minimizar bordes biselados. Si la antorcha está demasiado alta o demasiado baja, los bordes pueden quedar excesivamente biselados. También es crítico que la antorcha se mantenga perpendicular a la superficie.

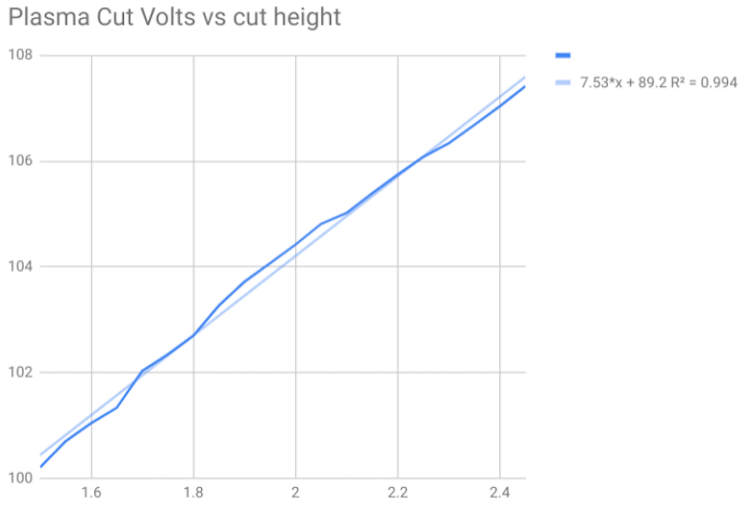
- La distancia de la antorcha al trabajo puede afectar el bisel del borde *



- Ángulo de corte negativo: * antorcha demasiado baja, aumente la distancia de la antorcha al trabajo.
- Ángulo de corte positivo: * antorcha demasiado alta, disminuir la antorcha a la distancia de trabajo.

NOTA: Una ligera variación en los ángulos de corte puede ser normal, siempre que esté dentro de tolerancia.

La capacidad de controlar con precisión la altura de corte en un entorno tan hostil y siempre cambiante es un desafío muy difícil. Afortunadamente, existe una relación muy lineal entre la altura de la antorcha (longitud del arco) y el voltaje del arco, como muestra este gráfico.



Este gráfico se preparó a partir de una muestra de aproximadamente 16,000 lecturas a diferentes alturas de corte y el análisis de regresión muestra 7.53 voltios por mm con un 99.4 % de confianza. En este caso particular, esta muestra fue tomada de una máquina Everlast de 50 amperios controlada por Linuxcnc.

El voltaje de la antorcha se convierte en una variable de control de proceso ideal para usarse en el ajuste de la altura de corte. Supongamos, por simplicidad, que el voltaje cambia en 10 voltios por mm. Esto se puede restablecer a 1 voltio por 0.1 mm (0.04”). Los principales fabricantes de máquinas de plasma (por ejemplo, Hypertherm, Thermal Dynamics y ESAB) producen tablas de corte que especifican la altura de corte recomendada y el voltaje de arco estimado a esta altura, así como algunos datos adicionales. Entonces, si el voltaje del arco es 1 voltio más alto que la especificación del fabricante, el controlador simplemente necesita bajar la antorcha en 0.1 mm (0.04 ”) para regresar a la altura de corte deseada. Tradicionalmente se usa una unidad de control de altura de la antorcha (THC) para administrar este proceso.

3.11.4. Elegir una máquina de plasma para operaciones CNC

Hay una gran cantidad de máquinas de plasma disponibles en el mercado hoy en día y no todas son aptas para el uso de CNC. El corte por plasma CNC es una operación compleja y se recomienda que los integradores elijan una máquina de plasma adecuada. Si no se hace, es probable que provoque horas y horas de problemas infructuosos tratando de evitar la falta de lo que muchos considerarían características obligatorias.

Si bien las reglas pueden romperse si se comprende completamente los motivos por los que se aplica la regla, consideramos que un constructor de mesas de plasma debería seleccionar una máquina con las siguientes características:

- Inicio blowback, para minimizar el ruido eléctrico y simplificar la construcción.
- Se prefiere una antorcha para maquina, pero muchos han usado antorchas manuales.
- Una punta de antorcha totalmente protegida para permitir la detección óhmica

Si tiene presupuesto, una máquina de gama alta le proporcionará:

- Tablas de corte del fabricante, que ahorrarán muchas horas y desperdicio de material al calibrar los parámetros de corte
- Contactos secos para ArcOK
- Terminales para el interruptor Arc On
- Voltaje de arco sin procesar o salida de voltaje de arco dividido
- Opcionalmente, una interfaz RS485 si utiliza un cortador de plasma Hypertherm y desea controlarlo desde la consola Linuxcnc.
- Ciclos de trabajo altos

En los últimos tiempos, una clase de máquina que incluye algunas de estas características está disponible por alrededor de 550 $. Un ejemplo es la Herocut55i disponible en Amazon, pero todavía no hay comentarios de los usuarios. Esta máquina cuenta con una antorcha blowback, salida ArcOK, contactos de inicio de antorcha y voltaje bruto de arco.

3.11.5. Tipos de control de altura de antorcha

La mayoría de las unidades de THC son dispositivos externos y muchas tienen un método de ajuste “bit bang” bastante burdo. Proporcionan dos señales al controlador LinuxCNC. Una se enciende si el eje Z debe moverse hacia arriba y la otra si el eje Z debe moverse hacia abajo. Ninguna señal es verdadera si la antorcha está a la altura correcta. El popular Proma 150 THC es un ejemplo de este tipo de THC. El componente Linuxcnc THCUD está diseñado para funcionar con este tipo de THC.

Con el lanzamiento de la interfaz de voltaje a frecuencia Mesa THCAD, LinuxCNC pudo decodificar el voltaje real de la antorcha a través de una entrada de encoder. Esto permitió que LinuxCNC controlara el eje Z y eliminara el hardware externo. Las primeras implementaciones que utilizan THCAD replicaron el enfoque "bit bang". El componente Linuxcnc THC es un ejemplo de este enfoque.

Jim Colt, de Hypertherm, dice que los mejores controladores de THC estaban completamente integrados en el controlador CNC. Por supuesto, se refería a los sistemas de gama alta fabricados por Hypertherm, Esab, Thermal Dynamics y otros, como Advanced Robotic Technology en Australia, sin soñar que el código abierto podría producir sistemas utilizando este enfoque que rivaliza con los sistemas de gama alta.

La inclusión de offsets externos en Linuxcnc V2.8 permitió que el control de plasma en LinuxCNC se elevara a un nivel completamente nuevo. Los offsets externos se refieren a la capacidad de aplicar un offset a la posición ordenada del eje, externa al controlador de movimiento. Esto es perfecto para el control de THC por plasma como un método para ajustar la altura de la antorcha en tiempo real según nuestra metodología de control de proceso elegida. Después de una serie de compilaciones experimentales, la configuración PlasmaC se incorporó a LinuxCNC 2.8. Este ha sido un proyecto extremadamente ambicioso y muchas personas en todo el mundo han participado en las pruebas y la mejora del conjunto de características. PlasmaC es único en el sentido de que su objetivo de diseño era admitir todos los THC, incluidos los de bit bang, hasta el sofisticado control de voltaje de la antorcha si el voltaje está disponible para LinuxCNC a través de un THCAD o algún otro sensor de voltaje. Además, PlasmaC está diseñado para ser un sistema independiente que no necesita ninguna subrutina adicional de código G y permite al usuario definir sus propias tablas de corte que están almacenadas en el sistema y accesibles mediante un menú desplegable.

3.11.6. Señal Arc OK

Las máquinas de plasma que tienen una interfaz CNC contienen un conjunto de contactos (por ejemplo, un relé) que se cierran cuando se establece un arco válido y cada lado de estos contactos serán pines en la interfaz CNC. Un integrador de mesas de plasma debe conectar un lado de estos pines a la alimentación de campo y el otro a un pin de entrada. Esto permite que el controlador CNC sepa cuándo se establece un arco válido y también cuándo se pierde un arco inesperadamente. Aquí hay una trampa potencial cuando la entrada es un circuito de alta impedancia, como una tarjeta Mesa. Si los contactos son un relé simple, existe una alta probabilidad de que la corriente que pasa a través del relé sea menor que la especificación de corriente mínima. En estas condiciones, los contactos del relé pueden sufrir una acumulación de óxido que con el tiempo puede dar como resultado una operación de contacto intermitente. Para evitar que esto suceda, se debe instalar una resistencia pull-down en el pin de entrada del controlador. Se debe tener cuidado para garantizar que esta resistencia se seleccione para asegurar que la corriente mínima pase a través del relé y tenga la potencia suficiente para manejar la potencia en el circuito. Finalmente, la resistencia debe montarse de tal manera que el calor generado no dañe nada mientras esté en funcionamiento.

Si tiene una señal ArcOK, se recomienda que se use por encima de cualquier señal sintetizada para eliminar posibles problemas de construccion. Una señal sintetizada disponible desde un THC externo o el Modo 0 de Plasmac no puede reemplazar completamente los circuitos ArcOK en un inverter de plasma. Se han observado algunos problemas de construcción en los que se ha producido una mala configuración o incompatibilidad con el inverter de plasma a partir de una señal sintetizada de ArcOK. Sin embargo, en general, una señal ArcOK sintetizada correctamente configurada está bien.

Se puede lograr una señal de arco de arco simple y efectiva con un simple relé reed. Envuelva 3 vueltas de uno de los cables gruesos de la cortadora de plasma (por ejemplo, el cable de sujeción del material) a su alrededor. Coloque el relé en un tubo para protección y conecte un lado del relé a la alimentación de campo y el otro extremo a su pin de entrada ArcOK.

3.11.7. Detección de altura inicial

Debido a que la altura de corte es un parámetro tan crítico del sistema y la superficie del material es inherentemente desigual, un mecanismo del eje Z necesita un método para detectar la superficie del material. Hay tres métodos que pueden lograr esto; detección de corriente para detectar un mayor par motor, un interruptor "flotante" y un circuito de detección eléctrico u "óhmico" que se cierra cuando el protector de la antorcha entra en contacto con el material. La detección de corriente no es una técnica viable para las mesas DIY, pero los interruptores de flotador y la detección óhmica si; se analizan a continuación:

3.11.7.1. Interruptores flotantes

La antorcha se monta en una plataforma deslizante que puede moverse hacia arriba cuando la punta de la antorcha hace contacto con la superficie del material y activa un interruptor o sensor. A menudo, esto se logra bajo el control de G-Code utilizando los comandos G38. Si este es el caso, luego de la prueba inicial, se recomienda sondear lejos de la superficie hasta que la señal de la sonda se pierda a una velocidad más lenta. Además, asegúrese de que se tenga en cuenta la histéresis del interruptor.

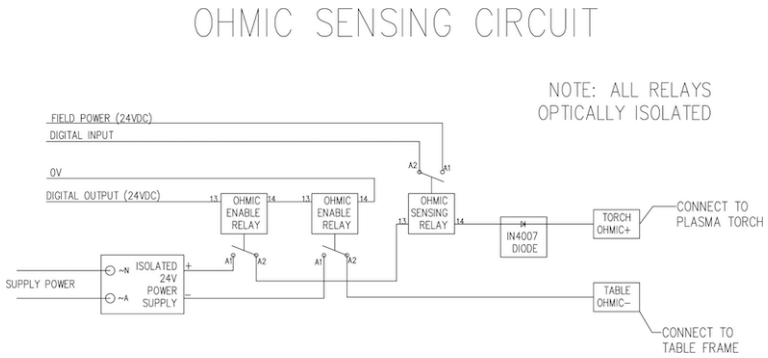
Independientemente del método de sondeo utilizado, se recomienda encarecidamente que se implemente el interruptor flotante para que haya una señal fallback o secundaria para evitar daños a la antorcha por un choque.

3.11.7.2. Detección óhmica

La detección óhmica se basa en el contacto entre la antorcha y el material que actúa como un interruptor para activar una señal eléctrica que es detectada por el controlador CNC. Siempre que el material esté limpio, este puede ser un método mucho más preciso para detectar el material que un interruptor flotante que puede causar la desviación de la superficie del material. Este circuito de detección óhmico está funcionando en un entorno extremadamente hostil, por lo que es necesario implementar una serie de medidas de seguridad para garantizar la seguridad tanto de la electrónica del CNC como del operador. En el corte por plasma, la abrazadera de tierra unida al material es positiva y la antorcha es negativa. Se recomienda que:

1. La detección óhmica solo se implementará cuando la antorcha tenga una envuelta aislada de la punta de la antorcha que transporta el arco de corte.
2. El circuito óhmico utiliza una fuente de alimentación aislada totalmente separada que activa un relé optoaislado para permitir que la señal de prueba se transmita al controlador CNC.
3. El lado positivo del circuito debe estar en la antorcha.
4. Ambos lados del circuito deben aislarse mediante relés optoaislados hasta que se realice el sondeo.
5. Se deben usar diodos de bloqueo para evitar que el voltaje de arco ingrese al circuito de detección óhmico.

El siguiente es un circuito de ejemplo que se ha demostrado que funciona y es compatible con la configuración de Linuxcnc PlasmaC.



3.11.8. Hipersensibilidad con MESA THCAD-5

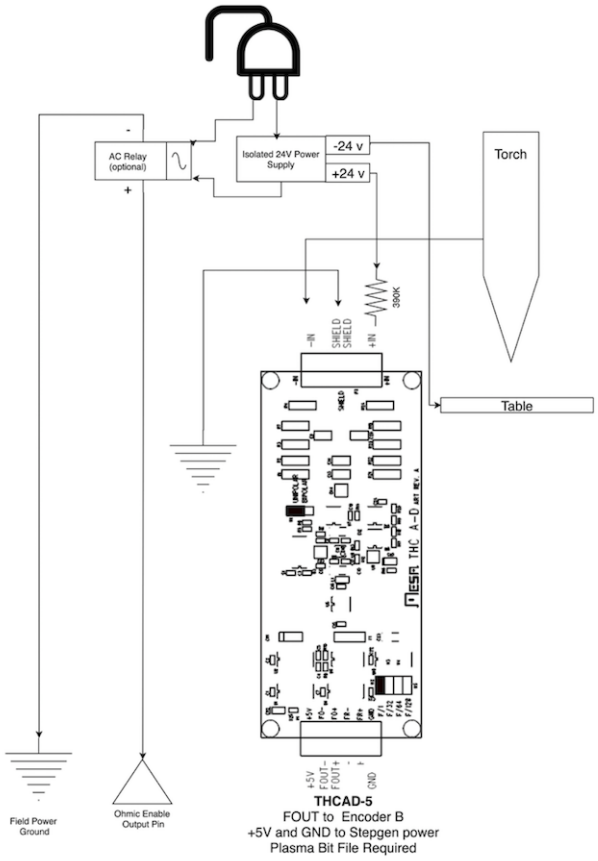
Un método más sofisticado de detección de material que elimina los relés y diodos es usar otro THCAD-5 para monitorear el voltaje del circuito de detección de material desde una fuente de alimentación aislada. La ventaja que tiene es que el THCAD está diseñado para el entorno eléctrico de plasma hostil y aísla totalmente y con seguridad el lado lógico del lado de alto voltaje.

Para implementar este método, se requiere una segunda entrada de encoder.

Si usa una tarjeta Mesa, hay disponible un firmware diferente para proporcionar 2 entradas adicionales del codificador A en los pines del codificador B y del codificador. Este firmware está disponible para descargar para las placas 7i76e y 7i96 desde el sitio web de Mesa en las páginas del producto.

El THCAD es lo suficientemente sensible como para ver el aumento en el voltaje del circuito a medida que aumenta la presión de contacto. El componente ohmic.comp incluido en Linuxcnc puede monitorear el voltaje de detección y establecer un umbral de voltaje por encima del cual se considera que se hace contacto y se habilita una salida. Al monitorear el voltaje, se puede establecer un umbral inferior de "circuito de interrupción" para construir una fuerte histéresis de interruptor. Esto minimiza los disparos falsos. En nuestras pruebas, descubrimos que la detección de material mediante este método era más sensible y robusta, además de ser más simple de implementar el cableado. Una ventaja adicional es el uso de salidas de software en lugar de pines físicos de E/S es que libera pines para usar para otros fines. Esta ventaja es útil para aprovechar al máximo la Mesa 7i96, que tiene pines de E/S limitados.

El siguiente diagrama de circuito muestra cómo implementar un circuito de hipersensibilidad.



Utilizamos una Fuente de alimentación aislada de riel DIN Mean Well HDR-15 ultradelgada de 15 vatios basada en riel DIN de 24 V .. Este es un dispositivo de doble aislamiento de clase II que resistirá cualquier voltaje de arco que pueda aplicarse a los terminales.

3.11.9. Ejemplo de código HAL para hipersensibilidad

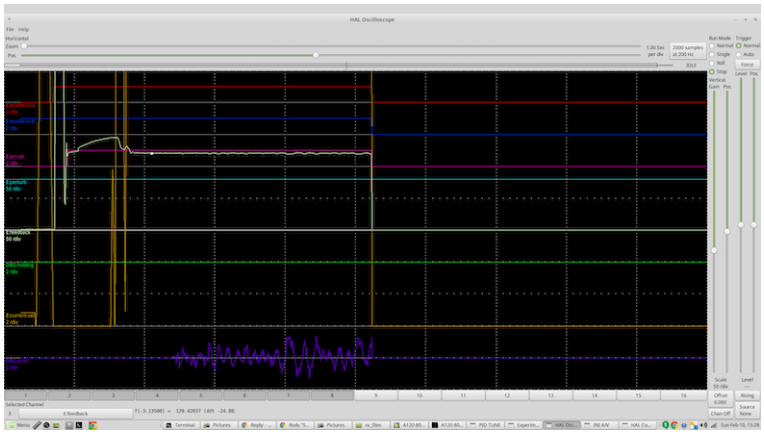
El siguiente código HAL se puede pegar en su plasmac_ connections.hal para habilitar la detección Ohmica en el codificador 2 de un 7i76e. Instale el archivo de bits correcto y conecte el THCAD a IDX + e IDX-. Asegúrese de cambiar la configuración de calibración para estar de acuerdo con su THCAD-5.

```
# --- Cargue el componente ---
```

```
loadrt ohmic names=ohmicsense
addf ohmicsense servo-thread

# --- 7i76e CONFIGURACIÓN DEL CODIFICADOR 2 PARA SENSADO OHMICO ---
setp hm2_7i76e.0.encoder.02.scale -1
setp hm2_7i76e.0.encoder.02.counter-mode 1

# --- Configure el componente ---
setp ohmicsense.thcad-0-volt-freq    140200
setp ohmicsense.thcad-max-volt-freq  988300
setp ohmicsense.thcad-divide         32
setp ohmicsense.thcad-fullscale      5
setp ohmicsense.volt-divider         4.9
setp ohmicsense.ohmic-threshold      22.0
setp ohmicsense.ohmic-low            1.0
net ohmic-vel ohmicsense.velocity-in <= hm2_7i76e.0.encoder.02.velocity

# --- Reemplace la señal de detección óhmica de Plasmac ---
unlinkp debounce.0.2.in
net ohmic-true ohmicsense.ohmic-on => debounce.0.2.in
net plasmac:ohmic-enable    =>  ohmicsense.is-probing
```

3.11.10. Retraso THC

Cuando se establece un arco, el voltaje del arco alcanza un pico significativo y luego se estabiliza a un voltaje estable a la altura de corte. Tal como se muestra por la línea verde en la imagen a continuación.



Es importante que el controlador de plasma "espere" antes de muestrear automáticamente el voltaje de la antorcha y comenzar el control de THC. Si se habilita demasiado pronto, el voltaje estará por encima de los voltios de corte deseados y la antorcha se bajará en un intento de abordar una condición de sobre-altura percibida.

En nuestras pruebas, esto varía entre máquinas y material de 0.5 a 1.5 segundos. Por lo tanto, una demora de 1.5 segundos después de que se recibe una señal válida de arcoOK antes de habilitar el control de THC es una configuración inicial segura. Si desea acortar esto para un material determinado, el Halscope de LinuxCNC le permitirá trazar el voltaje de la antorcha y tomar decisiones informadas sobre el menor retraso seguro a utilizar.

NOTA: Si la velocidad de corte no está cerca de la velocidad de corte deseada al final de este retraso, el controlador debe esperar hasta que esto se logre antes de habilitar el THC.

3.11.11. Muestreo de voltaje de antorcha

En lugar de confiar en las tablas de corte del fabricante para establecer el voltaje de la antorcha deseado, muchas personas (incluido el escritor) prefieren probar el voltaje a medida que el THC está habilitado y usarlo como un punto de ajuste.

3.11.12. Ruptura de antorcha

Se recomienda que se proporcione un mecanismo para permitir que la antorcha se "rompa" o se caiga en caso de impacto con el material o una parte cortada que se haya volcado. Debe instalarse un sensor para permitir que el controlador CNC detecte si esto ha ocurrido y pause el programa en ejecución. Por lo general, esto se implementa usando imanes para asegurar la antorcha a la etapa del eje Z.

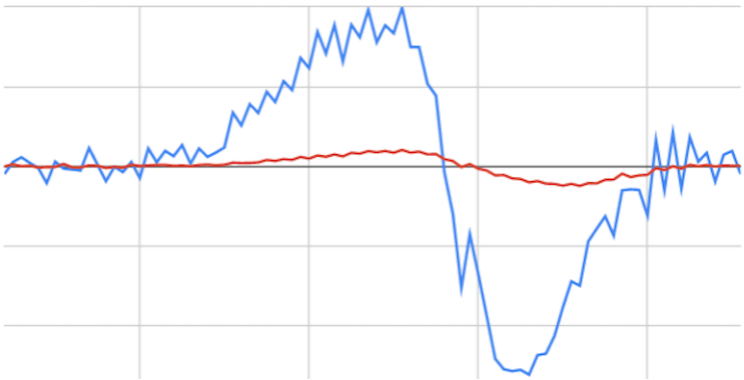
3.11.13. Esquinas; anti-inmersión por baja velocidad

El planificador de trayectoria Linuxcnc es responsable de traducir los comandos de velocidad y aceleración en movimiento que obedece las leyes de la física. Por ejemplo, el movimiento se ralentizará al negociar una esquina. Si bien esto no es un problema con las fresadoras o enrutadores, esto plantea un problema particular para el corte por plasma ya que el voltaje del arco aumenta a medida que el movimiento disminuye. Esto hará que el THC baje la antorcha. Una de las enormes ventajas de un control THC integrado en el controlador de movimiento LinuxCNC es que sabe lo que está sucediendo en todo momento. Por lo tanto, se convierte en una cuestión trivial controlar la velocidad actual (motion.current-speed) y mantener la operación de THC si cae por debajo de un umbral establecido (por ejemplo, 10 % por debajo de la velocidad de avance deseada.

3.11.14. Cruce de vacío / Kerf

Si la antorcha de plasma pasa sobre un vacío durante el corte, el voltaje del arco aumenta rápidamente y el THC responde con un movimiento violento hacia abajo que puede aplastar la antorcha en el material y dañarla. Esta es una situación que es difícil de detectar y manejar. Hasta cierto punto, puede mitigarse con buenas técnicas de anidación, pero aún puede ocurrir en material más grueso cuando un retal cae. Este es el único problema que aún no se ha resuelto dentro del movimiento de código abierto LinuxCNC.

Una técnica sugerida es monitorear la tasa de cambio en los voltios de la antorcha a lo largo del tiempo (dv / dt) porque este parámetro es de orden de magnitud mayor al cruzar un vacío que lo que ocurre debido a la deformación normal del material. El siguiente gráfico muestra una gráfica de baja resolución de dv / dt (en azul) mientras cruza un vacío. La curva roja es un promedio móvil de voltios de antorcha.



Por lo tanto, debería ser posible comparar el promedio móvil con dv/dt y detener la operación del THC una vez que dv/dt exceda el rango normal esperado debido a la deformación. Se necesita más trabajo en esta área para encontrar una solución de trabajo en LinuxCNC.

3.11.15. Agujero y corte de forma pequeña

Se recomienda reducir la velocidad de corte al cortar agujeros y formas pequeñas.

John Moore dice: “Si desea detalles sobre el corte de agujeros pequeños y precisos, consulte las hojas de ventas de Hypertherm " True Hole Technolog " y también en PlasmaSpider, el usuario Seanp ha publicado mucho sobre su trabajo utilizando plasma de aire simple.

El método generalmente aceptado para obtener buenos agujeros de 37 mm de diámetro. y hasta el grosor del material con un mínimo cono usando un plasma de aire es:

1. Utilice la corriente de corte recomendada para los consumibles.
2. Utilice la altura de corte recomendada fija (sin THC) para los consumibles.
3. Corte del 60 % al 70 % de la tasa de alimentación recomendada de consumibles y materiales.
4. Comience en o cerca del centro del agujero.
5. Use entrada perpendicular.
6. Sin salida, ya sea una ligera quemadura o un apagado de antorcha temprano dependiendo de lo que funcione mejor para usted.

Tendrá que experimentar para obtener el tamaño exacto del agujero porque el corte con este método será más ancho que su corte recto habitual ".

Esta ralentización se puede lograr manipulando la velocidad de alimentación directamente en su postprocesador o utilizando alimentación adaptativa y un pin analógico como entrada. Esto le permite usar M67 / M68 para establecer el porcentaje de alimentación deseada para cortar.

- Conociendo el avance

De la discusión anterior es evidente que el controlador de plasma necesita conocer la velocidad de alimentación establecida por el usuario. Esto plantea un problema con LinuxCNC porque LinuxCNC no guarda la velocidad de avance después de que el código G se almacena y analiza. Hay dos enfoques para solucionar esto:

1. Vuelva a asignar el comando F y guarde la velocidad de avance ordenada establecida en G-Code a través de un comando M67 / M68
2. Almacenar las tablas de corte en el controlador de plasma y permitir que el programa G-Code consulte la velocidad de avance actual (como lo hace PlasmaC)

Una rama experimental de Linuxcnc que sería útil para el corte por plasma fue la rama de etiquetas de estado. Esto agrega una "etiqueta" que está disponible para el movimiento que contiene las velocidades de avance y velocidad actuales para todos los comandos de movimiento activos. Se ha fusionado y estará en LinuxCNC v2.9

3.11.16. Pines de E/S para controladores de plasma

Los cortadores de plasma requieren varios pines adicionales. En LinuxCNC, no hay reglas estrictas sobre qué hace cada pin. En esta discusión asumiremos que el plasma tiene una interfaz CNC y que la tarjeta controladora tiene entradas activas altas en uso (por ejemplo, Mesa 7i76e).

Las mesas de plasma pueden ser máquinas grandes y le recomendamos que se tome el tiempo de instalar interruptores de límite máximo / mínimo e interruptores home separados para cada articulacion. La excepción podría ser el límite inferior del eje Z. Cuando se activa un interruptor home, la articulación se desacelera bastante lentamente para obtener la máxima precisión. Esto significa que si desea utilizar velocidades de homing que sean proporcionales al tamaño de la mesa, puede sobrepasar el punto de activación inicial en 50-100 mm. Si utiliza un interruptor de límite/home compartido, debe mover el sensor fuera del punto de activación con HOME_OFFSET final o activará una falla del interruptor de límite cuando la máquina salga de la referencia. Esto significa que podría perder 50 mm o más de recorrido del eje con interruptores home/límite compartidos. Esto no sucede si se usan interruptores home y límite separados.

Por lo general, se requieren los siguientes pines (tenga en cuenta que las conexiones sugeridas pueden no ser apropiadas para una configuración de PlasmaC):

3.11.16.1. Arco OK (entrada)

- El inverter cierra los contactos cuando se establece un arco válido.
- Conecte la alimentación de campo a un terminal ArcOK del inverter.
- Conecte el otro terminal al pin de entrada.
- Usualmente conectado a uno de los pines *motion.digital-<nn>* para usar desde G-Code con M66

3.11.16.2. Antorcha ON (salida)

- Activa un relé para cerrar la antorcha en el interruptor en el inverter
- Conecte la antorcha en los terminales del inverter a los terminales de salida del relé
- Conecte un lado de la bobina del relé al pin de salida
- Conecte el otro lado de la bobina del relé a tierra de la alimentación de campo.
- Si se utiliza un relé mecánico, conecte un diodo flyback (por ejemplo, uno de la serie IN400x) a través de los terminales de la bobina con la banda del diodo apuntando hacia el pin de salida
- Si se utiliza un relé de estado sólido, puede ser necesario observar la polaridad en las salidas
- En algunas circunstancias, se puede usar el relé del husillo integrado en las tarjetas Mesa en lugar de un relé externo.
- Generalmente conectado a *spindle.0.on*

[WARNING] Se recomienda encarecidamente que la antorcha no se pueda habilitar mientras este pin sea falso; de lo contrario, la antorcha no se apagará cuando se presione Estop;

3.11.16.3. Interruptor flotante (entrada)

- Utilizado para sondeo de superficie. Un sensor o interruptor que se activa si la antorcha se desliza hacia arriba cuando toca el material.
- Conecte la salida del sensor de proximidad al pin de entrada elegido. Si se utilizan interruptores mecánicos, conecte un lado del interruptor a la alimentación de campo y el otro lado a la entrada.
- Generalmente conectado a *motion.probe-input*

3.11.16.4. Activación del sensor óhmico (salida)

- Vea el esquema de detección óhmica.
- Conecte el pin de salida a un lado de los relés de aislamiento y el otro lado a tierra de la alimentación de campo.
- En una configuración que no sea PlasmaC, generalmente activado por un *motion.digital-out-<nn>* para que M62 / M63 / M64 / M65 pueda controlarlo en G-Code

3.11.16.5. Detección óhmica (entrada)

- Tenga cuidado de seguir el esquema de detección óhmica mostrado anteriormente.
- Una fuente de alimentación aislada activa un relé cuando el protector de la antorcha hace contacto con el material.
- Conecte la alimentación de campo a un terminal de salida y el otro a la entrada.
- Tenga cuidado de observar la polaridad del relé si se utilizan relés de estado sólido optoacoplados.
- Por lo general, está conectado a *motion.probe-input* y se puede conectar con el interruptor flotante.

Como se puede ver, las mesas de plasma son intensivas en pines y ya hemos consumido alrededor de 15 entradas antes de que se agreguen los estop normales. Otros tienen otros puntos de vista, pero es la opinión del escritor que la tarjeta Mesa 7i76e es preferible sobre la más barata 7i96 para permitir el MPG, switch de selección de escala y eje y otras características que puede agregar con el tiempo. Si su mesa usa servos, hay varias alternativas. Si bien hay otros proveedores, diseñar su máquina alrededor del ecosistema de Mesa simplificará el uso de su placa THCAD para leer el voltaje del arco.

3.11.16.6. Breakaway de la antorcha

- Como se mencionó anteriormente, se debe instalar un sensor de ruptura que se dispara si la antorcha se estrella y se cae.
- Por lo general, esto se conectaría a *halui.program-pause* para que la falla se pueda rectificar y se reanude el programa.

3.11.17. Código G para controladores de plasma

La mayoría de los controladores de plasma ofrecen un método para cambiar la configuración de G-Code. Linuxcnc admite esto a través de M67/M68 para comandos analógicos y M62-M65 para digital (comandos de encendido/apagado). Cómo se implemente esto es totalmente arbitrario. Veamos cómo la configuración de LinuxCNC PlasmaC hace esto:

3.11.17.1. Seleccione la configuración del material en PlasmaC y use la velocidad de avance para ese material:

```
M190 Pn
M66 P3 L3 Q1
F # <_ hal [plasmac.cut-feed-rate]>
M3 S1
```

NOTA: Los usuarios con una gran cantidad de entradas en la tabla de materiales de PlasmaC pueden necesitar aumentar el parámetro Q1 (por ejemplo, Q2)

3.11.17.2. Habilitar/deshabilitar la operación del THC:

```
M62 P2 desactivará el THC (sincronizado con movimiento)
M63 P2 habilitará el THC (sincronizado con movimiento)
M64 P2 desactivará el THC (inmediatamente)
M65 P2 habilitará el THC (inmediatamente)
```

3.11.17.3. Reducir las velocidades de corte: (por ejemplo, para cortar agujeros)

```
M67 E3 Q0 establecería la velocidad al 100% de la velocidad solicitada
M67 E3 Q40 establecería la velocidad al 40% de la velocidad solicitada
M67 E3 Q60 establecería la velocidad al 60% de la velocidad solicitada
M67 E3 Q100 establecería la velocidad al 100% de la velocidad solicitada
```

3.11.17.4. Compensación del cortador:

```
G41.1 D # <_ hal [plasmac_run.kerf-width-f]>; para la izquierda de la ruta programada
G42.1 D # <_ hal [plasmac_run.kerf-width-f]> para la derecha de la ruta programada
G40 para desactivar la compensación
```

NOTA: Los integradores deben familiarizarse con la documentación de Linuxcnc para los diversos comandos de Linuxcnc G-Code mencionados anteriormente.

3.11.18. Offsets externos y corte por plasma

Los offsets externos se introdujeron en Linuxcnc con la versión 2.8. Por externo, significa que podemos aplicar un desplazamiento externo al Código G del que el planificador de trayectorias no sabe nada. Es más fácil de explicar con un ejemplo. Imagine un torno con un desplazamiento externo aplicado por una fórmula matemática para mecanizar el lobulo en una leva. El torno gira a ciegas con el diámetro de corte establecido en un diámetro fijo y el desplazamiento externo mueve la herramienta hacia adentro y hacia afuera para mecanizar el lóbulo de la leva a través de un desplazamiento externo aplicado. Para configurar nuestro torno

para mecanizar esta leva, necesitamos asignar una parte de la velocidad y aceleración del eje a offsets externos o la herramienta no puede moverse. Aquí es donde entra en juego la variable ini OFFSET_AV_RATIO. Digamos que decidimos que necesitamos asignar el 20 % de la velocidad y la aceleración al offset externo al eje Z.Establecemos esto igual a 0.2. La consecuencia de esto es que su velocidad y aceleración máximas para el eje Z del Torno es solo el 80 % de lo que podría ser.

Los offsets externos son un método muy poderoso para hacer ajustes de altura de la antorcha al eje Z a través de un THC. Pero el plasma tiene que ver con las altas velocidades y la aceleración rápida, por lo que no tiene sentido limitar estos parámetros. Afortunadamente en una máquina de plasma, el eje Z está 100 % controlado por el THC o no lo está. Durante el desarrollo de los offsets externos de Linuxcnc se reconoció que el movimiento del eje Z por G-Code y por THC eran mutuamente excluyentes. Esto nos permite engañar a los offsets externos para que den el 100 % de la velocidad y la aceleración todo el tiempo. Podemos hacer esto duplicando los ajustes de velocidad y aceleración del eje Z de la máquina en el archivo ini y establecer OFFSET_AV_RATIO = 0.5. De esa manera, el 100 % de la velocidad y aceleración máximas estarán disponibles tanto para sondeo como para THC.

Ejemplo: En una máquina métrica con un motor NEMA23 con un accionamiento directo a un husillo de bolas de 5 mm, se determinó que una velocidad máxima de 60 mm/s y una aceleración de 700 mm/s/s eran valores seguros sin pérdida de pasos. Para esta máquina, configure el eje Z en el archivo ini de la siguiente manera:

```
[AXIS_Z]
OFFSET_AV_RATIO = 0.5
MAX_VELOCITY = 120
MAX_ACCELERATION = 1400
```

La articulacion asociada con este eje tendría las variables de velocidad y aceleración establecidas de la siguiente manera:

```
[JOINT_n]
MAX_VELOCITY = 60
MAX_ACCELERATION = 700
```

Para obtener más información sobre las compensaciones externas (para V 2.8 y superior), lea la Seccion [AXIS_<letter>] del documento del archivo INI y Offsets externos en la documentación de Linuxcnc.

3.11.19. Lectura de voltaje de arco con el THCAD de Mesa

La tarjeta Mesa THCAD es un convertidor de voltaje a frecuencia, preciso y con buen precio, que está diseñado para el entorno eléctrico ruidoso y hostil asociado con el corte por plasma. Internamente tiene un rango de 0-10 voltios. Este rango puede ampliarse simplemente agregando algunas resistencias como se describe en la documentación. Esta placa está disponible en tres versiones, el THCAD-5, más nuevo, con un rango de 0-5 voltios, el THCAD-10 con un rango de 0-10 voltios y el THCAD-300 que está precalibrado para un rango extendido de 300 voltios. Cada placa se calibra individualmente y se aplica una etiqueta adhesiva a la placa que indica la frecuencia a 0 voltios y a escala completa. Para usar con LinuxCNC, se recomienda que el divisor 1/32 sea seleccionado por el enlace apropiado en la placa. En este caso, asegúrese de dividir también las frecuencias establecidas por 32.Esto es más apropiado para el hilo servo de 1 kHz y también le da más tiempo al THCAD para promediar y suavizar la salida.

Hay mucha confusión acerca de cómo decodificar la salida THCAD, así que consideremos la Mesa 7i76e y el THCAD-10 por un momento con los siguientes datos de calibración hipotéticos:

- Escala completa 928,000 Hz (1/32 29,000 Hz)
- 0 voltios 121,600 Hz (1/32 3,800 Hz)

Debido a que la escala completa es de 10 voltios, entonces la frecuencia por voltio es:

```
(29,000 - 3,800) / 10 = 2,520 Hz por voltio
```

Asumiendo que tenemos una entrada de 5 voltios, la frecuencia calculada sería:

```
(2,520 * 5) + 3,800 = 16,400
```

Ahora debería quedar bastante claro cómo convertir la frecuencia a su voltaje equivalente:

```
Voltios = (frecuencia - 3,800) / 2,520
```

3.11.19.1. Conexiones THCAD

En el lado de alto voltaje:

- Conecte el voltaje de arco dividido o sin procesar a IN + e IN-
- Conecte el blindaje del cable de interconexión a la conexión de blindaje.
- Conecte el otro terminal del blindaje a tierra.

Suponiendo que está conectado a una 7i76e, conecte la salida a la entrada del codificador de husillo:

- THCAD + 5v a TB3 Pin 6 (+5 VP)
- THCAD -5v a TB3 Pin 1 (GND)
- THCAD FOUT + a TB3 Pin 7 (ENC A +)
- THCAD FOUT- a TB3 Pin 8 (ENC A-)

3.11.19.2. Prueba inicial de THCAD

Asegúrese de tener las siguientes líneas en su archivo ini (suponiendo una Mesa 7i76e):

```
setp hm2_7i76e.0.encoder.00.scale -1
setp hm2_7i76e.0.encoder.00.counter-mode 1
```

Encienda su controlador y abra Halshow (Axis: Mostrar Configuración HAL), profundice para encontrar el pin *hm2_7i76e.0.encoder.00.v*
Con 0 voltios aplicados, debería estar rondando la frecuencia de 0 voltios (3.800 en nuestro ejemplo). Tome una batería de 9 voltios y conéctela a IN + e IN-. Para un THCAD-10 ahora puede calcular la velocidad esperada. (26,480 en nuestro ejemplo hipotético). Si pasa esta prueba, está listo para configurar su controlador de plasma LinuxCNC.

3.11.19.3. Qué modelo THCAD usar

El THCAD-5 es útil si tiene la intención de usarlo para la detección óhmica. No hay duda de que el THCAD-10 es el dispositivo más flexible y es fácil alterar la escala. Sin embargo, hay una advertencia que puede entrar en juego con algunos cortadores de plasma más baratos con un divisor de voltaje incorporado. Es decir, las resistencias internas pueden ser percibidas por el THCAD como parte de su propia resistencia externa y devolver resultados erróneos. Por ejemplo, el divisor 16:1 en las cortadoras de plasma Everlast debe tratarse como 24:1 (y 50:1 se convierte en 75:1). Esto no es un problema con marcas de mayor reputación (p. Ej., Thermal Dynamics, Hypertherm, ESAB, etc.). Por lo tanto, si observa voltajes de corte inferiores a los esperados, podría ser preferible volver a configurar el THCAD para leer el voltaje de arco sin procesar.

Recordando que los voltajes de arco de plasma son potencialmente letales, aquí hay algunos criterios sugeridos.

Inicio del arco piloto Debido a que no es probable que haya ningún EMI significativo, debería poder instalar de forma segura el THCAD en su panel de control si ha seguido nuestras pautas de construcción.

- Si no tiene un divisor de voltaje, instale resistencias de escala dentro del cortador de plasma e instale el THCAD en el panel de control o siga las sugerencias para máquinas de arranque de alta frecuencia.
- Si tiene un divisor de voltaje, instale un THCAD-10 en su panel de control. No hemos tenido problemas con esta configuración con un cortador de plasma Thermal Dynamics de 120 amp.

  1. Inicio HF

Instale el THCAD en el inverter ya que la señal de frecuencia es mucho más inmune al ruido EMI.

- Si no tiene un divisor de voltaje y tiene espacio dentro del cortador de plasma, instale un THCAD-300 dentro del cortador de plasma.

- Si no tiene un divisor de voltaje y no tiene espacio dentro del cortador de plasma, instale un THCAD-10 en una caja de metal fuera del cortador de plasma e instale el 50 % de la resistencia de escala en cada uno de los IN + e IN- dentro la carcasa del cortador de plasma para que no salgan voltajes letales de la carcasa.
- Si tiene un divisor de voltaje, instale un THCAD-10 en una caja de metal fuera del cortador de plasma

Tensión de arco presentada en un conector En este caso, independientemente del método de inicio del arco, probablemente ya haya resistencias incluidas en los circuitos para evitar choques letales, por lo que se recomienda un THCAD-10 para que esta resistencia (generalmente 200k Ohms) pueda tenerse en cuenta al elegir una resistencia de escala en tanto estas resistencias distorsionarán el voltaje reportado por el THCAD-300.

3.11.20. Post procesadores y anidamiento

El plasma no es diferente a otras operaciones de CNC en tanto en cuanto:

1. Es diseñado en CAD (donde se emite como un formato DXF o, a veces, SVG).
2. Es procesado en CAM para generar el código G final que se carga en la máquina
3. Corta las piezas mediante comandos de código G del CNC.

Algunas personas logran buenos resultados con las herramientas Inkscape y G-Code, pero SheetCam es una solución de muy buen precio y hay una serie de posprocesadores disponibles para Linuxcnc. SheetCam tiene una serie de características avanzadas diseñadas para el corte por plasma y, por el precio, es una obviedad para cualquiera que realice un corte por plasma regular.

3.11.21. Diseñando para entornos eléctricos ruidosos

El corte por plasma es inherentemente un entorno eléctrico extremadamente hostil y ruidoso. Si tiene problemas de EMI, las cosas no funcionarán correctamente. En un ejemplo más que obvio, puede que al encender la antorcha, la computadora se reinicie, pero puede tener cualquier número de síntomas extraños. Casi todo sucederá solo cuando la antorcha esté cortando, a menudo cuando se dispara por primera vez.

Por lo tanto, los creadores de sistemas deben seleccionar los componentes con cuidado y diseñar desde cero para hacer frente a este entorno hostil para evitar el impacto de la interferencia electromagnética (EMI). De lo contrario, se podrían generar innumerables horas de resolución infructuosa.

Elegir placas ethernet como la Mesa 7i76e, o la más barata 7i96, ayuda al permitir que la PC se ubique lejos de la electrónica y la máquina de plasma. Este hardware también permite el uso de sistemas lógicos de 24 voltios que son mucho más tolerantes al ruido. Los componentes deben montarse en una carcasa metálica conectada a la tierra de la red electrica. Se recomienda encarecidamente que se instale un filtro EMI en la conexión de alimentación de red. La forma más sencilla es utilizar un conector IEC de alimentación de red con filtro EMI de uso común en PC y electrodomésticos, lo que permite lograrlo sin trabajo adicional. Planifique la disposición de los componentes en el cuadro para que la alimentación de la red, los cables del motor de alto voltaje y las señales lógicas se mantengan lo más separadas posible entre sí. Si tienen que cruzar, manténgalas a 90 grados.

Peter Wallace de Mesa Electronics sugiere; “Si tiene una fuente de plasma compatible con CNC con un divisor de voltaje, montaría el THCAD dentro de su caja electrónica con el resto del hardware de movimiento. Si tiene una fuente de plasma manual y está leyendo voltaje de plasma sin procesar, montaría el THCAD lo más cerca posible de la fuente de plasma (incluso dentro de la caja de la fuente de plasma si cabe). En este caso, asegúrese de que todo el lado bajo de las conexiones THCAD están completamente aisladas de la fuente de plasma. Si utiliza una caja blindada para el THCAD, el blindaje debe conectarse a la tierra de su caja de electrónica, no a la tierra de la fuente de plasma ".

Se recomienda pasar cables de tierra separados de las cajas de los motores y la antorcha de regreso a un punto central de conexión a tierra de la máquina. Conecte el cable de tierra de plasma a este punto y, opcionalmente, una varilla de tierra clavada en el suelo lo más cerca posible de la máquina (especialmente si se trata de una máquina de plasma de alta frecuencia).

El cableado externo a los motores debe estar blindado y dimensionado adecuadamente para manejar la corriente que pasa por el circuito. El blindaje debe dejarse desconectado en el extremo del motor y conectado a tierra en el extremo de la caja de control. Considere usar un pin adicional en cualquier conector en la caja de control para que la tierra pueda extenderse hasta la caja de control y conectarse a tierra al chasis directamente en el controlador del motor paso a paso / servo.

Somos conscientes de que al menos un fabricante de sistemas comerciales ha tenido problemas con el ruido eléctrico inducido en el circuito de detección óhmico. Si bien esto puede mitigarse utilizando ferritas y enrollando el cable, también se recomienda agregar un filtro de alimentación a través de la línea de alimentación donde la señal de detección óhmica ingresa al gabinete de la electrónica.

Tommy Berisha, el maestro de la construcción de máquinas de plasma con un presupuesto ajustado, dice: “Si tiene un presupuesto limitado, considere usar viejos alimentadores de energía para computadoras portátiles. Son muy buenos, el filtrado es bueno, completamente aislado, limitados por la corriente (esto se vuelve muy importante cuando algo sale mal), y ajustar 2 o 3 de ellos en serie es fácil ya que están aislados (tenga en cuenta que algunos tienen la conexión a tierra al terminal de salida negativa, por lo que tiene que ser desconectado, simplemente haciendo uso de un cable de alimentación sin contactos a tierra) ".

3.11.22. Mesas de agua

El nivel mínimo de agua debajo del nivel de corte de la antorcha debe ser de alrededor de 40 mm, tener espacio debajo de los listones es agradable para que el agua pueda nivelarse y escapar durante el corte, tener un poco de agua sobre la placa de metal que se está cortando es realmente agradable en la medida que se deshace del polvo, dejarlo sumergido es la mejor manera, pero no es preferible para sistemas con uso a tiempo parcial, ya que corroerá la antorcha. Agregar bicarbonato de sodio al agua mantendrá la mesa en buenas condiciones durante muchos años, ya que no permite la corrosión mientras los listones están bajo el agua y también reduce el olor del agua. Algunas personas usan un depósito de agua con una entrada de aire comprimido para poder empujar el agua desde el depósito hasta la mesa a demanda y así permitir cambios en los niveles de agua.

3.11.23. Mesas de tiro descendente

Muchas mesas comerciales utilizan un diseño de tiro descendente, por lo que los ventiladores se utilizan para aspirar el aire a través de los listones para capturar humos y chispas. A menudo, las mesas se dividen en zonas, por lo que solo una sección debajo de la antorcha se abre a la ventilación de salida, a menudo utilizando rams de aire y solenoides de aire para abrir las persianas. La activación de estas zonas es relativamente sencilla si utiliza la posición de la articulacion o del eje desde uno de los pines de movimiento y el componente lincurve para mapear las zonas de tiro descendente al pin de salida correcto.

3.11.24. Diseñando para velocidad y aceleración

En el corte por plasma, la velocidad y la aceleración son son los reyes Cuanto mayor es la aceleración, menos necesita reducir la velocidad de la máquina al negociar curvas. Esto implica que el pórtico debe ser lo más ligero posible sin sacrificar la rigidez torsional. Una caja de aluminio de seccion 100 mm x 100 mm x 2 mm tiene una rigidez torsional equivalente a un perfil extruido de ranuras T de 80 mm x 80 mm, pero es un 62 % más liviana. Por tanto, ¿la facilidad de las ranuras en T supera el trabajo de construcción adicional?

3.11.25. Distancia recorrida por revolución del motor

Los motores paso a paso sufren de resonancia y es probable que un piñón de transmisión directa signifique que el motor está funcionando en condiciones desfavorables. Idealmente, para máquinas de plasma, una distancia de alrededor de 15-25 mm por revolución del motor se considera ideal, pero incluso alrededor de 30 mm por revolución sigue siendo aceptable. Un tornillo de bolas de 5 mm de paso con una unidad de reducción de 3:1 o 5:1 es ideal para el eje Z.

3.11.26. PlasmaC, Configuración de plasma LinuxCNC

La configuración de PlasmaC se compone de un componente HAL (plasmac.hal) más configuraciones completas para Axis y Gmoccapy ha recibido una contribución considerable de muchos en el movimiento de código abierto LinuxCNC que ha avanzado en la comprensión de los controladores de plasma desde aproximadamente 2015. Se han realizado muchas pruebas y trabajos de desarrollo para lograr que PlasmaC alcance su estado de funcionamiento actual. Se ha incluido todo, desde el diseño del circuito hasta el control y la configuración del código G. Además, PlasmaC admite THC externos como el Proma 150, pero realmente se destaca cuando se combina con un controlador Mesa, ya que esto permite que el integrador incluya el convertidor de voltaje a frecuencia Mesa THCAD, diseñado específicamente para lidiar con el entorno de plasma hostil.

PlasmaC está diseñado para ser independiente e incluye la capacidad de incluir sus tablas de corte, pero también incluye características para ser utilizadas con un postprocesador como SheetCam.

El sistema PlasmaC ahora está incluido en la Versión 2.8 y superior de Linuxcnc. Ahora es bastante maduro y se ha mejorado significativamente desde que se escribió la primera versión de esta guía. Plasmac definirá el soporte de plasma de LinuxCNC durante muchos años, ya que incluye todas las características de un sistema patentado de control de plasma de gama alta a un precio de código abierto.

3.11.27. Control Hypertherm RS485

Algunas cortadoras de plasma Hypertherm tienen una interfaz RS485 para permitir que el controlador (por ejemplo, Linuxcnc) establezca amperios presión y el modo. Varias personas han utilizado un componente de espacio de usuario escrito en Python para lograr esto. Ahora, Plasmac ahora admite esta interfaz de forma nativa. Consulte la documentación de Plasmac para saber cómo usarlo.

La combinación de una velocidad de transmisión lenta utilizada por Hypertherm y el componente de espacio de usuario hace que esto sea bastante lento para alterar los estados de la máquina, por lo que generalmente no es viable cambiar la configuración sobre la marcha mientras se corta.

Al seleccionar una interfaz RS485 para usar en el extremo de la PC, los usuarios informaron que los interfaces USB a RS485 no son confiables. Se han logrado resultados buenos y confiables utilizando una interfaz RS232 basada en hardware (por ejemplo, PCI/PCIe o puerto de placa base) y un convertidor RS485 apropiado. Algunos usuarios han informado de éxito con una tarjeta Sunix P/N:SER5037A PCI RS2322 y un convertidor genérico XC4136 RS232 a RS485 (que a veces también puede incluir un cable USB).

3.11.28. Postprocesadores para corte por plasma

Los programas CAM (Fabricación asistida por computadora) son el puente entre CAD (Diseño asistido por computadora) y la operación final del CNC (Control numérico por computadora). A menudo incluyen un postprocesador configurable por el usuario para definir el código que se genera para una máquina o dialecto específico de G-Code.

Muchos usuarios de Linuxcnc están perfectamente contentos con el uso de Inkscape para convertir archivos basados en vectores .SVG a G-Code. Si está utilizando un plasma para hobby o uso doméstico, considere esta opción. Sin embargo, si sus necesidades son más complejas, probablemente la mejor y más razonable solución sea SheetCam. SheetCam es compatible con Windows y Linux y hay disponibles postprocesadores, incluida la configuración de PlasmaC. SheetCam le permite anidar partes sobre una hoja completa de material y le permite configurar conjuntos de herramientas y fragmentos de código para satisfacer sus necesidades. Los postprocesadores SheetCam son archivos de texto escritos en el lenguaje de programación Lua y, en general, son fáciles de modificar para adaptarse a sus requisitos exactos. Para obtener más información, consulte https://sheetcam.com [sitio web de SheetCam] y su foro de soporte.

Se incluye otro postprocesador popular con el popular paquete Fusion360, pero los postprocesadores incluidos necesitarán cierta personalización.

LinuxCNC es una aplicación CNC y las discusiones sobre técnicas CAM distintas a esta discusión introductoria están fuera del alcance de LinuxCNC.

Capítulo 4

Interfaces de usuario

4.1. GUI AXIS

4.1.1. Introducción

AXIS es una interfaz gráfica para LinuxCNC que cuenta con vista previa y backplot. Está escrito en Python y utiliza Tk y OpenGL para mostrar la interfaz de usuario.

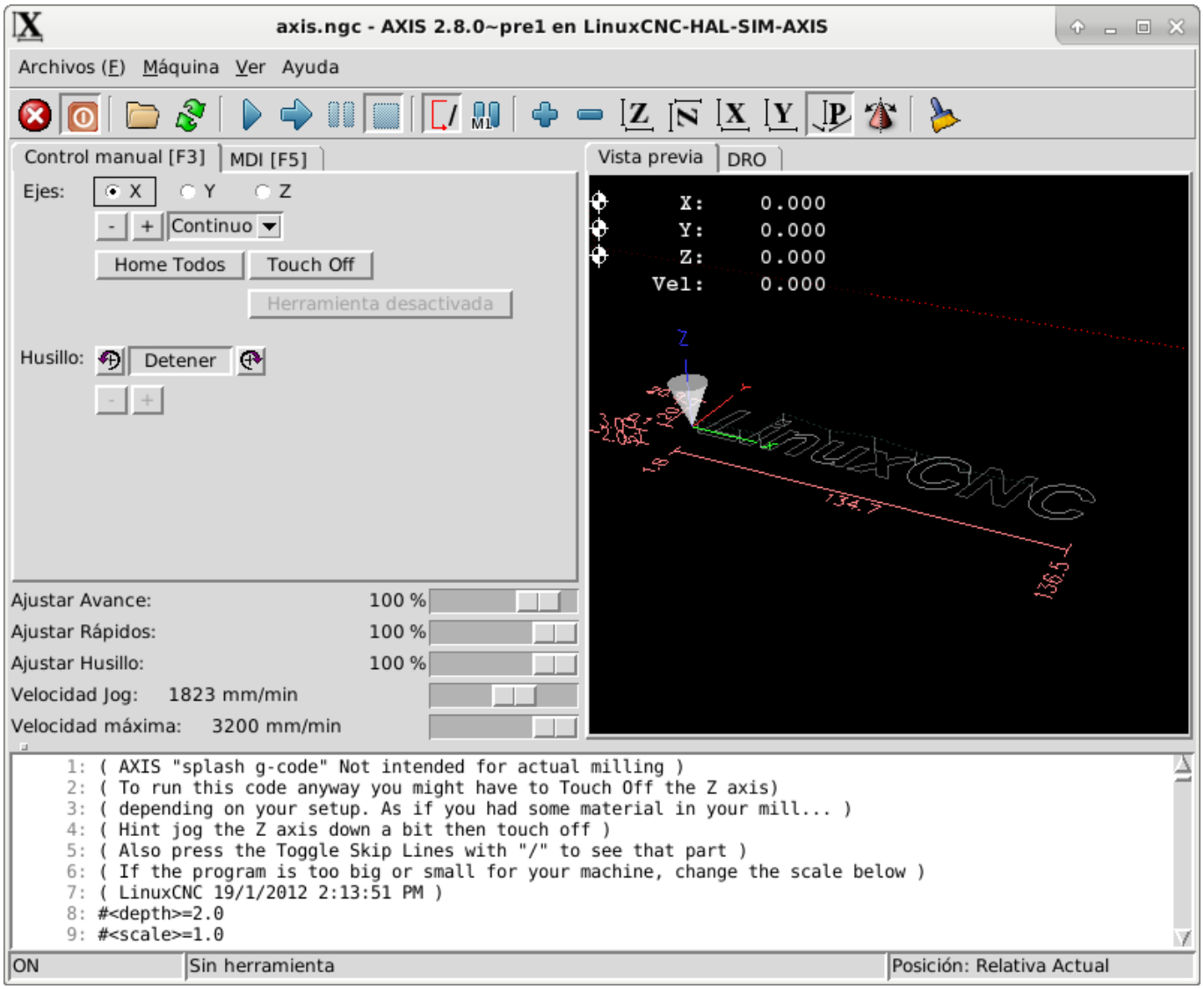


Figura 4.1: Ventana de AXIS

4.1.2. Primeros pasos

Si su configuración no está configurada actualmente para usar AXIS, puede cambiarlo editando el archivo .ini. En la sección *[DISPLAY]* indiquelo con la linea *DISPLAY = axis*.

La configuracion de ejemplo *sim/axis.ini* ya está configurada para usar AXIS como su interfaz de usuario.

4.1.2.1. Una sesión típica

1. Inicie LinuxCNC.
2. Reinicie E-STOP (F1) y encienda la alimentación de la máquina (F2).
3. Haga Homing de todos los ejes.
4. Cargue el archivo g-code.
5. Utilice el gráfico de vista previa para verificar que el programa es correcto.

6. Cargue el material.
7. Establezca el offset adecuado para cada eje haciendo jogging y usando la tecla Touch Off según sea necesario.
8. Ejecute el programa.

nota
Ejecutar el mismo programa de nuevo dependera de su configuración y requisitos. Es posible que deba cargar más material y establecer offsets o desplazarse y establecer un nuevo offset y luego ejecutar el programa de nuevo. Si la ubicacion del material es siempre la misma, entonces es posible que solo tenga que volver a ejecutar el programa. Ver la seccion Menu Maquina para obtener más información sobre el comando de ejecución.

4.1.3. Pantalla AXIS

La ventana de AXIS contiene los siguientes elementos:

- Un área de visualización que muestra uno de los siguientes elementos:
  - una vista previa del archivo cargado (en este caso, *axis.ngc*), así como la ubicación actual del *punto controlado* de la máquina. Más adelante, esta área mostrará el camino por el que la máquina se ha movido, llamado *backplot*
  - una lector grande que muestra la posición actual y todos los offsets.
- Una barra de menús y una barra de herramientas que le permiten realizar varias acciones
- Una pestaña de *control manual (F3)* - que te permite mover la máquina, encender o apagar el husillo y el refrigerante, si se incluye en el archivo ini.
- Una pestaña *MDI* - donde los programas con código G se pueden ingresar manualmente, una línea a la vez. También muestra los *Códigos G activos* que son los códigos G modales en vigor.
- *Feed Override* - que le permite escalar la velocidad de los movimientos programados. El máximo predeterminado es 120 % y se puede configurar a un valor diferente en el archivo ini. Consulte la Sección de visualización del archivo INI para más información.
- *Spindle Override* - que le permite escalar la velocidad del husillo hacia arriba o hacia abajo.
- *Jog Speed* - que te permite configurar la velocidad de jog dentro de los límites establecidos en el archivo ini. Ver la Sección de visualización del archivo INI para obtener más información.
- *Velocidad máxima* - que le permite restringir la velocidad máxima de todos los movimientos programados (excepto movimiento sincronizados del husillo).
- Una zona de visualización de texto que muestra el G-Code cargado.
- Una barra de estado que muestra el estado de la máquina. En la pantalla mostrada, la máquina está encendida, no tiene una herramienta insertada y la la posición mostrada es *Relativa* (mostrando todos los offsets) y *Actual* (mostrando la posición retroalimentada).

4.1.3.1. Elementos del menú

Algunos elementos del menú pueden estar en gris dependiendo de cómo tenga su archivo .ini configurado. Para más información sobre configuración vea el <<cha:ini-configuration,Capítulo INI.

MENÚ ARCHIVO

- *Abrir* . . . - abre un cuadro de diálogo estándar para abrir un archivo de código g para cargar en AXIS. Si ha configurado LinuxCNC para usar un programa de filtro, también puede abrirlo. Consulte la sección FILTRO de la configuración INI para más información.

- *Archivos recientes*- muestra una lista de los archivos abiertos recientemente.
- *Editar . . .* - abre el archivo de código G actual para editarlo si tienes un editor configurado en su archivo ini. Consulte la sección sección DISPLAY para obtener más información sobre la especificación del editor a usar.
- *Recargar*- vuelve a cargar el archivo de código g actual. Si lo edita, debe recargarlo para que los cambios se actualizen. Si detiene un archivo y quiere empezar desde el principio, vuelva a cargar el archivo. La recarga de la barra de herramientas es la misma qu la del menu.
- *Guardar gcode como . . .* - Guarda el archivo actual con un nuevo nombre.
- *Propiedades*- la suma de los movimientos rápidos y de avance. No tiene en cuenta modos de aceleración, fusion o ruta para que el tiempo reportado nunca sea menor que el tiempo de ejecución real.
- *Editar tabla de herramientas . . .* - Igual que Editar si ha definido un editor. Puede abrir la tabla de herramientas y editarla.
- *Recargar tabla de herramientas*: después de editar la tabla de herramientas, debe volve a cargarla.
- *Editor Ladder* - Si has cargado Classic Ladder puedes editarlo desde aquí. Vea el capítulo <<cha:classicladder,Classicladder para más información.
- *Salir* - Termina la sesión actual de LinuxCNC.

MENÚ DE MÁQUINA

- *Toggle Emergency Stop F1*- cambia el estado de la parada de emergencia.
- *Toggle Machine Power F2* - Cambiar el estado de encendido de la máquina si la parada de emergencia no está encendida.
- *Ejecutar programa*- Ejecuta el programa actualmente cargado desde el principio.
- *Ejecutar desde la línea seleccionada* - seleccione la línea desde la que desea comenzar. Use con precaución ya que esto moverá la herramienta a la posición esperada en la línea y luego se ejecutará el resto del código.

aviso
No use *Ejecutar desde la línea seleccionada* si su programa de código g contiene subrutinas.

- *Step* - Un solo paso a través de un programa.
- *Pausa* - Pausa un programa.
- *Resume* - reanudar la ejecución de una pausa.
- *Stop* - Detiene un programa en ejecución. Cuando se selecciona ejecutar después de una parada, el programa comenzará desde el principio.
- *Stop en M1* - Si se alcanza un M1, y esto esta activo, la ejecución del programa parará en la línea M1. Presione Resume para continuar.
- *Saltar líneas con "/"* - Si una línea comienza con / y esto está activo, la línea se saltará.
- *Borrar historial de MDI*: borra la ventana del historial de MDI.
- *Copiar desde el historial de MDI*: copia el historial MDI al portapapeles
- *Pegar al historial de MDI* - Pegar desde el portapapeles a la ventana del historial MDI
- *Calibración*: inicia el asistente de calibración (emccalib.tcl). La calibración lee el archivo HAL y para cada *setp* que usa una variable del archivo ini que se encuentra en las secciones [AXIS_L], [JOINT_N] o [TUNE], crea una entrada que puede ser editada y probada.

- *Mostrar configuración HAL*- abre la ventana de configuración HAL donde puede monitorear componentes HAL, pines, parámetros, señales, funciones y subprocesos.
- *HAL Meter*- abre una ventana donde puede monitorear un solo Pin HAL, señal o Parámetro.
- *HAL Scope*- abre un osciloscopio virtual que permite seguir valores HAL en función del tiempo.
- *Mostrar estado de LinuxCNC*- abre una ventana que muestra el estado de LinuxCNC.
- *Establecer nivel de depuración*- abre una ventana donde se pueden ver los niveles de depuración y se pueden configurar algunos.
- *Homing* - home uno o todos los ejes.
- *Unhoming* - Deshacer home de uno o todos los ejes.
- *Sistema de coordenadas cero*- establece todos los offsets a cero en el sistema de coordenadas elegido.

- *Tool touch off to workpiece* - Al realizar Touch Off, el valor ingresado es relativo al sistema de coordenadas de la pieza actual (*G5x*), modificado por el offset del eje (*G92*). Cuando se completa el Touch Off, la coordenada relativa para el eje elegido se convertirá en el valor ingresado. Consulte <<gcode:g10-l10,G10 L10 en el capítulo de código G.
- *Tool touch off to fixture* - Al realizar Touch Off, el valor ingresado es relativo al noveno (*G59.3*) sistema de coordenadas, con el offset del eje (*G92*) ignorado. Esto es útil cuando hay un accesorio para Tool touch off en una ubicación fija en la máquina, con el noveno (*G59.3*) sistema de coordenadas establecido de tal manera que la punta de una herramienta de longitud cero esté en el origen del montaje cuando las coordenadas relativas son 0. Consulte <<gcode:g10-l11,G10 L11 en el capítulo de códigos G.

MENÚ VER

- *Vista superior* - la vista superior (o vista Z) muestra la previsualizacion del código G mirando en direccion del eje Z de positivo a negativo. Esta vista es la mejor para mirar el plano XY.
- *Vista superior girada* - la vista superior girada (o vista Z girada) también se muestra el código G mirando a lo largo del eje Z de positivo a negativo. Pero a veces es conveniente mostrar los ejes X e Y girados 90 grados para ajustarse al mostrar mejor Este punto de vista es también mejor para mirar X y Y.
- *Vista lateral* - la vista lateral (o vista X) muestra el código G mirando hacia adelante El eje X de positivo a negativo. Esta vista es mejor para mirar a Y & Z.
- *Vista frontal* - la vista frontal (o vista en Y) muestra el código G mirando hacia adelante El eje Y de negativo a positivo. Esta vista es mejor para mirar X y Z.
- *Vista en perspectiva* - la vista en perspectiva (o vista P) muestra el código G mirando la pieza desde un punto de vista ajustable, por defecto a X+, Y-, Z+. La posición es ajustable usando el mouse y el selector de arrastrar/rotar. Esta vista es una vista de compromiso, y si bien hace un buen trabajo al tratar de mostrar tres (¡hasta nueve!) ejes en una pantalla bidimensional, a menudo habrá alguna característica que es difícil de ver o que requiere un cambio en el punto de vista. Esta vista es la mejor cuando le gustaría ver los tres (a nueve) ejes a la vez.

Punto de vista

El menú de selección de pantalla AXIS *Ver* se refiere a las vistas *Superior*, *Delantera* y *Lateral*. Estos términos son correctos si la máquina CNC tiene su eje Z vertical, con Z positivo hacia arriba. Esto es cierto para las fresadoras verticales, que es probablemente la aplicación más popular, y también es cierto para casi todas las máquinas EDM, e incluso tornos verticales de torreta, donde la pieza gira debajo de la herramienta.

Los términos *Superior*, *Delantera* y *Lateral* pueden ser confusos, sin embargo, en otras máquinas CNC, como un torno estándar, donde el eje Z es horizontal, o una fresa horizontal (de nuevo donde el eje Z es horizontal) o incluso un torno vertical de torreta invertido, donde la pieza gira sobre la herramienta y la dirección positiva del eje Z es hacia abajo!

Solo recuerde que el eje Z positivo está (casi) siempre *alejandose* de la pieza. Familiarícese con el diseño de su máquina e interprete la pantalla según sea necesario.

- *Mostrar pulgadas* - establece la escala de la pantalla AXIS en pulgadas.
- *Mostrar MM*: establece la escala de la pantalla AXIS en milímetros.
- *Mostrar programa* - la vista previa del programa de código G cargado puede ser completamente desactivada si lo desea.
- *Mostrar Rapidos* - la vista previa del programa de código G cargado siempre mostrará el avance (G1, G2, G3) en color blanco. Pero la vision de movimientos rápidos (G0) ,en cian, se puede desactivar si se desea.
- *Fusion alfa* - esta opción hace que la vista previa de programas complejos sea más fácil de ver, pero puede hacer que la vista previa se muestre más lentamente.
- *Mostrar Plot en vivo* - El resaltado de las rutas de avance (G1, G2, G3) a medida que la herramienta se mueve se puede desactivar si se desea.
- *Mostrar herramienta* - la visualización del cono/cilindro de la herramienta se puede desactivar si se desea.
- *Mostrar extensiones* - la visualización de las extensiones (recorrido máximo en cada dirección del eje) del programa de código G cargado se puede desactivar si se desea.
- *Mostrar Offsets* - Se puede mostrar la ubicación de origen del offset del montaje seleccionado (G54-G59.3) como un conjunto de tres líneas ortogonales, roja, azul y verde. Esta visualización de origen de offset (o punto cero) se puede desactivar si se desea.
- *Mostrar límites de máquina* - los límites máximos de desplazamiento de la máquina para cada eje, según lo establecido en el archivo ini, se muestran como una caja rectangular dibujada en líneas discontinuas rojas. Esto es útil cuando se carga un nuevo programa de código G, o cuando se comprueba si se necesitaría mucho offset del montaje para llevar el programa de código G dentro de los límites de recorrido de su máquina. Puede apagarse si no es necesario.
- *Mostrar Velocidad* - Una visualización de la velocidad a veces es útil para ver qué tan cerca está funcionando su máquina de la velocidad de diseño. Puede ser desactivado si se desea.
- *Mostrar distancia a recorrer* - Distancia a recorrer es un elemento muy útil que conocer cuando se ejecuta un programa de código G desconocido por primera vez. En combinación con los controles de anulación de velocidad rápida y de avance, se puede evitar el daño a herramientas y/o a la máquina. Una vez que el programa de código G se ha depurado y se está ejecutando sin problemas, la pantalla Distancia a ir se puede desactivar si se desea.
- *Limpiar Plot en vivo* - a medida que la herramienta se desplaza en la pantalla Axis, se resalta la ruta del código G. Para repetir el programa, o para ver mejor un área de interés, las rutas previamente resaltadas se pueden borrar.
- *Mostrar posición ordenada*: esta es la posición a la que intentará ir LinuxCNC. Una vez que el movimiento se ha detenido, esta es la posición que intentará mantener LinuxCNC.
- *Mostrar posición actual*: la posición real es la posición medida, leída desde los codificadores o desde el sistema simulado por los generadores de pasos. Esto puede diferir ligeramente de la posición ordenada por muchas razones, incluyendo afinacion del PID, restricciones físicas, o cuantización de la posición.
- *Mostrar posición de la máquina*: esta es la posición en coordenadas sin compensación, según lo establecido por Homing.
- *Mostrar posición relativa*: esta es la posición de la máquina modificada por las compensaciones *G5x*, *G92* y *G43*. .Menú de ayuda
- *Acerca de Axis* - Todos sabemos lo que es esto.
- *Referencia rápida*: muestra las teclas de método abreviado del teclado.

4.1.3.2. Botones de la barra de herramientas

De izquierda a derecha en la pantalla de Axis, los botones de la barra de herramientas (atajos de teclado mostrados [entre corchetes]) son:

- Stop de Emergencia [F1] (también llamado E-Stop)
- Encendido de Maquina [F2]
- Abrir archivo de código G [O]
- Recargar archivo actual [Ctrl-R]
- Comenzar a ejecutar el archivo actual [R]
- Ejecutar línea siguiente [T]
- Pausar ejecución [P] Reanudar ejecución [S]
- Detener la ejecución del programa [ESC]
- Saltar lineas con "/" [Alt-M- /]
- Pausa Opcional [Alt-M-1]
- Zoom (mas)
- Zoom (menos)
- Vista superior
- Vista superior girada
- Vista lateral
- Vista frontal
- Vista en perspectiva
- Alternar entre los modos de arrastrar/rotar [D]
- Limpiar backplot en vivo [Ctrl-K]

4.1.3.3. Área de visualización gráfica

Visualizacion de coordenadas En la esquina superior izquierda de la pantalla del programa está la visualizacion de las coordenadas de posicion para cada eje. A la derecha del número, un símbolo de origen que se muestra si el eje ha sido dotado de home.

Una símbolo de límite se muestra en el lado derecho del número de coordenada de posición, si el eje está en uno de sus interruptores de límite.

Para interpretar correctamente los números de coordenadas de posición, consulte el indicador *Posición:* en la barra de estado. Si la posición es *Máquina actual*, entonces el número mostrado está en el sistema de coordenadas de la máquina. Si se muestra *Relative Actual*, entonces el número mostrado está en la coordenada del sistema con desplazamiento. Cuando las coordenadas mostradas son relativas y se ha establecido un desplazamiento, la pantalla incluirá un marcador origen de máquina cian.

Si la posición es *Comandada*, entonces la coordenada exacta dada en un comando de código G es la mostrada. Si es *Actual*, entonces es la posición real a la que la máquina se ha movido. Estos valores pueden ser diferentes de la posición ordenada debido al error de seguimiento, banda muerta, resolución del codificador o tamaño de paso. Por ejemplo, si ordena un movimiento a X 0.0033 en su fresadora, pero el paso de su motor paso a paso o su resolucion de encoder es 0.00125, la posición *Comandada* podría ser 0.0033, pero la posición *Actual* será 0.0025 (2 pasos) o 0.00375 (3 pasos).

Plot de Vista Previa Cuando se carga un archivo,se muestra una vista previa en el área de visualización. Los movimientos rápidos (como los producidos por el comando *G0*) se muestran como lineas cian. Los movimientos a velocidad de avance (como los producidos por el comando *G1*) se muestran como líneas blancas sólidas. Dwells (como los producidos por el comando *G4*) se muestran como pequeñas marcas *X* rosadas.

Movimientos G0 (rápido), antes de un movimiento de alimentación no se mostrará en el plot de vista previa. Los movimientos rápidos después de una T<n> (Cambio de herramienta) no se mostrarán en la vista previa hasta después del primer movimiento de alimentación. Para desactivar cualquiera de estas funciones, programe un G1 sin ningún movimiento antes de los movimientos G0.

Dimensiones (fisicas necesarias) del programa Se muestran las *dimensiones* resultantes del programa en cada eje. En los extremos, se indican los valores de coordenadas mínimo y máximo. En el medio, se muestra la diferencia entre las coordenadas, o dimension.

Cuando algunas coordenadas exceden los *límites soft* del archivo .ini, la dimensión *culpable* se muestra en un color diferente y está encerrada en un cuadro. En la figura de abajo se sobrepasa el límite soft máximo en el eje X, que se indica en el cuadro que rodea el valor de la coordenada. El mínimo recorrido X del programa es -1.95, el recorrido máximo de X es 1.88, y el programa requiere 3,83 pulgadas de recorrido X. Para que el movimiento programado esté dentro del recorrido de la máquina en este caso, haga jog a la izquierda y vuelva a hacer Touch Off X.

Herramienta Cono Cuando no se ha cargado ninguna herramienta, la ubicación de la punta de la herramienta esta indicada por un *cono de herramienta*. La *herramienta cono* no proporciona orientación sobre la forma, longitud, o radio de una herramienta real.

Cuando se carga una herramienta (por ejemplo, con el comando MDI *T1 M6*), el cono cambia a un cilindro que muestra el diámetro de la herramienta, dado en el archivo de tabla de herramientas.

Backplot Cuando la máquina se mueve, deja un rastro en pantalla llamado backplot. El color de la línea indica el tipo de movimiento: Amarillo para jogs, verde claro para movimientos rápidos, rojo para movimientos rectos a velocidad de avance y magenta para movimientos circulares a velocidad de avance.

Cuadrícula Axis puede, opcionalmente, mostrar una cuadrícula en las vistas ortogonales. Habilite o deshabilite la cuadrícula usando *Cuadrícula* en el menu *Ver*. Cuando esta habilitada, la cuadrícula se muestra en las vistas superior y superior girada. Cuando el sistema de coordenadas no esta girado, la cuadrícula se muestra también en las vistas frontal y lateral. Los preajustes en el menú *Grid* están controlados por el elemento del archivo ini `[DISPLAY]GRIDS`. Si no se especifica, el valor predeterminado es `10mm 20mm 50mm 100mm 1in 2in 5in 10in`.

Especificar una cuadrícula muy pequeña puede disminuir el rendimiento.

Interaccion Al hacer clic izquierdo en una parte del plot de vista previa,la línea será resaltada tanto en las pantallas gráficas como en las de texto. Al hacer clic izquierdo en un área vacía, se eliminará el resaltado.

Al arrastrar con el botón izquierdo del ratón presionado, la trama de vista previa se desplazará (panorámico).

Al arrastrar con Mayús y el botón izquierdo del ratón presionado, o arrastrando con la rueda del ratón presionada, La trama de vista previa se rotará. Cuando una línea está resaltada, el centro de rotación es el centro de la línea. De lo contrario, el centro de rotación es el centro de todo el plot.

Al girar la rueda del ratón, o arrastrando con el botón derecho del ratón presionado, o arrastrando con Control y presionando el botón izquierdo del ratón, el plot de vista previa se acercará o alejará.

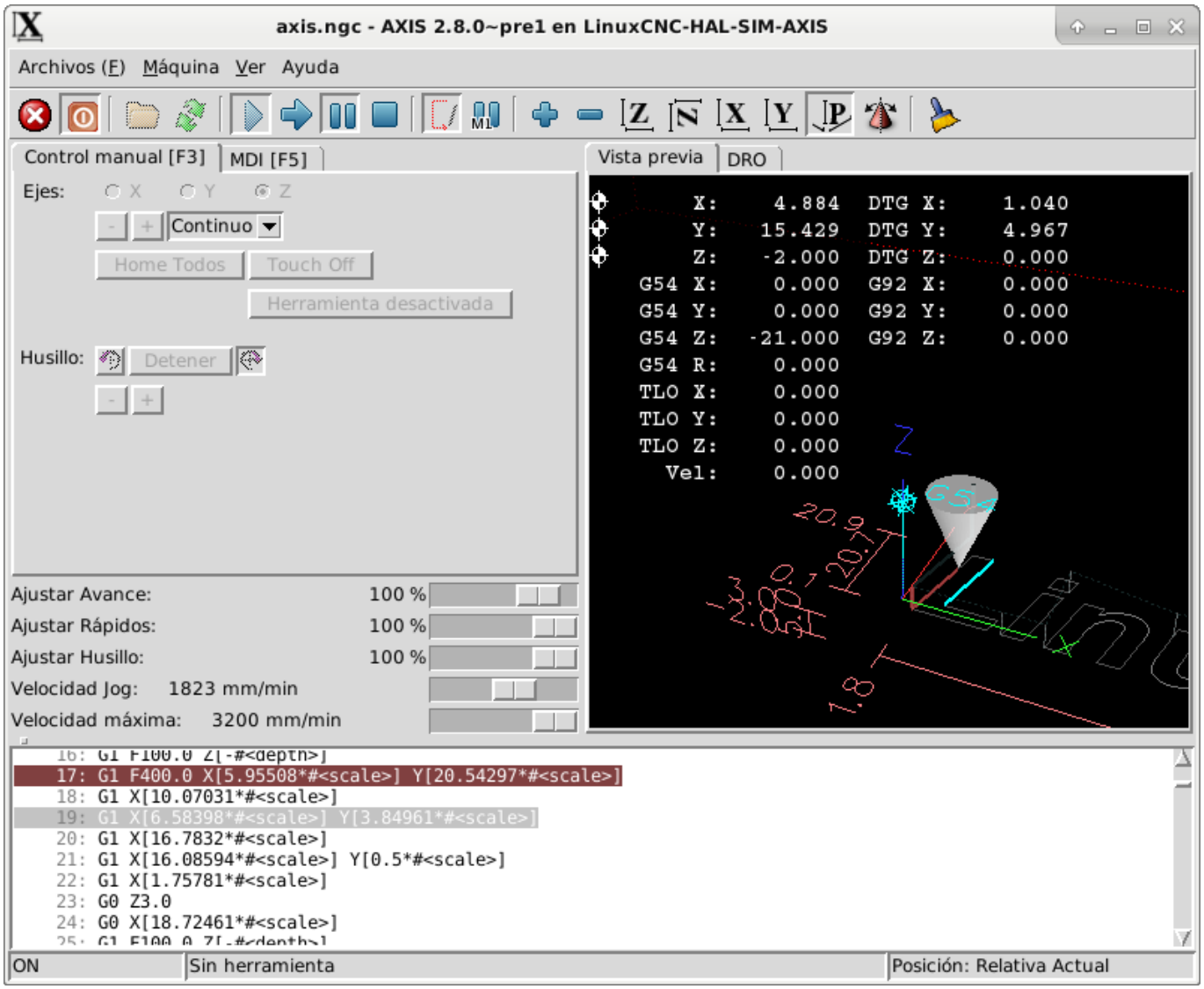
Al hacer clic en uno de los iconos de *Vista predefinida*, o presionando *V*, se pueden seleccionar varias vistas preestablecidas.

4.1.3.4. Área de visualización de texto

Al hacer clic izquierdo en una línea del programa, la línea se resaltará, tanto en las pantallas gráficas como en las de texto.

Cuando el programa se está ejecutando, la línea que se está ejecutando actualmente es resaltada en rojo. Si el usuario no ha seleccionado ninguna línea, la pantalla de texto se desplazará automáticamente para mostrar la línea actual.

Líneas actuales y seleccionadas



4.1.3.5. Control manual

Mientras la máquina está encendida pero no ejecuta un programa, los elementos de la pestaña *Control Manual* se pueden utilizar para mover la máquina o controlar su husillo y el refrigerante.

Cuando la máquina no está encendida, o cuando se está ejecutando un programa, los controles manuales no están disponibles.

Muchos de los elementos descritos a continuación no son útiles en todas las máquinas. Cuando AXIS detecta que un pin en particular no está conectado en HAL, se elimina el elemento correspondiente en la pestaña Control manual. Por ejemplo, si el pin HAL *motion.spindle-brake* no está conectado, entonces el botón *Freno* no aparecerá en la pantalla. Si la variable de entorno *AXIS_NO_AUTOCONFIGURE* está establecida, este comportamiento está deshabilitado y todos los elementos aparecerán.

El grupo Axis *Axis* le permite mover manualmente la máquina. Esta acción se conoce como *jogging*. Primero, seleccione el eje a mover haciendo clic en él. Luego, haga clic y manténga presionado el botón + o - dependiendo de la dirección de movimiento deseada. Los primeros cuatro ejes también pueden ser movidos por las teclas de flecha (X e Y), Teclas PAGE UP y PAGE DOWN (Z), y las teclas [and] (A).

Si se selecciona *Continuo*, el movimiento continuará mientras se presiona el botón o la tecla. Si se selecciona otro valor, la máquina se moverá exactamente la distancia mostrada cada vez que se hace clic en el botón o se presiona la tecla. Por defecto, los valores disponibles son *0.1000, 0.0100, 0.0010, 0.0001*

Consulte la <<sec:display-section, sección DISPLAY para obtener más información sobre la configuración los incrementos.

Homing Si la máquina dispone de micros de home y una secuencia definida para homing de todos los ejes, en el botón mostrara *Home All*. El botón *Home All* o las teclas Ctrl-HOME llevará a home todos los ejes utilizando la secuencia. La tecla HOME llevará a home el eje actual, incluso si esta definida una secuencia de inicio.

Si su máquina tiene interruptores home y no se define una secuencia de inicio o no todos los ejes tienen una secuencia home, el botón mostrara *Home* y solo llevara a home el eje seleccionado. Cada eje debe ser seleccionado y llevado a home por separado.

Si su máquina no tiene interruptores home definidos en la configuración, el botón *Home* establecerá la posición actual del eje seleccionado como la posición absoluta 0 para ese eje y activara el bit *is-homed* para ese eje.

Consulte el <<cha:homing-configuration,Capítulo de configuración de Homing para obtener más información.

Touch Off Al presionar *Touch Off* o la tecla END, el *offset G5x* para el el eje actual se cambia para que el valor del eje actual sea el valor específicado. Las expresiones se pueden ingresar usando las reglas para los programas rs274ngc, excepto que las variables no pueden ser referidas. El valor resultante se muestra como un número.

Figura 4.2: Touch Off

Consulte también las opciones *Tool touch off to workpiece* y *Tool touch off to fixture* en el menú Machine.

Anulacion de Límites Pulsando Anulacion de Límites, se permitirá jog temporalmente en la máquina mas alla de un final de carrera físico. Esta casilla solo está disponible cuando se dispara un interruptor de límite. La anulación desaparece después de un jog. Si El eje está configurado con interruptores de límite positivo y negativo separados, LinuxCNC permitirá el jog solo en la dirección correcta. *La anulación de límites no permite un jog más allá de un límite soft. La única manera de deshabilitar un límite soft en un eje es con unhome.*

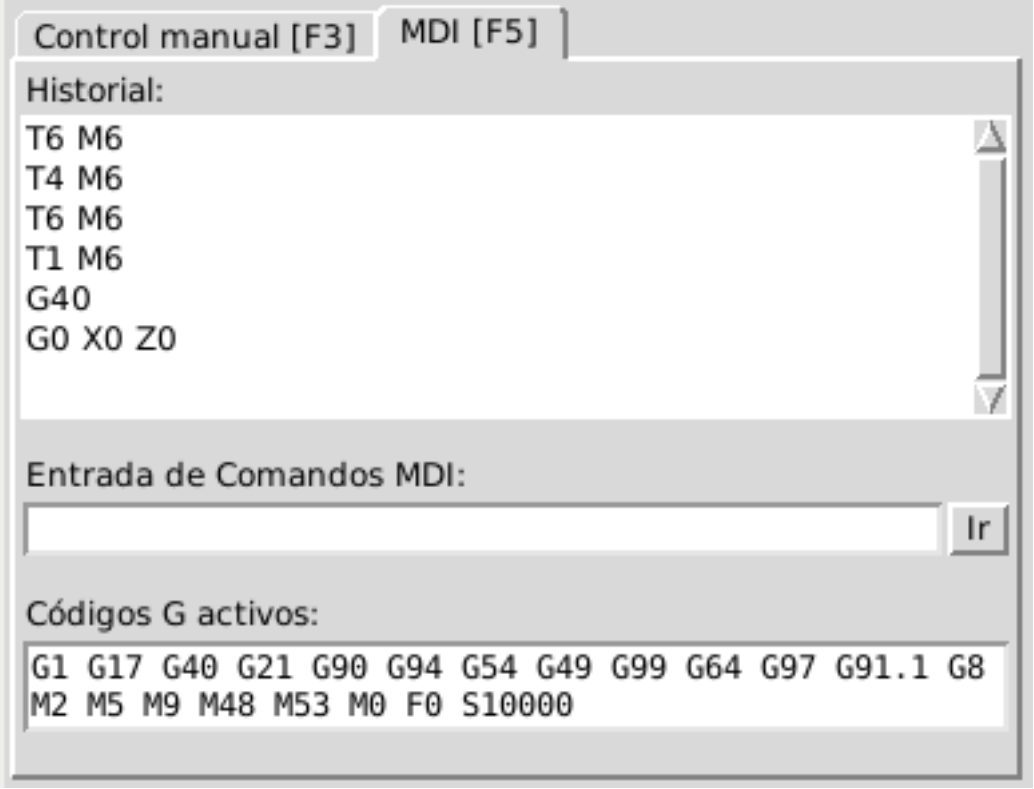
El grupo del husillo. Los botones de la primera fila seleccionan la dirección de giro del husillo; en sentido contrario a las agujas del reloj, detenido y en el sentido de las agujas del reloj. El sentido antihorario solo aparece si el pin *motion.spindle-reverse* está en el archivo HAL (por ejemlo, *net trick-axis motion.spindle-reverse*). Los botones en la siguiente fila aumentan o disminuyen la velocidad de rotación. La casilla de verificación en la tercera fila permite que el freno del husillo sea accionado o liberado. Dependiendo de la configuración de su máquina,pueden no aparecer todos los elementos en este grupo. Presionando el botón de arranque del husillo se establece la velocidad *S* en 1.

El grupo de refrigerante Dos botones permiten encender los refrigerantes *Mist* y *Flood* y apagarlos. Dependiendo de la configuración de su máquina, puede no aparecer todos los elementos en este grupo.

4.1.3.6. MDI

MDI permite que los comandos de código G se ingresen manualmente. Cuando la máquina no está encendida, o cuando un programa está en ejecución, los controles MDI no están disponibles.

La pestaña MDI



- *Historial* - Muestra los comandos MDI que se han escrito anteriormente en esta sesión.
- *Comando MDI* - Esto le permite ingresar un comando de código g para ser ejecutado. Ejecute el comando pulsando Intro o haciendo clic en *Go*.
- *Códigos G activos* - Se muestran los *códigos modales* que están activos en el intérprete. Por ejemplo, *G54* indica que el *offset G54* se aplica a todas las coordenadas que se introduzcan. En modo Auto los G-Códigos Activos representan los códigos después la lectura por el intérprete.

4.1.3.7. Porcentaje de alimentacion

Al mover este control deslizante (feed override), se puede modificar la velocidad de alimentación programada. Por ejemplo, si un programa solicita *F60* y el control deslizante se establece en 120 %, entonces la velocidad de alimentación resultante será 72.

4.1.3.8. Porcentaje de velocidad del husillo

Al mover este control deslizante (Spindle Speed Override), la velocidad del husillo programada puede ser modificada. Por ejemplo, si un programa solicita S8000 y el control deslizante es establecido en 80 %, entonces la velocidad del husillo resultante será 6400. Este elemento solo aparece cuando el pin HAL *motion.spindle-speed-out* está conectado.

4.1.3.9. Velocidad de Jog

Al mover este control deslizante, se puede modificar la velocidad de joging. Por ejemplo, si el control deslizante se establece en 1 pulgada/min, entonces un avance de .01 pulgadas tardara aproximadamente .6 segundos, o 1/100 de minuto. Cerca del lado izquierdo (jog lento) los valores están espaciados muy cerca, mientras que cerca del lado derecho (jogs rápidos) están espaciados mucho más separados, permitiendo una amplia gama de velocidades de jog con control fino cuando sea importante.

En las máquinas con un eje giratorio, se muestra un segundo control deslizante de velocidad de desplazamiento. Este control deslizante establece la velocidad de desplazamiento de los ejes giratorios (A, B y C).

4.1.3.10. Velocidad máxima

Al mover este control deslizante, se puede establecer la velocidad máxima. Esto limita la velocidad máxima para todos los movimientos programados, excepto en movimientos sincronizados con el husillo.

4.1.4. Controles del teclado

Casi todas las acciones en AXIS se pueden realizar con el teclado. La lista completa de atajos de teclado se puede encontrar en AXIS Quick Reference, que se puede mostrar seleccionando Ayuda> Quick Reference. Muchos de los accesos directos no están disponibles cuando se está en modo MDI.

Teclas de porcentaje de alimentación

Las teclas de anulación de la alimentación se comportan de manera diferente cuando están en modo manual.

---

nota
En el teclado *Español de España*, el simbolo " ’ " se refiere al signo de acentuacion junto a la tecla "P". Este signo precisa doble pulsacion de dicha tecla.

Las teclas ’,1,2,3,4,5,6,7 y 8 seleccionarán un eje si está programado. Si tiene 3 eje ’ seleccionará el eje 0, 1 seleccionará el eje 1, y 2 seleccionará eje 2. El resto de las teclas numéricas establecerán el porcentaje de alimentación. Al ejecutar un programa, ’,1,2,3,4,5,6,7,8,9,0 establecerán el porcentaje de alimentación a 0% - 100%.

Los métodos abreviados de teclado más utilizados se muestran en la siguiente tabla

Cuadro 4.1: Atajos de teclado más comunes

Pulsar tecla	Acción tomada	Modo
F1	Stop de emergencia	Todos
F2	Encender/apagar la máquina	Todos
`, 1 .. 9, 0	Establecer porcentaje alimentación de 0% a 100%	Varía
X, `	Activar primer eje	Manual
Y, 1	Activar segundo eje	Manual
Z, 2	Activar tercer eje	Manual
A, 3	Activar cuarto eje	Manual
I	Seleccion incremento jog	Manual
C	Jog continuo	Manual
Control-Inicio	Realizar secuencia homing	Manual
Fin	Touch off: offset G5x para el eje activo	Manual
Izquierda, Derecha	Jog primer eje	Manual
Arriba, Abajo	Jog segundo eje	Manual
Pg Arriba, Pg Dn	Jog tercer eje	Manual
[,]	Jog cuarto eje	Manual
O	Abrir archivo	Manual
Control-R	Recargar archivo	Manual
R	Ejecutar archivo	Manual
P	Pausar ejecución	Auto
S	Reanudar ejecución	Auto
ESC	Detener ejecucion	Auto
Control-K	Borrar backplot	Manual/auto
V	Ciclo entre vistas preestablecidas	Manual/auto
Shift-izda, dcha	Eje X, rapido	Manual
Shift-Up, Abajo	Eje Y, rapido	Manual

Cuadro 4.1: (continued)

Pulsar tecla	Acción tomada	Modo
Shift-PgUp, PgDn	Eje Z, rápido	Manual
@	conmutar actual/comandado	Todos
#	conmutar relativo/máquina	Todos

4.1.5. Mostrar estado de LinuxCNC (linuxcnctop)

AXIS incluye un programa llamado *linuxcnctop* que muestra algunos de los detalles del estado de LinuxCNC. Puedes ejecutar este programa mediante Maquina > Mostrar estado de LinuxCNC

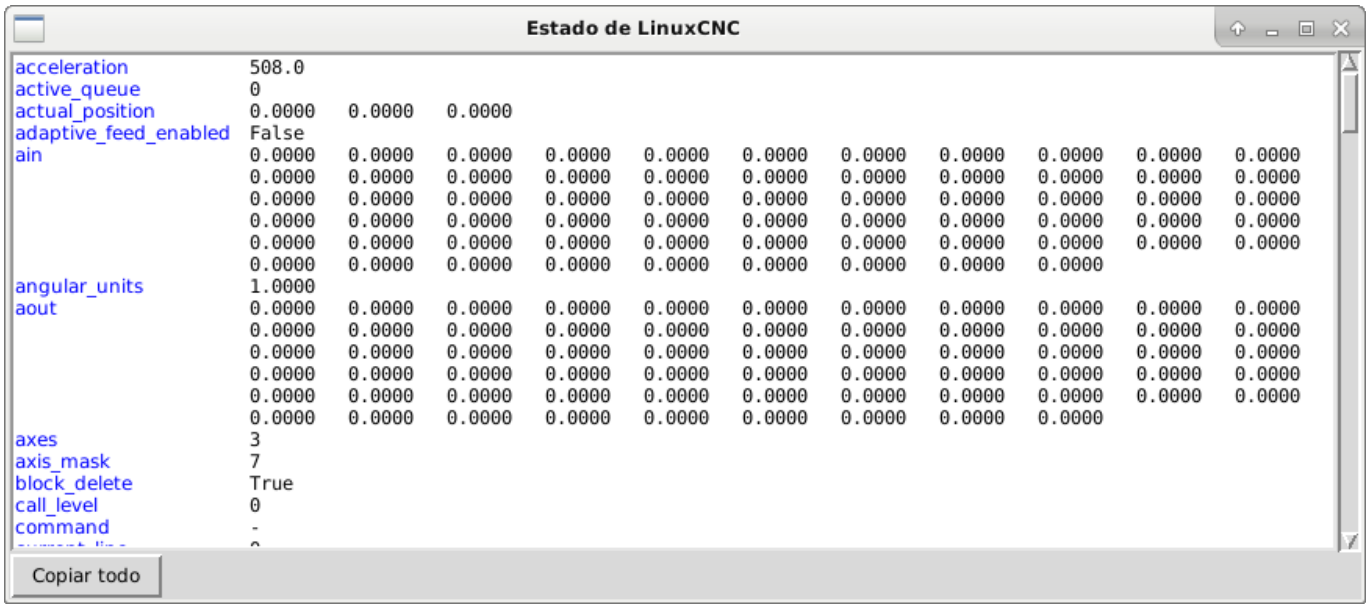


Figura 4.3: Ventana de estado de LinuxCNC

El nombre de cada elemento se muestra en la columna izquierda. El valor actual se muestra en la columna derecha. Si el valor ha cambiado recientemente, se muestra sobre un fondo rojo.

4.1.6. Interfaz MDI

AXIS incluye un programa llamado `mdi` que permite la entrada en modo texto de comandos MDI a una sesión de LinuxCNC en ejecución. Puede ejecutar este programa abriendo un terminal y escribiendo

```
mdi
```

Una vez que se está ejecutando, muestra el mensaje *MDI*>. Cuando se ingresa una linea en blanco, se muestra la posición actual de la máquina. Cuando se ingresa un comando, se envía a LinuxCNC para ser ejecutado. Para salir de mdi, pulse Ctrl-c en el terminal.

Esta es una sesión de muestra de mdi.

```
$ mdi
MDI>
(0.0, 0.0, 0.0, 0.0, 0.0, 0.0)
```

```
MDI> G1 F5 X1
MDI>
(0.5928500000000374, 0.0, 0.0, 0.0, 0.0, 0.0)
MDI>
(1.0000000000000639, 0.0, 0.0, 0.0, 0.0, 0.0)
```

4.1.7. axis-remote

AXIS incluye un programa llamado *axis-remote* que puede enviar ciertos comandos a un AXIS en ejecución. Los comandos disponibles se muestran ejecutando *help-axis* y ayuda a verificar si AXIS se está ejecutando (*--ping*), cargando un archivo por nombre, recargando el archivo cargado actualmente archivo (*--reload*), y hacer que AXIS salga (*--quit*).

4.1.8. Cambio de herramienta manual

LinuxCNC incluye un componente HAL de espacio de usuario llamado *hal_manualtoolchange*, que muestra una ventana que le indica qué herramienta se espera cuando se emite el comando *M6*. Después de presionar el botón OK, la ejecución del programa continuará.

El componente hal_manualtoolchange incluye un pin hal para un botón que se puede conectar a un botón físico para completar el cambio de herramienta y eliminar el indicador de ventana (hal_manualtoolchange.change_button).

El archivo de configuración de HAL *configs /sim/axis_manualtoolchange.hal* muestra los comandos HAL necesarios para usar este componente.

hal_manualtoolchange se puede usar incluso cuando AXIS no se usa como GUI. Este componente es más útil si tiene herramientas predefinidas y usa la tabla de herramientas

---

nota
Nota importante: los rápidos no se mostrarán en la vista previa después de emitir un T<n> hasta el siguiente movimiento de alimentación después de M6. Esto puede ser muy confuso para la mayoría de los usuarios. Para desactivar esta función para el cambio de herramienta actual, programe un G1 sin movimiento después de T<n>.

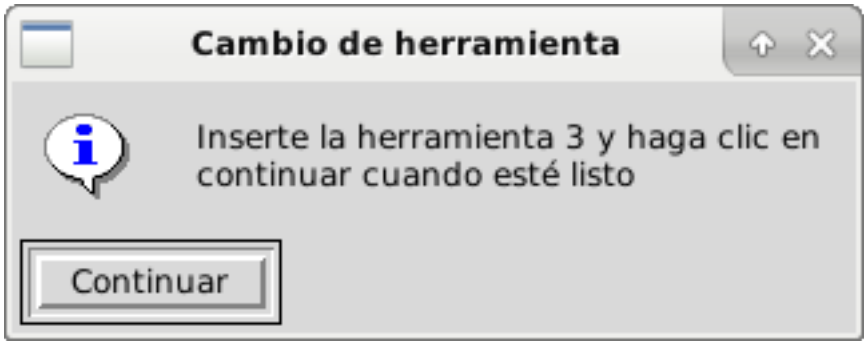


Figura 4.4: Ventana de cambio de herramientas manual

4.1.9. Módulos de Python

AXIS incluye varios módulos Python que pueden ser útiles para otros. Para más información sobre uno de estos módulos, use *pydoc <nombre del módulo>* o lea el código fuente. Estos módulos incluyen:

- *emc* proporciona acceso a los canales de comando, estado y error de LinuxCNC

- *gcode* proporciona acceso al intérprete rs274ngc
- *rs274* proporciona herramientas adicionales para trabajar con archivos rs274ngc
- *hal* permite la creación de componentes HAL de espacio de usuario escritos en Python
- *_togl* proporciona un widget OpenGL que puede usarse en aplicaciones Tkinter
- *minigl* proporciona acceso al subconjunto de OpenGL utilizado por AXIS

Para utilizar estos módulos en sus propios scripts, debe asegurarse de que el directorio donde residen está en la ruta de módulos de Python. Cuando se ejecuta una versión instalada de LinuxCNC, esto debería suceder automáticamente. Cuando ejecutando una RIP, esto se puede hacer mediante el guion *scripts/rip-environment*.

4.1.10. Usando AXIS en el modo Torno

Incluyendo la línea *LATHE = 1* en la sección [DISPLAY] del archivo ini, AXIS selecciona el modo de torno. El eje *Y* no se muestra en las lecturas de coordenadas, la vista se cambia mostrando el eje Z extendido hacia la derecha y el eje X que se extiende hacia en la parte inferior de la pantalla. Varios controles (como los de vistas preestablecidas) se eliminan. Se reemplazan las lecturas de coordenadas para X con diámetro y radio.

Al presionar *V* se hace zoom para mostrar el archivo completo, si hay uno cargado.

En el modo de torno, se muestra la forma de la herramienta cargada (si existe).

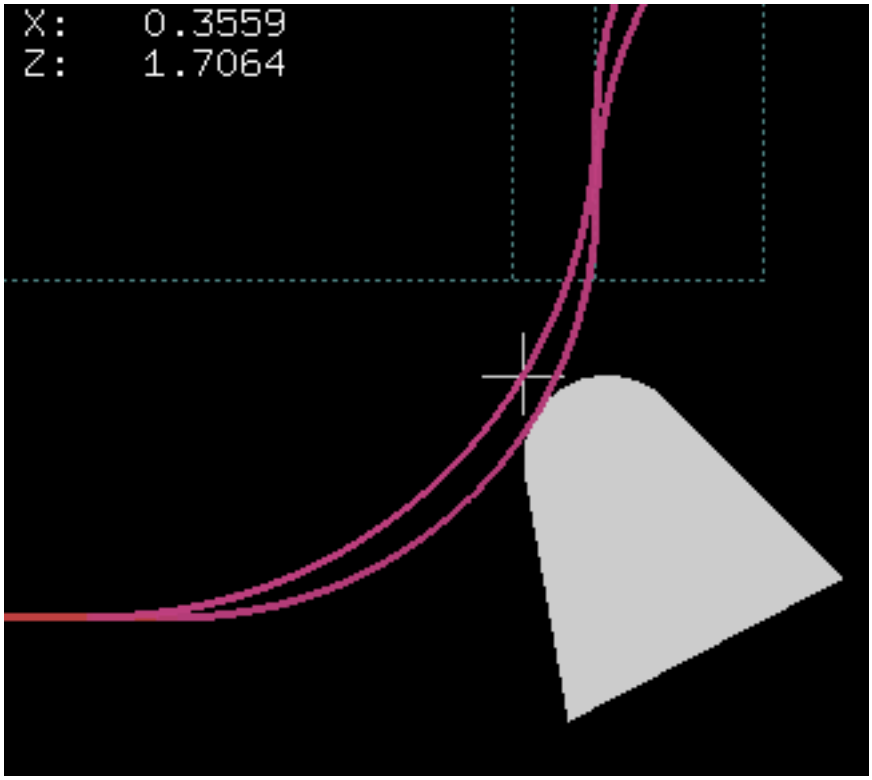


Figura 4.5: La forma de la herramienta del torno

4.1.11. Usando AXIS en el modo de corte de espuma

Incluyendo la línea *FOAM = 1* En la sección [DISPLAY] del archivo ini, AXIS selecciona el modo de corte de espuma. En la vista previa del programa, los movimientos XY se muestran en un plano, y los movimientos UV en otro. En el plot en vivo,

se dibujan líneas entre los puntos correspondientes en el plano XY y el plano UV. Los comentarios especiales (XY_Z_POS) y (UV_Z_POS) establecen las coordenadas Z de estos planos, que por defecto son 0 y 1,5 unidades de máquina.

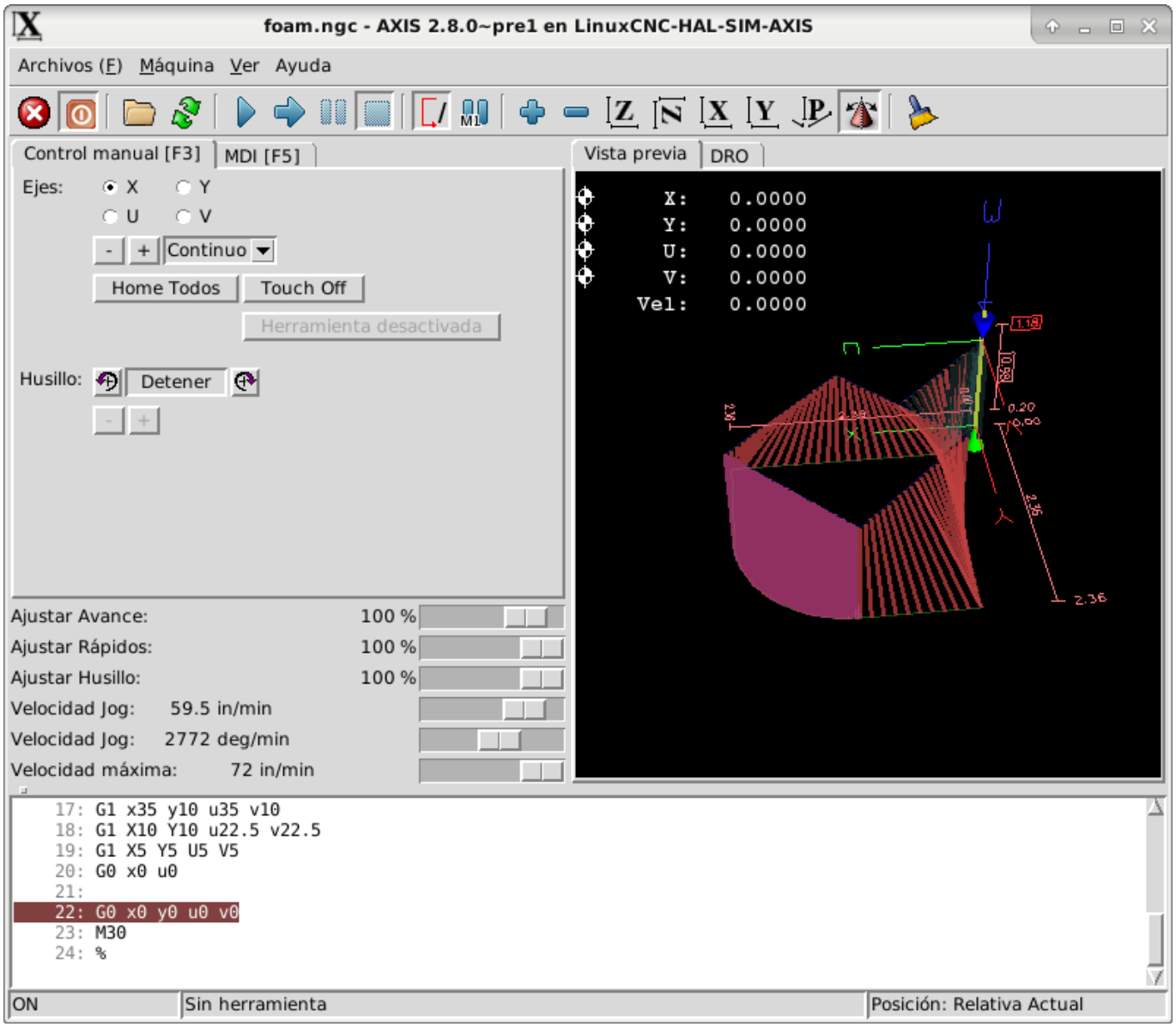


Figura 4.6: Modo de corte de espuma.

4.1.12. Configuración avanzada

Cuando se inicia AXIS, se crean los pines HAL para la GUI y se ejecuta el archivo HAL nombrado en *[HAL]POSTGUI_HALFILE* en el archivo ini. Solo se puede utilizar un archivo POSTGUI. Coloque todos los comandos HAL que se conecten a los pines HAL GUI en el archivo de postgui HAL.

Para obtener más información sobre la configuración del archivo ini que puede cambiar la forma en que AXIS trabaja, consulte la sección << sec:display-section,Seccion Display>> del capitulo de configuración INI.

4.1.12.1. Filtros de programa

AXIS tiene la capacidad de enviar archivos cargados a través de un *programa de filtro*. Este filtro puede realizar cualquier tarea deseada: algo tan simple como asegurarse el archivo termina con *M2*, o algo tan complicado como generar Código G de una imagen.

La sección *[FILTER]* del archivo ini controla cómo funcionan los filtros. Primero, para cada tipo de archivo, escriba una línea *PROGRAM_EXTENSION*. Luego, especifique el programa a ejecutar para cada tipo de archivo. Este programa recibe el nombre del archivo de entrada como su primer argumento, y debe escribir el código rs274ngc en la salida estándar. Esta salida es lo que se mostrará en el área de texto, se previsualizará en el área de visualización y sera ejecutado por LinuxCNC con *Run*. Las siguientes líneas agregan soporte para el convertidor de *imagen a gcode* incluido con LinuxCNC:

```
[FILTER]
PROGRAM_EXTENSION = .png,.gif Imagen de profundidad en escala de grises
png = image-to-gcode
gif = image-to-gcode
```

También es posible especificar un intérprete:

```
PROGRAM_EXTENSION = .py Script Python
py = python
```

De esta manera, cualquier script de Python se puede abrir, y su salida es tratada como g-code. Un ejemplo de este script está disponible en *nc_files/holecircle.py*. Este script crea g-code para perforar una serie de agujeros a lo largo de la circunferencia de un círculo.

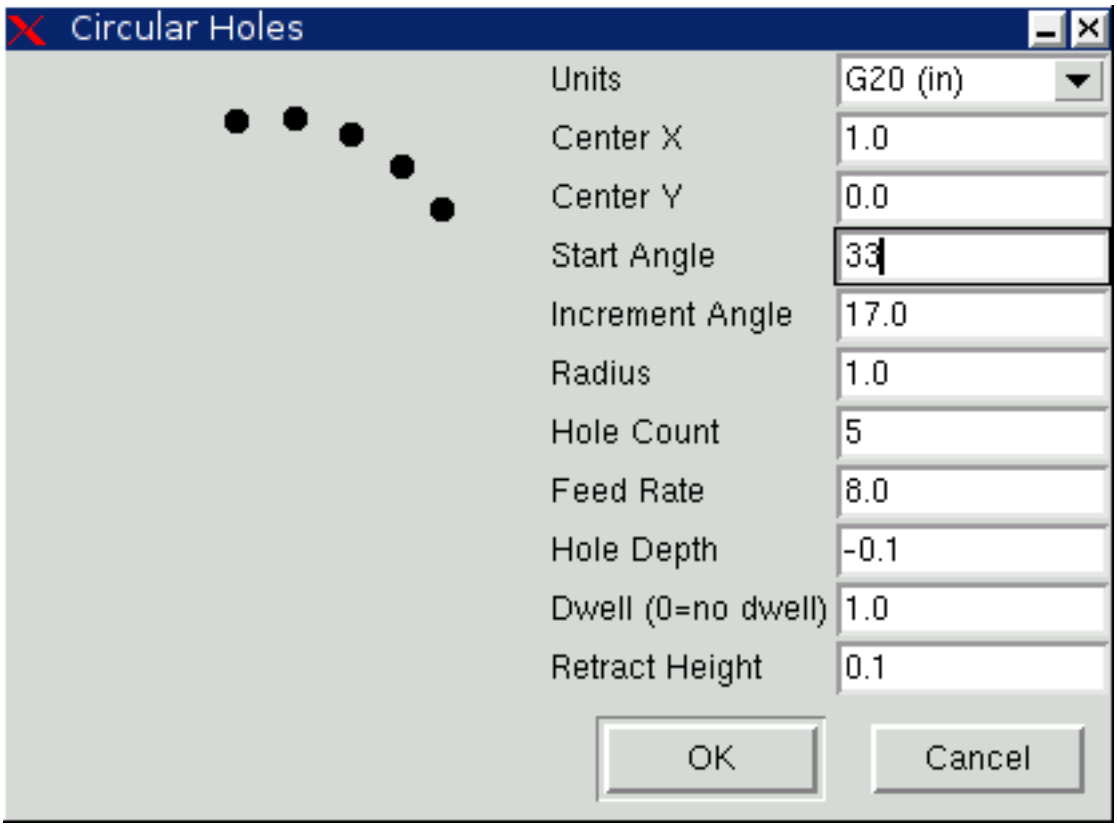


Figura 4.7: Agujeros circulares

Si la variable de entorno AXIS_PROGRESS_BAR está establecida, entonces las líneas escriben al stderr del formulario

```
FILTER_PROGRESS=%d
```

establecerá la barra de progreso de AXIS en el porcentaje dado. Esta característica debe ser utilizada por cualquier filtro que se ejecute durante mucho tiempo.

4.1.12.2. La base de datos de recursos X

Los colores de la mayoría de los elementos de la interfaz de usuario AXIS pueden ser personalizado a través de la base de datos de recursos X. El archivo de ejemplo *axis_light_background* cambia los colores de la ventana de backplot a *líneas oscuras en fondo blanco*, y también sirve como una referencia para elementos configurables en el área de visualización. El archivo de ejemplo *axis_big_dro* cambia la posición de lectura a una fuente de tamaño más grande. Para utilizar estos archivos:

```
xrdb -merge /usr/share/doc/emc2/axis_light_background

xrdb -merge /usr/share/doc/emc2/axis_big_dro
```

Para obtener información sobre los otros elementos que se pueden configurar en Tk aplicaciones Tk, ver las páginas del manual de Tk.

Dado que los entornos de escritorio modernos hacen algunas configuraciones automáticamente en la base de datos de recursos X que afectan adversamente a AXIS, estos ajustes son ignorados por defecto. Para hacer que los elementos de la base de datos de recursos X se anulen, los valores predeterminados de AXIS incluyen la siguiente línea en sus Recursos X:

```
    *Axis*optionLevel: widgetDefault
```

esto hace que las opciones integradas se creen en el nivel de opción *widgetDefault*, de modo que X Resources (que son nivel *userDefault*) puededan anularlas

4.1.12.3. ~/.axisrc

Si existe, el contenido de `~/.axisrc` se ejecuta como código fuente Python justo antes de la interfaz gráfica de usuario de AXIS. Los detalles de lo que se puede escribir en `~/.axisrc` están sujetos a cambios durante el ciclo de desarrollo.

Lo siguiente agrega Control-Q como método abreviado de teclado para Salir.

```
root_window.bind("<Control-q>", "destroy .")
help2.append(("Control-Q", "Quit"))
```

Lo siguiente detiene el cuadro de diálogo "¿Realmente desea salir?".

```
root_window.tk.call("wm","protocol",".","WM_DELETE_WINDOW","destroy .")
```

4.1.12.4. USER_COMMAND_FILE

Puede especificarse un archivo python específico de configuración con un archivo ini configurando *[DISLAY]USER_COMMAND_FILE=* Al igual que un archivo `~/.axisrc`, este archivo corre justo antes de que se muestre la GUI de AXIS. Este archivo es específico de una configuración de archivo ini, no del directorio de inicio del usuario. Cuando se especifica este archivo, se ignora un archivo `~/.axisrc` existente.

4.1.12.5. user_live_update()

La gui Axis incluye una función no-op (marcador de posición) llamada *user_live_update()* que se ejecuta al final de la funcion update() de su clase LivePlotter. Esta función puede ser implementada dentro de los script python `~/.axisrc` o *[DISPLAY]USER_COMMAND_FILE* para realizar acciones personalizadas periódicas. Los detalles de lo que puede lograrse cos esta función dependerán de la implementacion de la gui Axis y sujeto a cambios durante el ciclo de desarrollo.

4.1.12.6. Editor externo

Las opciones de menú Archivo > Editar … y Archivo > Editar tabla de herramientas … estaran disponible después de definir el editor en la sección ini [DISPLAY]. Los valores útiles incluyen EDITOR=gedit y EDITOR=gnome-terminal -e vim. Para obtener más información, consulte la sección Sección display del capítulo de Configuración INI.

4.1.12.7. Panel de control virtual

AXIS puede mostrar un panel de control virtual personalizado en la zona derecha. Puede programar botones, indicadores, pantallas de datos y más cosas. Para más información, consulte los capitulos <<cha:pyvcp,PyVCP y <<cha:glade-vcp,GladeVCP.

4.1.12.8. Control de vista previa

Se pueden insertar comentarios especiales en el archivo de Código G para controlar cómo se comporta la vista previa de AXIS. En el caso de que quiera limitar el dibujo de la vista previa utiliza estos comentarios especiales, cualquier cosa entre (AXIS,hide) y (AXIS,show) no se dibujará durante la vista previa. (AXIS,hide) y (AXIS,show) deben usarse en pares con, (AXIS,hide) primero. Cualquier cosa después de (AXIS,stop) no se dibujará durante la vista previa.

Estos comentarios son útiles para limpiar la visualización de vista previa (por ejeplo, mientras se depura un archivo g-code más grande, se puede deshabilitar la vista previa de ciertas partes que ya están trabajando bien).

- (AXIS,hide) Detiene la vista previa (debe ser la primera)
- (AXIS,show) Reanuda la vista previa (debe seguir un hide)
- (AXIS,stop) Detiene la vista previa desde aquí hasta el final del archivo.
- (AXIS,notify, el_texto) Muestra el_texto como una pantalla de información. Esta pantalla puede ser útil en la vista previa de Axis cuando los comentarios (debug,message) no se muestran.

4.1.12.9. Pines Axisui

Para mejorar la interacción de AXIS con jogwheels físicos, el eje actualmente seleccionado en la GUI también se reporta en un pin con un nombre como *axisui.jog.x*. Uno de estos pines es *VERDADERO* y el resto son *FALSO*. Están diseñados para controlar los pines de habilitacion de movimiento jog.

Axisui Pins El eje tiene pines Hal para indicar qué botón de selección de jog está seleccionado en el Pestaña *Control manual*.

```
Tipo Dir Nombre
bit  OUT axisui.jog.x
bit  OUT axisui.jog.y
bit  OUT axisui.jog.z
bit  OUT axisui.jog.a
bit  OUT axisui.jog.b
bit  OUT axisui.jog.c
bit  OUT axisui.jog.u
bit  OUT axisui.jog.v
bit  OUT axisui.jog.w
```

Axis tiene un pin Hal para indicar el incremento de jog seleccionado en *Control Manual*.

```
Tipo  Dir Nombre
float OUT axisui.jog.increment
```

Axis tiene pines de entrada Hal para borrar las notificaciones emergentes de errores e información.

```
Tipo  Dir Nombre
bit   IN  axisui.notifications-clear
bit   IN  axisui.notifications-clear-error
bit   IN  axisui.notifications-clear-info
```

Axis tiene un pin de entrada Hal que deshabilita/habilita la función *Pausa/Reanudar*.

```
Tipo  Dir Nombre
bit   IN  axisui.resume-inhibit
```

4.1.13. Sugerencias de personalización de Axis

Axis es un código bastante grande y difícil de penetrar. Esto es útil para mantener el código estable pero dificulta la personalización.
Aquí mostraremos fragmentos de código para modificar comportamientos o imágenes de la pantalla. Tenga en cuenta que el código interno de AXIS puede cambiar de vez en cuando.
No se garantiza que estos fragmentos continúen funcionando; pueden necesitar ajustes.

4.1.13.1. La función de actualización

Hay una función en Axis llamada user_live_update que se llama cada vez que Axis se actualiza solo. Puede usar esto para actualizar sus propias funciones.

```
# función de actualización continua
def user_live_update():
    print 'i am printed every update...'
```

4.1.13.2. Deshabilitar el cuadro de diálogo Cerrar

```
# disable the do you want to close dialog
root_window.tk.call("wm","protocol",".","WM_DELETE_WINDOW","destroy .")
```

4.1.13.3. Cambiar la fuente del texto

```
# cambiar la fuente

font = 'sans 11'
fname,fsize = font.split()
root_window.tk.call('font','configure','TkDefaultFont','-family',fname,'-size',fsize)

# rehace el texto en pestañas para que cambien el tamaño de la nueva fuente predeterminada

root_window.tk.call('.pane.top.tabs','itemconfigure','manual','-text',' Manual - F3 ')
root_window.tk.call('.pane.top.tabs','itemconfigure','mdi','-text',' MDI - F5 ')
root_window.tk.call('.pane.top.right','itemconfigure','preview','-text',' Preview ')
root_window.tk.call('.pane.top.right','itemconfigure','numbers','-text',' DRO ')


# la fuente gcode es independiente

root_window.tk.call('.pane.bottom.t.text','configure','-foreground','blue')
#root_window.tk.call('.pane.bottom.t.text','configure','-foreground','blue','-font',font)
#root_window.tk.call('.pane.bottom.t.text','configure','-foreground','blue','-font',font,'- ↩
    height','12')
```

4.1.13.4. Modificar velocidad rápida con atajos de teclado

```
# use control + ` o 1-0 como atajos de teclado para acelerar y mantener ` o 1-0 para avance
# también agrega texto a la referencia rápida en la ayuda

help1.insert(10,("Control+ `,1..9,0", _("Establecer ajuste de rápidos de 0% a 100%")),)

root_window.bind('<Control-Key-quoteleft>',lambda event: set_rapidrate(0))
root_window.bind('<Control-Key-1>',lambda event: set_rapidrate(10))
root_window.bind('<Control-Key-2>',lambda event: set_rapidrate(20))
root_window.bind('<Control-Key-3>',lambda event: set_rapidrate(30))
root_window.bind('<Control-Key-4>',lambda event: set_rapidrate(40))
root_window.bind('<Control-Key-5>',lambda event: set_rapidrate(50))
root_window.bind('<Control-Key-6>',lambda event: set_rapidrate(60))
root_window.bind('<Control-Key-7>',lambda event: set_rapidrate(70))
root_window.bind('<Control-Key-8>',lambda event: set_rapidrate(80))
root_window.bind('<Control-Key-9>',lambda event: set_rapidrate(90))
root_window.bind('<Control-Key-0>',lambda event: set_rapidrate(100))
root_window.bind('<Key-quoteleft>',lambda event: set_feedrate(0))
root_window.bind('<Key-1>',lambda event: set_feedrate(10))
root_window.bind('<Key-2>',lambda event: set_feedrate(20))
root_window.bind('<Key-3>',lambda event: set_feedrate(30))
root_window.bind('<Key-4>',lambda event: set_feedrate(40))
root_window.bind('<Key-5>',lambda event: set_feedrate(50))
root_window.bind('<Key-6>',lambda event: set_feedrate(60))
root_window.bind('<Key-7>',lambda event: set_feedrate(70)
root_window.bind('<Key-8>',lambda event: set_feedrate(80))
root_window.bind('<Key-9>',lambda event: set_feedrate(90))
root_window.bind('<Key-0>',lambda event: set_feedrate(100))
```

4.1.13.5. Leer el archivo INI

```
# leer un elemento del archivo ini

machine = inifile.find('EMC','MACHINE')
print 'machine name =',machine
```

4.1.13.6. Leer el estado de linuxcnc

```
# El estado de linuxcnc se puede leer desde s.

print s.actual_position
print s.paused
```

4.1.13.7. Cambiar la vista actual

```
# establecer la vista de la vista previa
# las vistas válidas son view_x view_y view_y2 view_z view_z2 view_p

command.set_view_z()
```

4.1.13.8. Crear nuevos pines AXISUI HAL

```
def user_hal_pins():
    comp.newpin('my-new-in-pin', hal.HAL_BIT, hal.HAL_IN)
    comp.ready()
```

4.1.13.9. Crear nuevos componentes y pines HAL

```
# crear un componente

mycomp = hal.component('my_component')
mycomp.newpin('idle-led',hal.HAL_BIT,hal.HAL_IN)
mycomp.newpin('pause-led',hal.HAL_BIT,hal.HAL_IN)
mycomp.ready()

# pines de conexión

hal.new_sig('idle-led',hal.HAL_BIT)
hal.connect('halui.program.is-idle','idle-led')
hal.connect('my_component.idle-led','idle-led')

# establecer un pin

hal.set_p('my_component.pause-led','1')

# obtener un pin de rama 2,8+

value = hal.get_value('halui.program.is-idle')
print 'value is a',type(value),'value of',value
```

4.1.13.10. Cambiar pestañas con pines HAL

```
# los pines hal de un panel GladeVCP no estarán listos cuando se ejecute user_live_update
# para leerlos necesita ponerlos en un bloque try/except

# el siguiente ejemplo supone 5 botones HAL en un panel GladeVCP utilizado para cambiar
# las pestañas en la pantalla Axis.
# los nombres de los botones son 'manual-tab', 'mdi-tab', 'preview-tab', 'dro-tab', 'user0- ↩
    tab'
# la pestaña user_0, si existe, sería la primera pestaña incrustada GladeVCP

# para la rama linuxCNC 2.8+

def user_live_update():
    try:
        if hal.get_value('gladevcp.manual-tab'):
            root_window.tk.call('.pane.top.tabs','raise','manual')
        elif hal.get_value('gladevcp.mdi-tab'):
            root_window.tk.call('.pane.top.tabs','raise','mdi')
        elif hal.get_value('gladevcp.preview-tab'):
            root_window.tk.call('.pane.top.right','raise','preview')
        elif hal.get_value('gladevcp.numbers-tab'):
            root_window.tk.call('.pane.top.right','raise','numbers')
        elif hal.get_value('gladevcp.user0-tab'):
            root_window.tk.call('.pane.top.right','raise','user_0')
    except:
        pass
```

4.1.13.11. Agregar un botón de inicio GOTO

```
def goto_home(axis):
    if s.interp_state == linuxcnc.INTERP_IDLE:
        home = inifile.find('JOINT_' + str(inifile.find('TRAJ', 'COORDINATES').upper(). ↩
            index(axis)), 'HOME')
        mode = s.task_mode
        if s.task_mode != linuxcnc.MODE_MDI:
            c.mode(linuxcnc.MODE_MDI)
        c.mdi('G53 G0 ' + axis + home)

# hacer un botón para home del eje Y
root_window.tk.call('button','.pane.top.tabs.fmanual.homey','-text','Home Y','-command',' ↩
    goto_home Y','-height','2')

# colocar el botón
root_window.tk.call('grid','.pane.top.tabs.fmanual.homey','-column','1','-row','7','- ↩
    columnspan','2','-padx','4','-sticky','w')

# cualquier función llamada desde tcl debe agregarse a TclCommands
TclCommands.goto_home = goto_home
commands = TclCommands(root_window)
```

4.1.13.12. Agregar botón al marco manual

```
# crea un nuevo botón y ponerlo en el marco manual

root_window.tk.call('button','.pane.top.tabs.fmanual.mybutton','-text','My Button','- ↩
    command','mybutton_clicked','-height','2')
root_window.tk.call('grid','.pane.top.tabs.fmanual.mybutton','-column','1','-row','6','- ↩
    columnspan','2','-padx','4','-sticky','w')

# lo anterior envía el comando "mybutton_clicked" cuando se hace clic
# otras opciones son vincular un comando de pulsar o soltar (o ambos) al botón
# estos pueden ser adicionales o en lugar del comando seleccionado
# si en lugar de eliminar '-command', 'mybutton_clicked', de la primera línea

# Botón-1 = botón izquierdo del mouse, 2 = derecho o 3 = medio

root_window.tk.call('bind','.pane.top.tabs.fmanual.mybutton','<Button-1>','mybutton_pressed ↩
    ')
root_window.tk.call('bind','.pane.top.tabs.fmanual.mybutton','<ButtonRelease-1>',' ↩
    mybutton_released')

# funciones llamadas desde los botones

def mybutton_clicked():
    print 'mybutton was clicked'
def mybutton_pressed():
    print 'mybutton was pressed'
def mybutton_released():
    print 'mybutton was released'

# cualquier función llamada desde tcl debe agregarse a TclCommands

TclCommands.mybutton_clicked = mybutton_clicked
TclCommands.mybutton_pressed = mybutton_pressed
TclCommands.mybutton_released = mybutton_released
commands = TclCommands(root_window)
```

4.1.13.13. Lectura de variables internas

```
# las siguientes variables pueden leerse desde la instancia de vars

print vars.machine.get()
print vars.emcini.get()
```

```
active_codes            = StringVar
block_delete            = BooleanVar
brake                   = BooleanVar
coord_type              = IntVar
display_type            = IntVar
dro_large_font          = IntVar
emcini                  = StringVar
exec_state              = IntVar
feedrate                = IntVar
flood                   = BooleanVar
grid_size               = DoubleVar
has_editor              = IntVar
has_ladder              = IntVar
highlight_line          = IntVar
interp_pause            = IntVar
interp_state            = IntVar
ja_rbutton              = StringVar
jog_aspeed              = DoubleVar
jog_speed               = DoubleVar
kinematics_type         = IntVar
linuxcnctop_command     = StringVar
machine                 = StringVar
max_aspeed              = DoubleVar
max_maxvel              = DoubleVar
max_queued_mdi_commands = IntVar
max_speed               = DoubleVar
maxvel_speed            = DoubleVar
mdi_command             = StringVar
metric                  = IntVar
mist                    = BooleanVar
motion_mode             = IntVar
on_any_limit            = BooleanVar
optional_stop           = BooleanVar
override_limits         = BooleanVar
program_alpha           = IntVar
queued_mdi_commands     = IntVar
rapidrate               = IntVar
rotate_mode             = BooleanVar
running_line            = IntVar
show_distance_to_go     = IntVar
show_extents            = IntVar
show_live_plot          = IntVar
show_machine_limits     = IntVar
show_machine_speed      = IntVar
show_program            = IntVar
show_pyvcppanel         = IntVar
show_rapids             = IntVar
show_tool               = IntVar
show_offsets            = IntVar
spindledir              = IntVar
```

```
spindlerate              = IntVar
task_mode                = IntVar
task_paused              = IntVar
task_state               = IntVar
taskfile                 = StringVar
teleop_mode              = IntVar
tool                     = StringVar
touch_off_system         = StringVar
trajcoordinates          = StringVar
tto_g11                  = BooleanVar
view_type                = IntVar
```

4.1.13.14. Ocultar widgets

```
# ocultar un widget
# use 'grid' o 'pack' dependiendo de cómo se colocó originalmente

root_window.tk.call('grid','forget','.pane.top.tabs.fmanual.jogf.zerohome.tooltouch')
```

4.1.13.15. Cambiar una etiqueta

```
# cambiar la etiqueta de un widget
root_window.tk.call('setup_widget_accel','.pane.top.tabs.fmanual.mist','Downdraft')

# asegúrese de que aparezca (solo es necesario en este caso si el botón de niebla estaba ↩
    oculto)
root_window.tk.call('grid','.pane.top.tabs.fmanual.mist','-column','1','-row','5','- ↩
    columnspan','2','-padx','4','-sticky','w')
```

4.1.13.16. Redirigir un comando existente

```
# secuestrar un comando existente
# originalmente el botón de niebla llama a la función de niebla

root_window.tk.call('.pane.top.tabs.fmanual.mist','configure','-command','hijacked_command' ↩
    )

# La nueva función

def hijacked_command():
    print 'hijacked mist command'

# agrega la función a TclCommands

TclCommands.hijacked_command = hijacked_command
commands = TclCommands(root_window)
```

4.1.13.17. Cambiar el color DRO

```
# cambiar la pantalla dro

root_window.tk.call('.pane.top.right.fnumbers.text','configure','-foreground','green','- ↩
    background','black')
```

4.1.13.18. Cambiar los botones de la barra de herramientas

```
# cambiar los botones de la barra de herramientas

buW = '3'
buH = '2'
boW = '3'

root_window.tk.call('.toolbar.machine_estop','configure','-image','','-text','ESTOP','- ↩
    width',buW,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.machine_power','configure','-image','','-text','POWER','- ↩
    width',buW,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.file_open','configure','-image','','-text','OPEN','-width', ↩
    buW,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.reload','configure','-image','','-text','RELOAD','-width',buW ↩
    ,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.program_run','configure','-image','','-text','RUN','-width', ↩
    buW,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.program_step','configure','-image','','-text','STEP','-width' ↩
    ,buW,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.program_pause','configure','-image','','-text','PAUSE','- ↩
    width',buW,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.program_stop','configure','-image','','-text','STOP','-width' ↩
    ,buW,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.program_blockdelete','configure','-image','','-text','Skip /' ↩
    ,'-width',buW,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.program_optpause','configure','-image','','-text','M1','- ↩
    width',buW,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.view_zoomin','configure','-image','','-text','Zoom+','-width' ↩
    ,buW,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.view_zoomout','configure','-image','','-text','Zoom-','-width ↩
    ',buW,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.view_z','configure','-image','','-text','Top X','-width',buW, ↩
    '-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.view_z2','configure','-image','','-text','Top Y','-width',buW ↩
    ,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.view_x','configure','-image','','-text','Right','-width',buW, ↩
    '-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.view_y','configure','-image','','-text','Front','-width',buW, ↩
    '-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.view_p','configure','-image','','-text','3D','-width',buW,'- ↩
    height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.rotate','configure','-image','','-text','Rotate','-width',buW ↩
    ,'-height',buH,'-borderwidth',boW)
root_window.tk.call('.toolbar.clear_plot','configure','-image','','-text','Clear','-width', ↩
    buW,'-height',buH,'-borderwidth',boW)
```

4.2. GMOCCAPY

4.2.1. Introducción

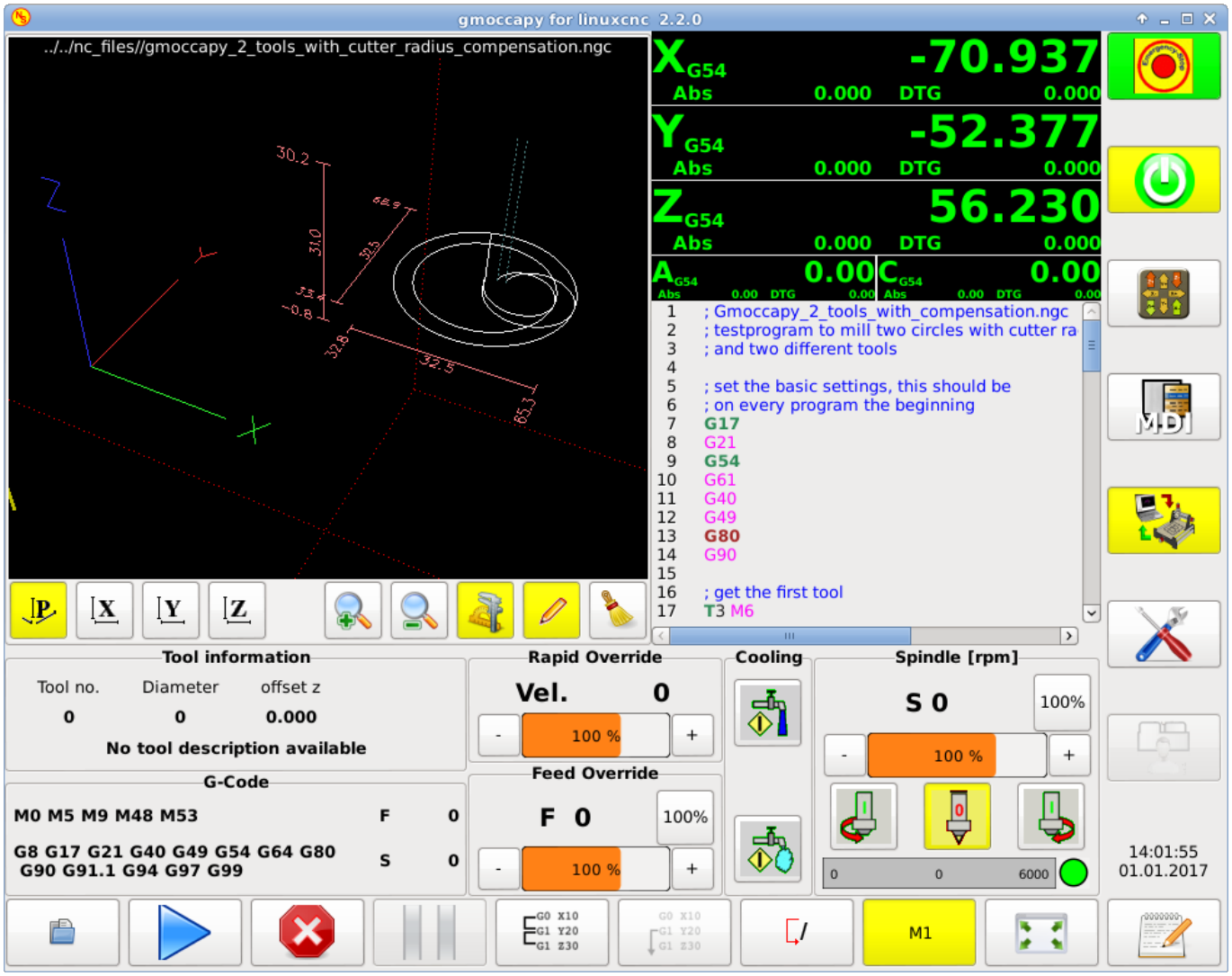
GMOCCAPY es una GUI para LinuxCNC, diseñada para ser utilizada con una pantalla táctil, pero también se puede usar en pantallas normales con un mouse o botones de hardware y volantes MPG, ya que presenta pines HAL para las necesidades más comunes. Por favor, vea mas información a continuación.

Ofrece la posibilidad de mostrar hasta 4 ejes, admite un modo de torno para herramienta normal y posterior, que puede adaptarse a casi cualquier necesidad, ya que gmoccapy soporta pestañas y paneles laterales incrustados. Como un buen ejemplo de esto, puede verse. gmoccapy_plasma

- gmoccapy 3 soporta hasta 9 ejes y 9 articulaciones. Como el código ha cambiado en gmoccapy 3 para admitir los cambios articulaciones/ejes en LinuxCNC, ¡NO trabaja en versiones 2.7 o 2.6!

Tiene soporte para teclado virtual (onboard o matchbox), por lo que no hay necesidad de un teclado o mouse de hardware, pero también se puede usar con ese hardware. Gmoccapy ofrece una página de configuración separada para configurar la GUI sin editar archivos.

GMOCCAPY se puede localizar (usar idioma local) muy fácilmente, porque los archivos correspondientes estan separados de los archivos .po de linuxcnc, por lo que no hay necesidad de traducir cosas innecesarias. Los archivos se colocan en **/src/po/gmoccapy**. Simplemente copiar el archivo gmoccapy.pot a algo como fr.po, pl.po, es.po, etc. y traducir ese archivo con gtranslator o poedit. Después de la recompilacion, tendra la GUI en su idioma preferido. Por favor, publique su traducción para que pueda ser incluido en los paquetes oficiales y ser publicado para otros usuarios. En el momento está disponible en inglés, alemán, español, polaco, serbio y húngaro. Siéntete libre de ayudarme a introducir más idiomas, **nieson@web.de**. Si necesita ayuda, no dude en preguntar.



4.2.2. Requisitos

Gmoccapy 3 se ha probado en Debian Jessie, Debian Stretch y MINT 18 con la rama master y la 2.8 de LinuxCNC. Es totalmente compatible con los cambios articulaciones/ejes de LinuxCNC, haciendolo adecuado como GUI para Scara, Robots o cualquier otra configuración con más articulaciones que ejes. También soporta configuraciones de pórtico. Si usas otras versiones, por favor, informar sobre problemas y/o soluciones en LinuxCNC forum o en German CNC Ecke Forum o en LinuxCNC lista de correo de usuarios

La resolución de pantalla mínima para gmoccapy, usándola sin paneles laterales, es **979 x 750 píxeles**, por lo que debe ajustarse a todas las pantallas estándar. Se recomienda una resolucion de 1024x748.

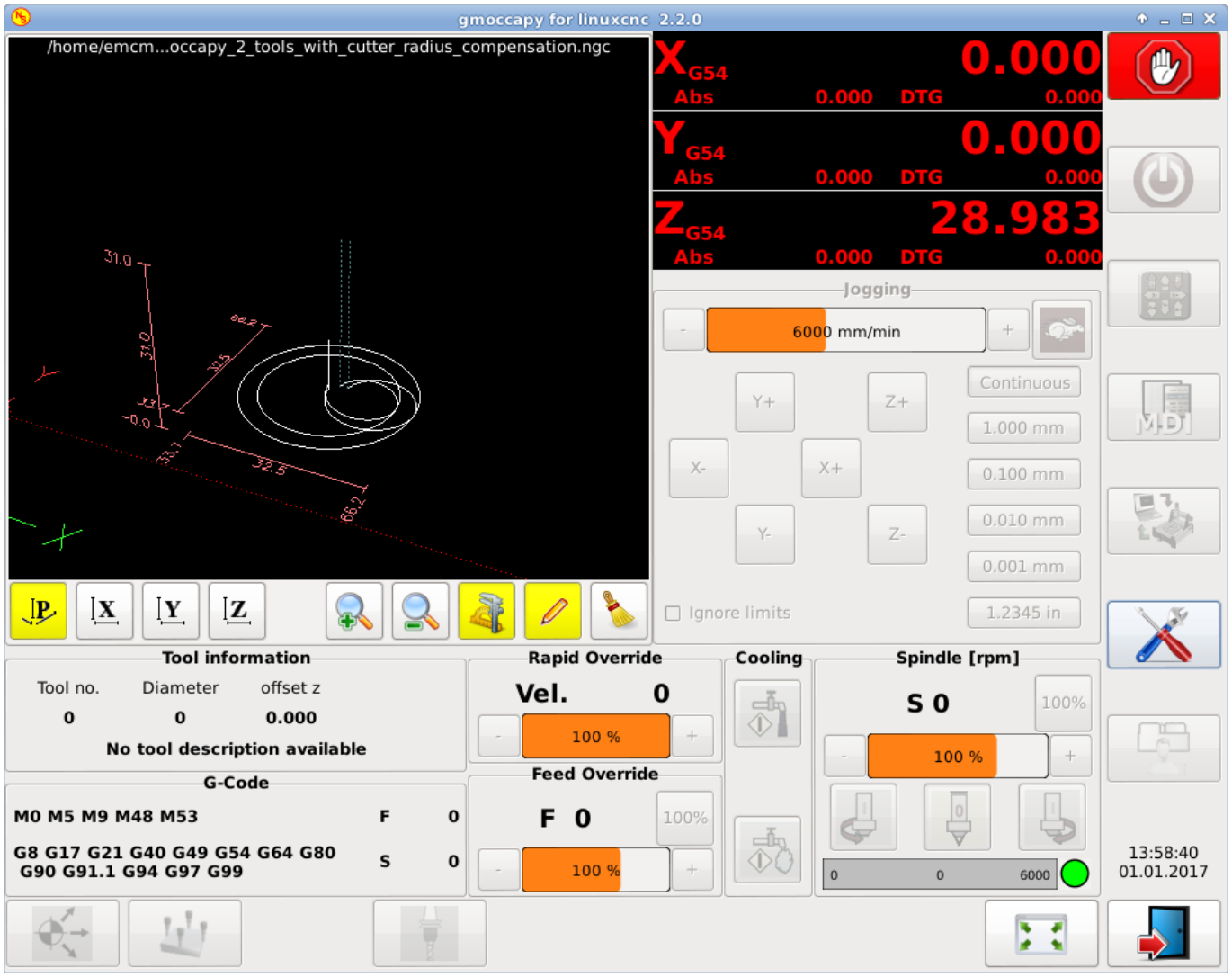
4.2.3. Cómo obtener gmoccapy

A partir de LinuxCNC 2.8 gmoccapy se incluye en la instalación estándar. Así que la forma más fácil de obtener gmoccapy en su PC de control, es simplemente obtener ISO e instalar desde el DVD/Stick-USB.

Usted recibirá actualizaciones con los paquetes deb regulares.

Gmoccapy 3 solo se incluye en la versión master y en la 2.8.

Obtendrá una pantalla similar a la siguiente: (el diseño puede variar dependiendo de su configuración.)



4.2.4. Configuración básica

Realmente no hay mucho que configurar para ejecutar gmoccapy, pero hay algunos puntos con los que debe tener cuidado si desea utilizar todas las funciones de la GUI.

Encontrará los siguientes archivos INI incluidos, solo para mostrar lo básico:

* gmoccapy.ini

* gmoccapy_4_axis.ini
* lathe_configs/gmoccapy_lathe.ini
* lathe_configs/gmoccapy_lathe_imperial.ini
* gmoccapy_left_panel.ini
* gmoccapy_right_panel.ini
* gmoccapy_messages.ini
* gmoccapy_pendant.ini
* gmoccapy_sim_hardware_button.ini
* gmoccapy_tool_sensor.ini
* gmoccapy_with_user_tabs.ini
* gmoccapy_XYZAB.ini
* gmoccapy_XYZAC.ini
* gmoccapy_XYZCW.ini
* gmoccapy-JA/Gantry/gantry_mm.ini
* gmoccapy-JA/scara/scara.ini
* gmoccapy-JA/table-rotary-tilting/xyzac-trt.ini
* y algunos mas…

Los nombres deben explicar la intención principal de los diferentes archivos INI.

Si utiliza una configuración existente en su máquina, simplemente edite su INI de acuerdo con este documento.

IMPORTANTE: si desea utilizar MACROS, no olvide configurar la ruta a sus macros o carpeta de subrutinas como se describe a continuación.

Echemos un vistazo más de cerca al archivo INI y lo que necesita incluir para usar gmoccapy en su máquina:

4.2.4.1. La sección DISPLAY

```
[DISPLAY]
DISPLAY = gmoccapy
PREFERENCE_FILE_PATH = gmoccapy_preferences
MAX_FEED_OVERRIDE = 1.5
MAX_SPINDLE_OVERRIDE = 1.2
MIN_SPINDLE_OVERRIDE = 0.5
LATHE = 1
BACK_TOOL_LATHE = 1
PROGRAM_PREFIX = ../../nc_files/
```

La parte más importante es decirle a LinuxCNC que use gmoccapy, editando la sección [DISPLAY].

```
[DISPLAY]
DISPLAY = gmoccapy

PREFERENCE_FILE_PATH = gmoccapy_preferences
```

gmoccapy 3 admite las siguientes opciones de línea de comando:

- -user_mode: si está configurado, el botón de configuración se deshabilitará, por lo que los operadores normales de la máquina no podrán editar la configuración *

- -logo <ruta de acceso al archivo de logotipo>: si se proporciona, el logotipo ocultará la pestaña del botón de jog en modo manual, esto solo es útil para máquinas con botón de hardware para jogging y selección de incrementos

La línea PREFERENCE_FILE_PATH proporciona la ubicación y el nombre del archivo de preferencias que se utilizará. En la mayoría de los casos, esta línea no será necesaria; es utilizada por gmoccapy para almacenar su configuración de la GUI, como temas, unidades DRO, colores y configuraciones de teclado, etc., vea la página de configuración para más detalles.

nota
Si no se proporciona una ruta o archivo, gmoccapy usará como predeterminado <your_machinename>.pref. Si no se da un nombre de máquina en el archivo INI, usara gmoccapy.pref El archivo se almacenará en su directorio de configuración, por lo que la configuración no se mezclará si utiliza varias configuraciones. Si solo quiere usar un archivo para varias máquinas, debe incluir PREFERENCE_FILE_PATH en su INI.

```
MAX_FEED_OVERRIDE = 1.5
```

Establece el porcentaje de alimentación máxima. En el ejemplo dado, se le permitirá variar la alimentación hasta en un 150 %.

nota
Si no se da ningún valor, se establecerá en 1.0

```
MAX_SPINDLE_OVERRIDE = 1.2
MIN_SPINDLE_OVERRIDE = 0.5
```

Le permitirá cambiar porcentajes del husillo dentro de un límite de 50 % a 120 %.

nota
Si no se dan valores, MAX se establecerá en 1.0 y MIN en 0.1

```
LATHE = 1
BACK_TOOL_LATHE = 1
```

La primera línea establece la pantalla con el diseño para controlar un torno.

La segunda línea es opcional y cambiará el eje X de la manera que se necesita para un torno con herramienta posterior. También los atajos de teclado reaccionarán de una manera diferente. Con gmoccapy se permite configurar un torno también con eje adicional, por lo que también puede usar una configuración XCW para un torno.

sugerencia
Consulte también la sección específica del torno

- PROGRAM_PREFIX = ../../nc_files/

Es la entrada para indicar a linuxcnc/gmoccapy dónde buscar los archivos ngc.

nota
Si no se especifica, Gmoccapy buscará en el siguiente orden los archivos ngc: linuxcnc/nc_files y luego el directorio home de los usuarios.

Configuración de pestañas y paneles laterales.

Puede agregar programas integrados a gmoccapy como lo puede hacer en axis, touchy y gscreen. Todo se hace automáticamente por gmoccapy si incluye algunas líneas en su archivo INI en la sección DISPLAY.

Si nunca usó un panel glade, se recomienda leer la excelente documentación. Glade VCP

Ejemplo

```
EMBED_TAB_NAME = DRO
EMBED_TAB_LOCATION = ntb_user_tabs
EMBED_TAB_COMMAND = gladevcp -x {XID} dro.glade

EMBED_TAB_NAME = Segunda pestaña de usuario
EMBED_TAB_LOCATION = ntb_preview
EMBED_TAB_COMMAND = gladevcp -x {XID} vcp_box.glade
```

Todo lo que debe tener en cuenta es incluir para cada pestaña o panel lateral las tres líneas mencionadas,

- EMBED_TAB_NAME = Representa el nombre de la pestaña o el panel lateral. Depende de usted qué nombre usar, pero debe estar presente.
- EMBED_TAB_LOCATION = Es el lugar donde se colocará su programa en la GUI.

LOS VALORES VÁLIDOS SON:

- ntb_user_tabs (como pestaña principal, que cubre la pantalla completa)
- ntb_preview (como pestaña en el lado de vista previa)
- box_left (a la izquierda, arriba de la pantalla)
- box_right (a la derecha, entre la pantalla normal y la lista de botones)
- box_coolant_and_spindle (ocultará los marcos del refrigerante y del husillo e introducirá aqui su archivo glade)
- box_cooling (ocultará el cuadro de refrigerante e introducirá su archivo glade)
- box_spindle (ocultará el marco del husillo e introducirá su archivo glade)
- box_vel_info (ocultará los cuadros de velocidad e introducirá tu archivo glade)
- box_custom_1 (presentará tu archivo glade a la izquierda de vel_frame)
- box_custom_2 (presentará tu archivo glade a la izquierda de cooling_frame)
- box_custom_3 (presentará tu archivo glade a la izquierda de spindle_frame)
- box_custom_4 (presentará su archivo glade a la derecha de spindle_frame)

Vea los diferentes archivos INI incluidos para ver las diferencias.

- EMBED_TAB_COMMAND = el comando a ejecutar, por ejemplo,

```
gladevcp -x {XID} dro.glade
```

Incluye un archivo glade personalizado llamado dro.glade en la ubicación mencionada El archivo se debe colocar en la carpeta de configuración de su máquina.

```
gladevcp h_buttonlist.glade
```

Solo abrirá una nueva ventana de usuario llamada h_buttonlist.glade. Note la diferencia; esta es independiente y puede moverse independientemente de la ventana gmoccapy.

```
gladevcp -c gladevcp -u hitcounter.py -H manual-example.hal manual-example.ui
```

Agregará el panel manual-example.ui, incluirá un controlador personalizado de python, hitcounter.py y realizará todas las conexiones después de realizar el panel de acuerdo con manual-example.hal.

nota
Si realiza alguna conexión hal al panel de glade personalizado, debe hacerlo en el archivo hal especificado en la línea EMBEDDED_TAB_COMMAND; de lo contrario, puede obtener un error de que el pin hal no existe. Esto se debe a las condiciones de carga de los archivos hal . Las conexiones a pines gmoccapy hal deben realizarse en el archivo hal postgui especificado en su archivo INI, porque este pin no existe antes de realizar la GUI

Aquí hay unos ejemplos:

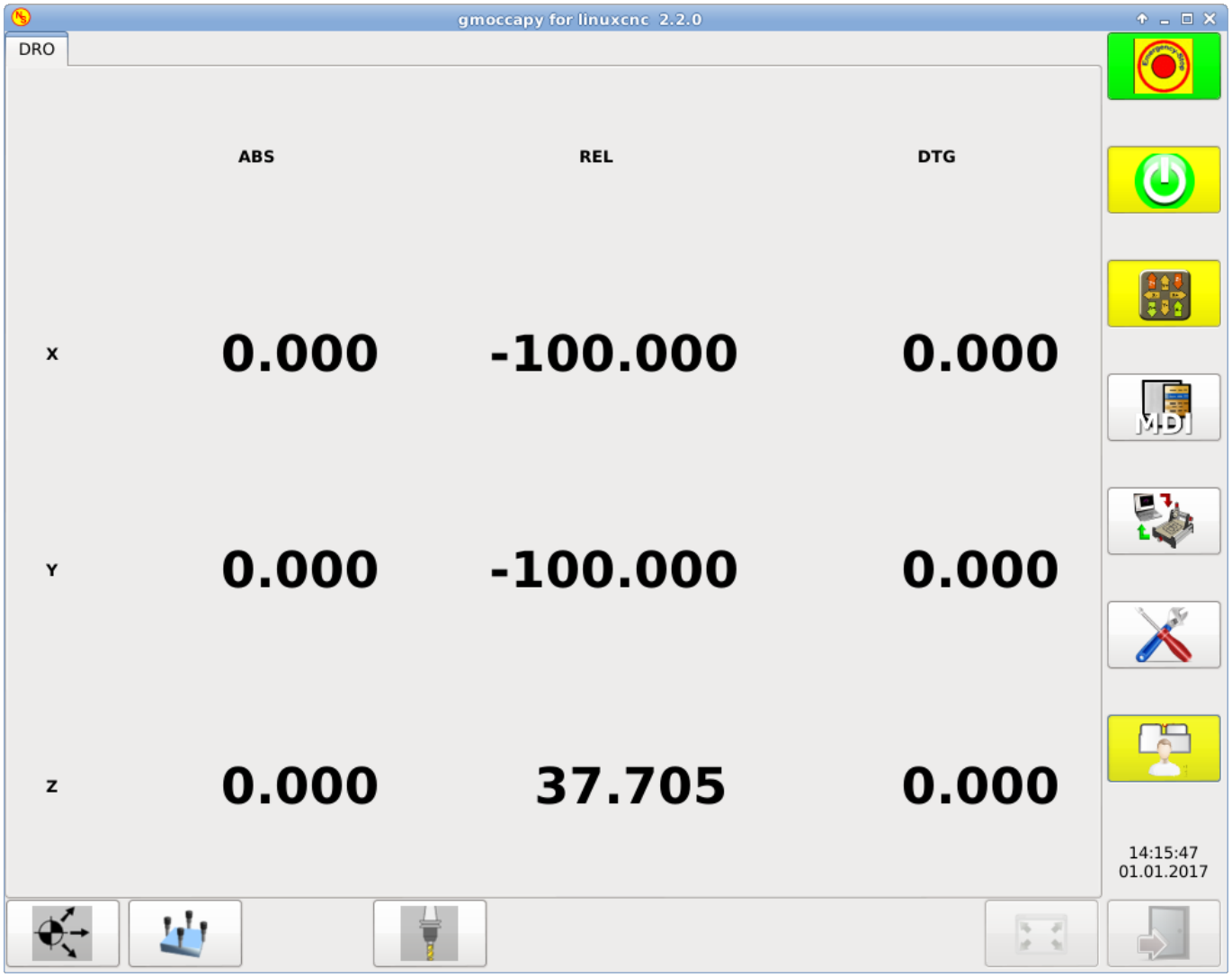


Figura 4.8: ntb_preview - como versión maximizada

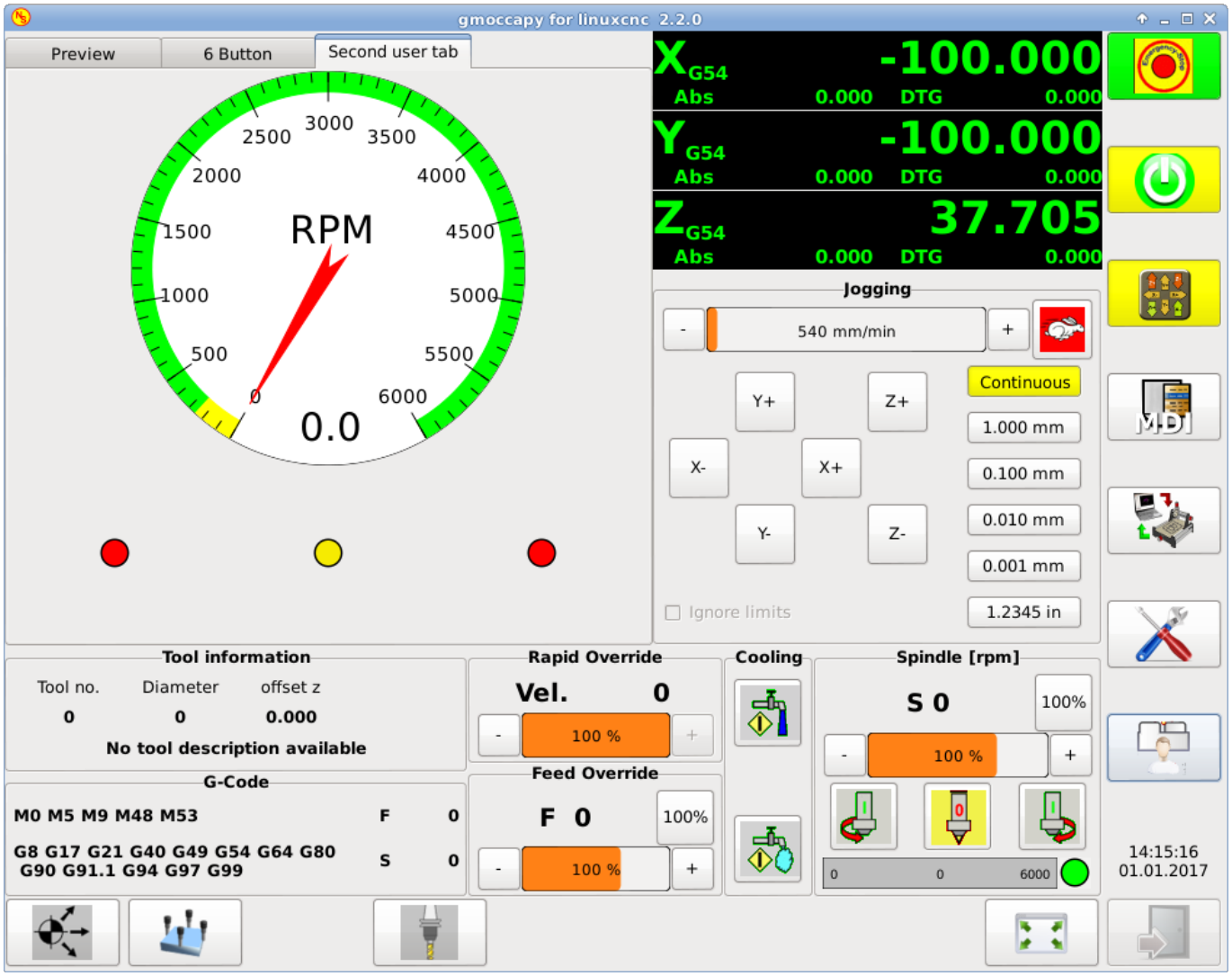


Figura 4.9: ntb_preview

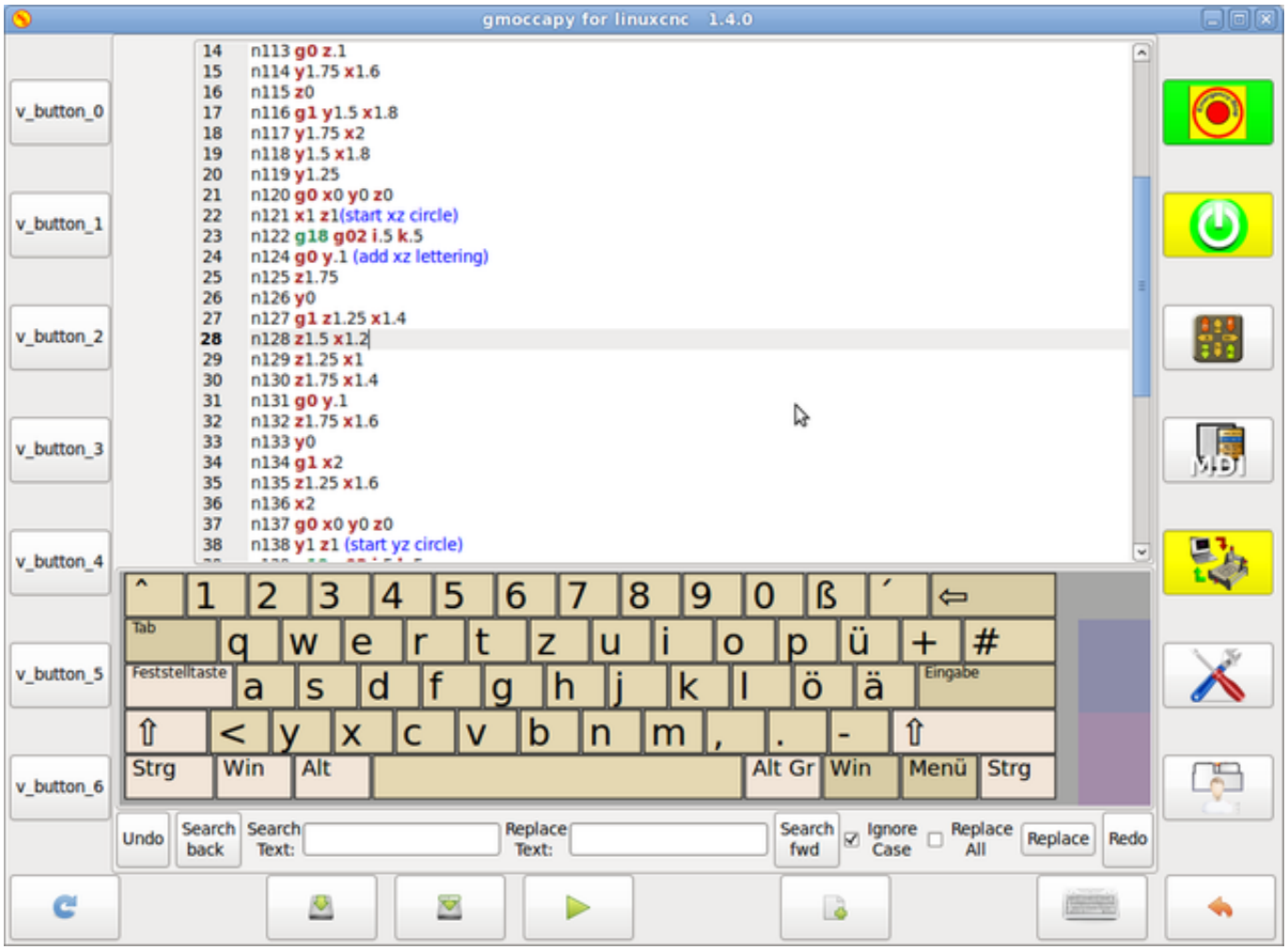


Figura 4.10: box_left - mostrando gmoccapy en modo de edición

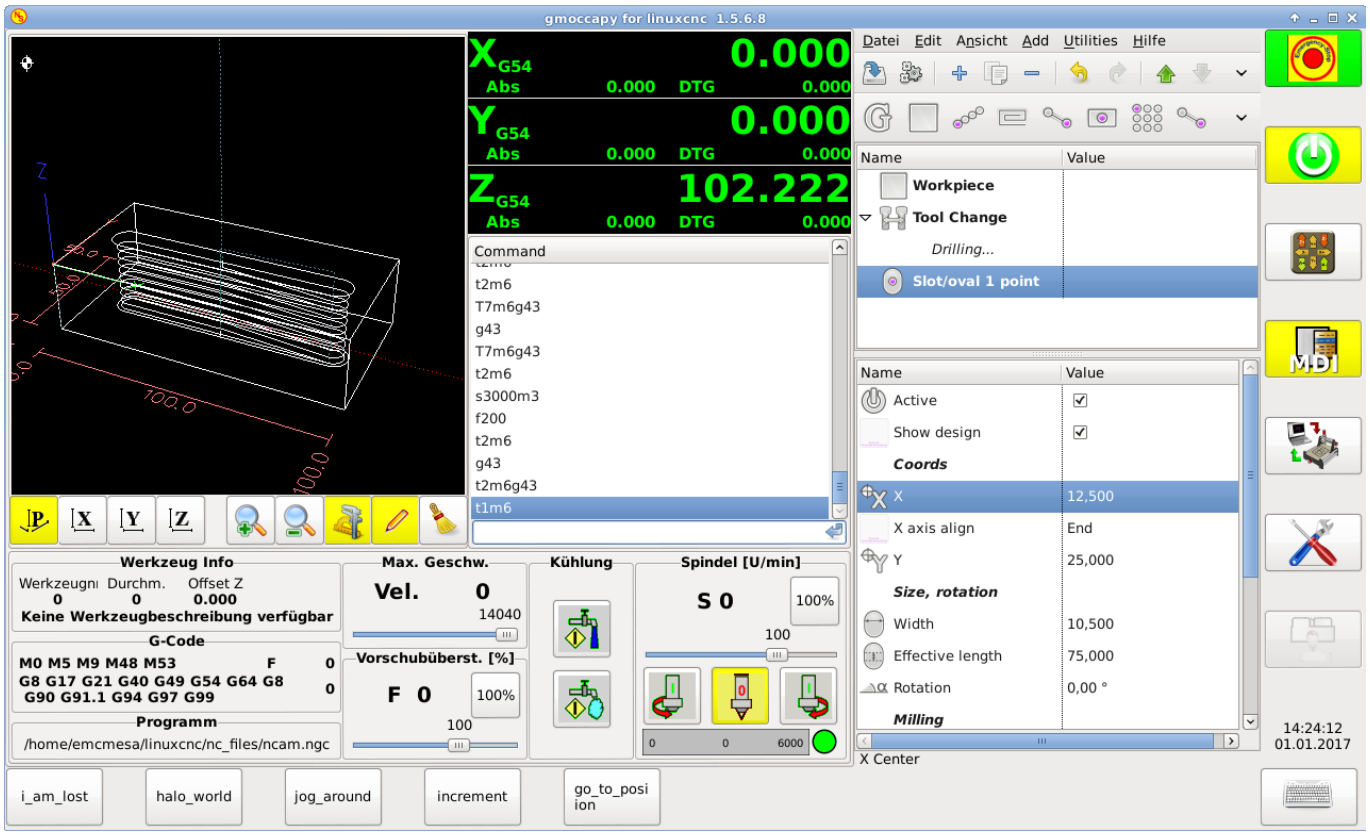


Figura 4.11: box_right - y gmoccapy en modo MDI

Configuración de mensajes creados por el usuario Gmoccapy tiene la capacidad de crear mensajes de usuario desde hal. Para usarlos es necesario introducir algunas líneas en la sección [DISPLAY] del archivo INI.

Aquí se muestra cómo configurar 3 cuadros de diálogo de mensaje emergente de usuario. Los mensajes admiten el lenguaje de marcado pango. Puede encontrar información detallada sobre el lenguaje de marcado en Pango Markup

```
MESSAGE_TEXT = El texto a mostrar, puede tener formato pango
MESSAGE_TYPE = "status", "okdialog", "yesnodialog"
MESSAGE_PINNAME = es el nombre del grupo de pines hal que se creará
```

- *status* : solo mostrará un mensaje como ventana emergente, usando el sistema de mensajes de gmoccapy
- *okdialog* : mantendrá el foco en el cuadro de diálogo del mensaje y activará un Hal_Pin OUT "-waiting". Cerrar el mensaje restablecerá el pin de espera
- *yesnodialog* : mantendrá el foco en el cuadro de diálogo del mensaje y lo activará un bit Hal_Pin OUT "-waiting" y también dará acceso a un bit Hal_Pin Out "-response". Este pin mantendrá 1 si el usuario hace clic en Aceptar, y en todos los otros estados será 0. Al cerrar el mensaje se restablecerá el pin de espera. El pin hal de respuesta permanecerá a 1 hasta que se vuelva a llamar al diálogo.

Ejemplo

```
MESSAGE_TEXT = Este es un <span background="#ff0000" foreground="#ffffff">
mensaje-info</span>  de prueba
MESSAGE_TYPE = status
MESSAGE_PINNAME = statustest

MESSAGE_TEXT = Esta es una prueba de diálogo sí/no
```

```
MESSAGE_TYPE = yesnodialog
MESSAGE_PINNAME = yesnodialog

MESSAGE_TEXT = El texto puede ser <small>small</small>, <big>big</big>, <b>bold</ b<i> ↩
    italic</i>, e incluso
puede ser <span color="red">coloreado</span>.
MESSAGE_TYPE = okdialog
MESSAGE_PINNAME = okdialog
```

Las convenciones específicas de pines hal para esto se pueden encontrar en la seccion mensajes de usuario.

4.2.4.2. La sección RS274NGC

```
[RS274NGC]
SUBROUTINE_PATH = macros
```

Establece la ruta para buscar macros y otras subrutinas. Si quiere usar varias rutas de subrutinas, simplemente sepárelas con ":"

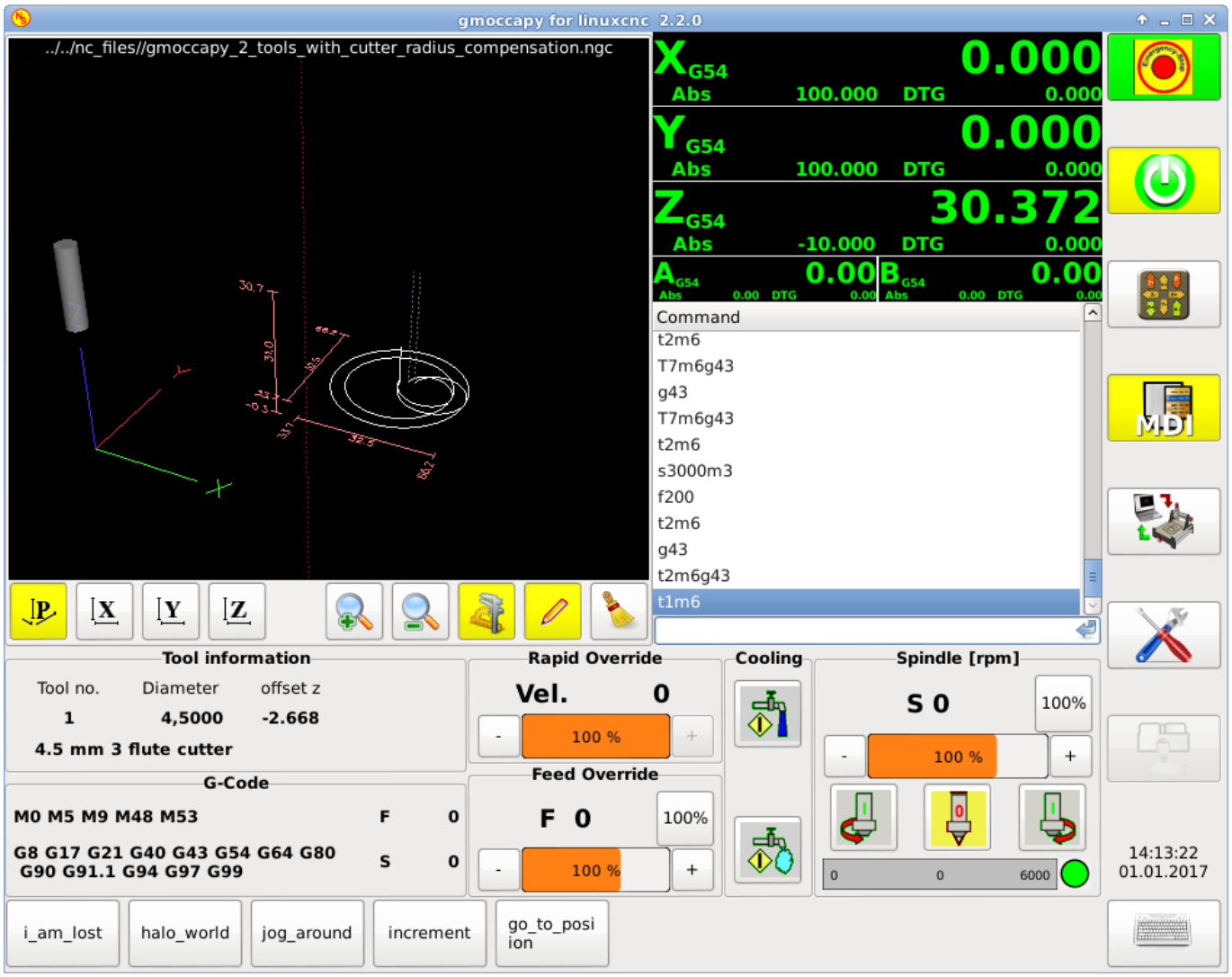
4.2.4.3. La sección MACRO

Puede agregar macros a gmoccapy, de manera parecida a Touchy. Una macro no es nada más que un archivo ngc. Asi se pueden ejecutar programas completos de CNC en modo MDI, simplemente pulsando un botón. Para hacerlo, tiene que agregar una sección parecida a:

```
[MACROS]
MACRO = i_am_lost
MACRO = hello_world
MACRO = jog_around
MACRO = increment xinc yinc
MACRO = go_to_position X-pos Y-pos Z-pos
```

Esto agregará 5 macros a la lista de botones MDI.

nota
Como aparecerán un máximo de 16 macros en la GUI, debido a razones de espacio, es posible que deba hacer clic en una flecha para cambiar de página y mostrar el botón de macro oculto. No es un error colocar más en su archivo INI. El botón de macro se mostrará en el orden de las entradas INI.



El nombre del archivo debe ser **exactamente el mismo** que el nombre dado en la linea MACRO. Así, la macro ***i_am_lost*** llamará al archivo ***i_am_lost.ngc***.

Los archivos de macros deben seguir algunas reglas:

- el nombre del archivo debe ser el mismo que el nombre mencionado en la linea MACRO, con la extensión ngc
- El archivo debe contener una subrutina como: ***O<i_am_lost> sub***; el nombre de la subrutina debe coincidir exactamente (**distingue entre mayúsculas y minúsculas**)con el nombre de la macro
- el archivo debe terminar con un endsub ***O<i_am_lost> endsub*** seguido de un comando ***M2***
- los archivos deben colocarse en una carpeta especificada en su archivo INI en la sección [RS274NGC] (ver RS274NGC)

El código entre sub y endsub se ejecutará presionando el botón de macro correspondiente.

nota
Encontrará macros de muestra en la carpeta de macros colocadas en la carpeta sim de gmoccapy. Si ha dado varias rutas de subrutinas, se buscarán en el orden de los caminos dados. Se utilizará el primer archivo encontrado.

Gmoccapy también aceptará macros que soliciten parámetros como:

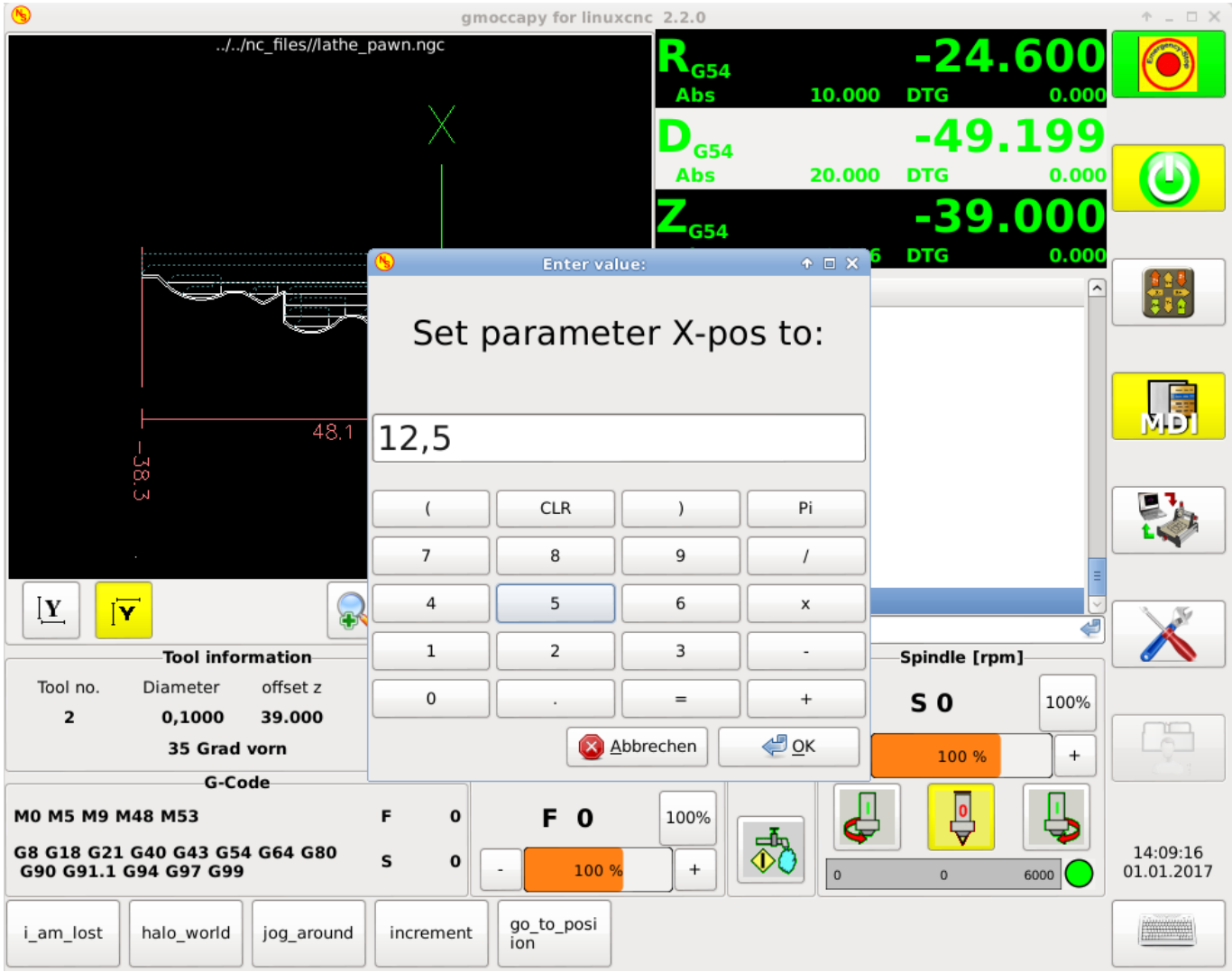
```
go_to_position X-pos Y-pos Z-pos
```

Los parámetros deben estar separados por espacios. Esto llama a un archivo *go_to_position.ngc* con el siguiente contenido:

```
; Archivo de prueba ir a la posición
; moverá la máquina a una posición dada

O<go_to_position> sub

G17
G21
G54
G61
G40
G49
G80
G90

;#1 = <X-Pos>
;#2 = <Y-Pos>
;#3 = <Z-Pos>

(DBG, Ahora moverá la máquina a X = #1, Y = #2, Z = #3)
G0 X #1 Y #2 Z #3

O<go_to_position> endsub
M2
```

Después de presionar el botón ***ejecutar macro***, se le pedirá que ingrese valores para ***X-pos Y-pos Z-pos*** y la macro solo se ejecutará si todos los valores han sido dados.

nota
Si desea usar una macro sin ningún movimiento, vea también las notas en problemas conocidos



4.2.4.4. La sección TRAJ

```
DEFAULT_LINEAR_VELOCITY = 85.0
MAX_VELOCITY = 230.000
```

Establece la velocidad máxima y la velocidad de jog predeterminada de la máquina.

nota
Si no se proporciona DEFAULT_LINEAR_VELOCITY, se utilizará la mitad de MAX_VELOCITY. Si ese valor tampoco se da, se establecerá de forma predeterminada en 180 Si no se da MAX_VELOCITY, se establecerá por defecto en 600

4.2.5. Pines HAL

gmoccapy exporta varios pines hal para poder reaccionar a dispositivos de hardware. El objetivo es obtener una GUI que pueda operarse completamente o en su mayor parte sin ratón ni teclado.

nota
Tendrá que hacer todas las conexiones a los pines gmoccapy en su archivo postgui.hal, porque no están disponibles antes de cargar la GUI completamente. Es posible llamar a varios archivos postgui hal, facilitando la depuración de la configuración. Para ello, use un archivo postgui_call_list.hal. Las conexiones al panel de usuario deben hacerse en un archivo hal separado, ya que los paneles se cargan después de la GUI. Ver pestañas y paneles laterales para detalles.

4.2.5.1. Listas de botones derecha e inferior

La pantalla tiene dos listas de botones principales, una en el lado derecho y otra en la parte inferior. Los botones de la derecha no cambiarán durante la operación, pero la lista de botones inferior cambiará muy a menudo. Los botones se numeran de arriba hacia abajo y de izquierda a derecha comenzando con "0".

nota
los nombres de pin para **Gmoccapy 3** ha cambiado para ordenarlos mejor:

En hal_show verá que los botones derechos (verticales) son:

- gmoccapy.v-button.button-0
- gmoccapy.v-button.button-1
- gmoccapy.v-button.button-2
- gmoccapy.v-button.button-3
- gmoccapy.v-button.button-4
- gmoccapy.v-button.button-5
- gmoccapy.v-button.button-6

y los botones inferiores (horizontales) son:

- gmoccapy.h-button.button-0
- gmoccapy.h-button.button-1
- gmoccapy.h-button.button-2
- gmoccapy.h-button.button-3
- gmoccapy.h-button.button-4
- gmoccapy.h-button.button-5
- gmoccapy.h-button.button-6
- gmoccapy.h-button.button-7
- gmoccapy.h-button.button-8
- gmoccapy.h-button.button-9

A medida que los botones en la lista inferior cambien de acuerdo con el modo y otras consideraciones, los botones de hardware activarán diferentes funciones, y usted no tendra que cuidar de cambiar las funciones en hal, porque eso se hace completamente dentro de gmoccapy!

for a 3 axis XYZ machine the hal pin will react as follows:

in manual mode:

- gmoccapy.h-button.button-0 == open homing button
- gmoccapy.h-button.button-1 == open touch off stuff
- gmoccapy.h-button.button-2 ==
- gmoccapy.h-button.button-3 == open tool dialogs
- gmoccapy.h-button.button-4 ==
- gmoccapy.h-button.button-5 ==
- gmoccapy.h-button.button-6 ==
- gmoccapy.h-button.button-7 ==
- gmoccapy.h-button.button-8 == full-size preview
- gmoccapy.h-button.button-9 == exit if machine is off, otherwise no reaction

in mdi mode:

- gmoccapy.h-button.button-0 == macro_0 or nothing
- gmoccapy.h-button.button-1 == macro_1 or nothing
- gmoccapy.h-button.button-2 == macro_2 or nothing
- gmoccapy.h-button.button-3 == macro_3 or nothing
- gmoccapy.h-button.button-4 == macro_4 or nothing
- gmoccapy.h-button.button-5 == macro_5 or nothing
- gmoccapy.h-button.button-6 == macro_6 or nothing
- gmoccapy.h-button.button-7 == macro_7 or nothing
- gmoccapy.h-button.button-8 == macro_8 or switch page to additional macros
- gmoccapy.h-button.button-9 == open keyboard or abort if macro is running

in auto mode

- gmoccapy.h-button.button-0 == open file
- gmoccapy.h-button.button-1 == reload program
- gmoccapy.h-button.button-2 == run
- gmoccapy.h-button.button-3 == stop
- gmoccapy.h-button.button-4 == pause
- gmoccapy.h-button.button-5 == step by step
- gmoccapy.h-button.button-6 == run from line if enabled in settings, otherwise Nothing
- gmoccapy.h-button.button-7 == optional blocks
- gmoccapy.h-button.button-8 == full-size preview
- gmoccapy.h-button.button-9 == edit code

in settings mode:

- gmoccapy.h-button.button-0 == delete MDI history
- gmoccapy.h-button.button-1 ==
- gmoccapy.h-button.button-2 ==
- gmoccapy.h-button.button-3 ==
- gmoccapy.h-button.button-4 == open classic ladder
- gmoccapy.h-button.button-5 == open hal scope
- gmoccapy.h-button.button-6 == open hal status
- gmoccapy.h-button.button-7 == open hal meter
- gmoccapy.h-button.button-8 == open hal calibration
- gmoccapy.h-button.button-9 == open hal show

in homing mode:

- gmoccapy.h-button.button-0 ==
- gmoccapy.h-button.button-1 == home all
- gmoccapy.h-button.button-2 ==
- gmoccapy.h-button.button-3 == home x
- gmoccapy.h-button.button-4 == home y
- gmoccapy.h-button.button-5 == home z
- gmoccapy.h-button.button-6 ==
- gmoccapy.h-button.button-7 ==
- gmoccapy.h-button.button-8 == unhome all
- gmoccapy.h-button.button-9 == back

in touch off mode:

- gmoccapy.h-button.button-0 == edit offsets
- gmoccapy.h-button.button-1 == touch X
- gmoccapy.h-button.button-2 == touch Y
- gmoccapy.h-button.button-3 == touch Z
- gmoccapy.h-button.button-4 ==
- gmoccapy.h-button.button-5 ==
- gmoccapy.h-button.button-6 == zero G92
- gmoccapy.h-button.button-7 ==
- gmoccapy.h-button.button-8 == set selected
- gmoccapy.h-button.button-9 == back

in tool mode:

- gmoccapy.h-button.button-0 == delete tool(s)
- gmoccapy.h-button.button-1 == new tool
- gmoccapy.h-button.button-2 == reload tool table
- gmoccapy.h-button.button-3 == apply changes
- gmoccapy.h-button.button-4 == change tool by number T? M6
- gmoccapy.h-button.button-5 == set tool by number without change M61 Q?
- gmoccapy.h-button.button-6 == change tool to the selected one
- gmoccapy.h-button.button-7 ==
- gmoccapy.h-button.button-8 == touch of tool in Z
- gmoccapy.h-button.button-9 == back

in edit mode:

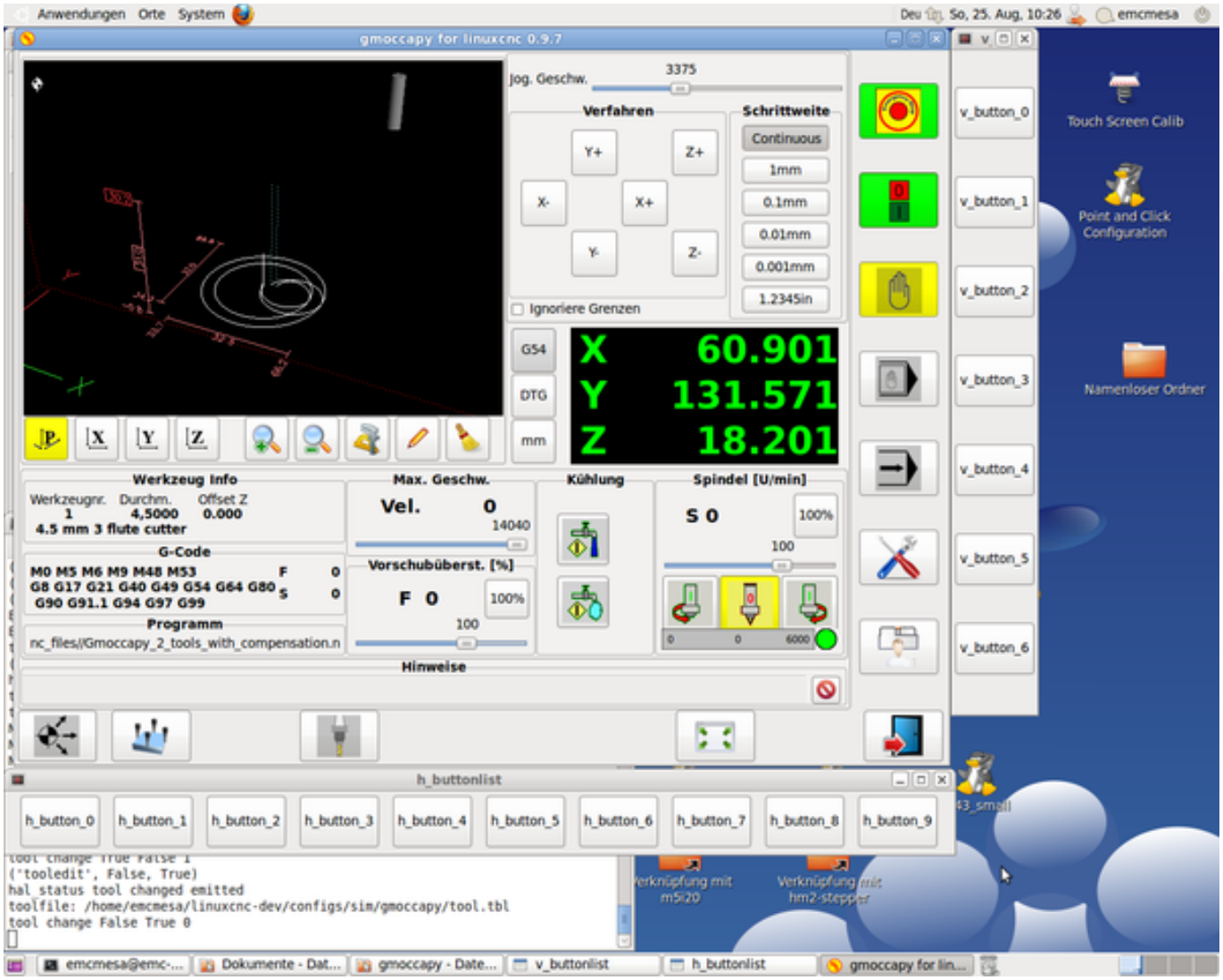
- gmoccapy.h-button.button-0 ==
- gmoccapy.h-button.button-1 == reload file
- gmoccapy.h-button.button-2 == save
- gmoccapy.h-button.button-3 == save as
- gmoccapy.h-button.button-4 ==
- gmoccapy.h-button.button-5 ==
- gmoccapy.h-button.button-6 == new file
- gmoccapy.h-button.button-7 ==
- gmoccapy.h-button.button-8 == show keyboard
- gmoccapy.h-button.button-9 == back

in select file mode:

- gmoccapy.h-button.button-0 == go to home directory
- gmoccapy.h-button.button-1 == one directory level up
- gmoccapy.h-button.button-2 ==
- gmoccapy.h-button.button-3 == move selection left
- gmoccapy.h-button.button-4 == move selection right
- gmoccapy.h-button.button-5 == jump to directory as set in settings
- gmoccapy.h-button.button-6 ==
- gmoccapy.h-button.button-7 == select / ENTER
- gmoccapy.h-button.button-8 ==
- gmoccapy.h-button.button-9 == back

So we have 67 reactions with only 10 hal pin!

Estos pines están disponibles para poder utilizar la pantalla sin un panel tactil, o protégerlos del uso excesivo colocando botones de hardware alrededor el panel.



4.2.5.2. Velocidades y porcentajes

Todos los controles deslizantes de gmoccapy se pueden conectar a encoders o potenciómetros hardware.

---

nota
para Gmoccapy 3 los nombres de pines hal han cambiado, a medida que se implementaron nuevos controles. La velocidad máxima ya no existe, ya que se ha implementado el porcentaje de rapidos. Este cambio se ha hecho ya que lo pidieron muchos usuarios.

Para conectar los *encoders* se exportan los siguientes pines:

PIN	TIPO	FUNCION
gmoccapy.jog.jog-velocity.counts	HAL_S32	velocidad Jog

PIN	TIPO	FUNCION
gmoccapy.jog.jog-velocity.count-enable	HAL_BIT	verdadero habilita conteos
gmoccapy.feed.feed-override.counts	HAL_S32	porcentaje de alimentacion
gmoccapy.feed.feed-override.count-enable	HAL_BIT	verdadero habilita conteos
gmoccapy.feed.reset-feed-override	HAL_BIT	restablece alimentación al 100 %
gmoccapy.spindle.spindle-override.counts	HAL_S32	porcentaje del husillo
gmoccapy.spindle.spindle-override.count-enable	HAL_BIT	verdadero habilita conteos
gmoccapy.spindle.reset-spindle-override	HAL_BIT	restablece husillo al 100 %
gmoccapy.rapid.rapid_override.counts	HAL_S32	Velocidad máxima de la máquina
gmoccapy.rapid.rapid_override.count-enable	HAL_BIT	verdadero habilita conteos

Para conectar *potenciómetros*, use los siguientes pines hal:

PIN	TIPO	FUNCION
gmoccapy.jog.jog-velocity.direct-value	HAL_FLOAT	velocidad Jog
gmoccapy.jog.jog-velocity.analog-enable	HAL_BIT	verdadero habilita analogico
gmoccapy.feed.feed-override.direct-value	HAL_FLOAT	porcentaje de alimentacion
gmoccapy.feed.feed-override.analog-enable	HAL_BIT	verdadero habilita analogico
gmoccapy.spindle.spindle-override.direct-value	HAL_FLOAT	porcentaje del husillo
gmoccapy.spindle.spindle-override.analog-enable	HAL_BIT	verdadero habilita analogico
gmoccapy.rapid.rapid_override.direct-value	HAL_FLOAT	Velocidad máxima de la máquina
gmoccapy.rapid.rapid_override.analog-enable	HAL_BIT	verdadero habilita analogico

Además gmoccapy 3 ofrece pines hal adicionales para controlar los nuevos widgets deslizantes con pulsadores. Los valores qué daran la rapidez del aumento o disminución deben establecerse en el archivo glade. En un lanzamiento futuro esto estara integrado en la página de configuración.

PIN	TIPO	FUNCION
VELOCIDAD		
gmoccapy.spc_jog_vel.increase	HAL_BIT IN	Si True, el valor del control deslizante aumentará
gmoccapy.spc_jog_vel.decrease	HAL_BIT IN	Si True, el valor del control deslizante disminuira
gmoccapy.spc_jog_vel.scale	HAL_FLOAT IN	Escalado del valor de salida (util en udes/min a udes/seg.)
gmoccapy.spc_jog_vel.value	HAL_FLOAT OUT	valor del widget
gmoccapy.spc_jog_vel.scaled-value	HAL_FLOAT OUT	valor escalado del widget

PIN	TIPO	FUNCION
ALIMENTACION		
gmoccapy.spc_feed.increase	HAL_BIT IN	Si True, el valor del control deslizante aumentará
gmoccapy.spc_feed.decrease	HAL_BIT IN	Si True, el valor del control deslizante disminuira
gmoccapy.spc_feed.scale	HAL_FLOAT IN	Escalado del valor de salida (util en udes/min a udes/seg.)
gmoccapy.spc_feed.value	HAL_FLOAT OUT	valor del widget
gmoccapy.spc_feed.scaled-value	HAL_FLOAT OUT	valor escalado del widget
HUSILLO		
gmoccapy.spc_spindle.increase	HAL_BIT IN	Si True, el valor del control deslizante aumentará
gmoccapy.spc_spindle.decrease	HAL_BIT IN	Si True, el valor del control deslizante disminuira
gmoccapy.spc_spindle.scale	HAL_FLOAT IN	Escalado del valor de salida (util en udes/min a udes/seg.)
gmoccapy.spc_spindle.value	HAL_FLOAT OUT	valor del widget
gmoccapy.spc_spindle.scaled-value	HAL_FLOAT OUT	valor escalado del widget
RAPIDOS		
gmoccapy.spc_rapid.increase	HAL_BIT IN	Si True, el valor del control deslizante aumentará
gmoccapy.spc_rapid.decrease	HAL_BIT IN	Si True, el valor del control deslizante disminuira
gmoccapy.spc_rapid.scale	HAL_FLOAT IN	Escalado del valor de salida (util en udes/min a udes/seg.)
gmoccapy.spc_rapid.value	HAL_FLOAT OUT	valor del widget
gmoccapy.spc_rapid.scaled-value	HAL_FLOAT OUT	valor escalado del widget

Los pines float aceptan valores de 0.0 a 1.0, siendo el valor de porcentaje que desea establecer como valor del control deslizante

[AVISO] Si usa ambos tipos de conexión, no conecte el mismo control deslizante a ambos pines, ya que las influencias entre los dos no han sido probadas. Diferentes deslizadores pueden estar conectados a uno u otro tipo de conexión hal.

[IMPORTANT] Tenga en cuenta que la velocidad de jog depende del estado del botón tortuga, con diferentes escalas deslizantes dependiendo del modo (tortuga o conejo). Por favor, eche un vistazo también a velocidades jog y pin hal jog-tortuga para más detalles

Ejemplo

```
Spindle Override Min Value =  20 %
Spindle Override Max Value = 120 %
gmoccapy.analog-enable = 1
gmoccapy.spindle-override-value = 0.25

value to set = Min Value + (Max Value - Min Value) * gmoccapy.spindle-override-value
value to set = 20 + (120 - 20) * 0.25
value to set = 45 %
```

4.2.5.3. Pines Jog Hal

Todos los ejes dados en el archivo INI tienen un pin jog-plus y un jog-minus, por lo que se pueden utilizar interruptores momentáneos de hardware para mover el eje.

nota
el nombre de este pin ha cambiado para Gmoccapy 3

Para la configuración estándar de XYZ, los siguientes pines estarán disponibles:

- gmoccapy.jog.axis.jog-x-plus
- gmoccapy.jog.axis.jog-x-minus
- gmoccapy.jog.axis.jog-y-plus
- gmoccapy.jog.axis.jog-y-minus
- gmoccapy.jog.axis.jog-z-plus
- gmoccapy.jog.axis.jog-z-minus

Si usa un archivo INI de 4 ejes, habrá dos pines adicionales

- gmoccapy.jog.<cuarta letra de eje>-plus
- gmoccapy.jog.< cuarta letra de eje>-minus

Para un eje "C" se veria:

- gmoccapy.jog.axis.jog-c-plus
- gmoccapy.jog.axis.jog-c-minus

4.2.5.4. Velocidades Jog y pin hal jog-tortuga

La velocidad de jog se puede seleccionar con el control deslizante correspondiente. La escala del control deslizante se modificará si el botón tortuga (el que muestra un conejo o una tortuga) ha sido pulsado. Si el botón no es visible, podría haber sido desactivado en la página de configuración. Si el botón muestra el icono de conejo, la escala de velocidad de la máquina es de min. a máx. . Si muestra una tortuga, la escala alcanzará solo 1/20 de velocidad máxima por defecto. La fraccion se puede configurar en la página de configuración.

Con ello, el usar una pantalla táctil es mucho más fácil al seleccionar velocidades más pequeñas.

gmoccapy ofrece un pin hal para alternar entre jogs tortugas y conejos

- gmoccapy.jog.turtle-jog (Hal Bit In)

4.2.5.5. Pines hal de incremento de jog

Los incrementos de jog se pueden seleccionar a través de pines hal, por lo que un interruptor hardware de selección se puede usar para seleccionar el incremento a usar. Habrá un máximo de 10 pines hal para los incrementos dados en el archivo INI; si da más incrementos en su archivo INI, no serán accesibles desde la GUI ya que no se mostrarán.

nota
Gmoccapy 3 utiliza diferentes nombres de pin hal

Si tiene 6 incrementos en su hal, obtendrá * 7 * pines: jog-inc-0 no se puede cambiar y representará jogging continuo.

- gmoccapy.jog.jog-inc-0

- gmoccapy.jog.jog-inc-1
- gmoccapy.jog.jog-inc-2
- gmoccapy.jog.jog-inc-3
- gmoccapy.jog.jog-inc-4
- gmoccapy.jog.jog-inc-5
- gmoccapy.jog.jog-inc-6

4.2.5.6. Pin de desbloqueo hardware

Para poder usar un interruptor de llave para desbloquear la página de configuración, se exporta siguiente pin:

- gmoccapy.unlock-settings

La página de configuración está desbloqueada si el pin está alto. Para usar este pin, debes activarlo en la página de configuración.

4.2.5.7. Pines de error

- gmoccapy.error
- gmoccapy.delete-message

gmoccapy.error es un pin de salida, para indicar un error, por lo que una luz puede encenderse o incluso la máquina puede ser detenida. Se resetea con el pin gmoccapy.delete-message, que elimina el primer error y restablece el pin gmoccapy.error en False una vez que se haya borrado el último error.

NOTA: Los mensajes o las informaciones de usuario no afectarán el pin gmoccapy.error, pero el mensaje gmoccapy.delete pin borrará el último mensaje si no se muestra ningún error!

4.2.5.8. Pines HAL de Mensajes creados por el usuario

gmoccapy puede reaccionar a errores externos, utilizando 3 mensajes de usuario diferentes (todos son HAL_BIT):

Estado

- gmoccapy.messages.statustest

dialogo-YesNo

- gmoccapy.messages.yesnodialog
- gmoccapy.messages.yesnodialog-waiting
- gmoccapy.messages.yesnodialog-responce

dialogo-Ok

- gmoccapy.messages.okdialog
- gmoccapy.messages.okdialog-waiting

Para agregar un mensaje creado por el usuario, debe agregar el mensaje al archivo INI en el Sección [DISPLAY]. Aquí hay un par de ejemplos.

```
MESSAGE_BOLDTEXT = FALLO EN SISTEMA DE LUBRICACION
MESSAGE_TEXT = Fallo de lubricación
MESSAGE_TYPE = okdialog
MESSAGE_PINNAME = lube-fault

MESSAGE_BOLDTEXT = NONE
MESSAGE_TEXT = X SHEAR PIN BROKEN
MESSAGE_TYPE = status
MESSAGE_PINNAME = pin
```

Para *conectar* los nuevos pines debe hacerlo en el archivo HAL Postgui. Aquí hay algunos ejemplos de conexiones que tienen la señal conectada a una entrada en algún otro lugar en el archivo HAL.

```
net gmoccapy-lube-fault gmoccapy.messages.lube-fault
net gmoccapy-lube-fault-waiting gmoccapy.messages.lube-fault-waiting
red gmoccapy-pin gmoccapy.messages.pin
```

Para obtener más información sobre los archivos HAL y el comando net, consulte la referencia HAL básica.

4.2.5.9. Pines de realimentación del husillo

Hay dos pines para la retroalimentación del husillo.

- gmoccapy.spindle_feedback_bar
- gmoccapy.spindle_at_speed_led

gmoccapy.spindle_feedback_bar aceptará una entrada float para mostrar la velocidad del husillo. *gmoccapy.spindle_at_speed_led* es un pin de bit para encender el led GUI si el husillo está a su velocidad.

4.2.5.10. Pines para indicar información sobre el progreso del programa

Hay tres pines que dan información sobre el progreso del programa:

- gmoccapy.program.length, HAL_S32, que muestra el número total de líneas de programa
- gmoccapy.program.current-line, HAL_S32, que indica la línea de trabajo actual del programa
- gmoccapy.program.progress, HAL_FLOAT, que da el progreso del programa en porcentaje

Los valores pueden no ser muy precisos, si está trabajando con subrutinas o procedimientos de remapeo grandes; también los bucles causarán diferentes valores.

4.2.5.11. Pines relacionado con la herramienta

Pin de cambio de herramienta Este pin se proporciona para usar el diálogo de cambio de herramienta interno de gmoccapy, similar al conocido de AXIS, pero con varias modificaciones. No solo le mostrara el mensaje para cambiar *número de herramienta 3*, sino también la descripción de esa herramienta, como *Fresa 7,5 mm 3 filos*. La información se toma de la tabla de herramientas, por lo que depende de usted lo qué se muestre.

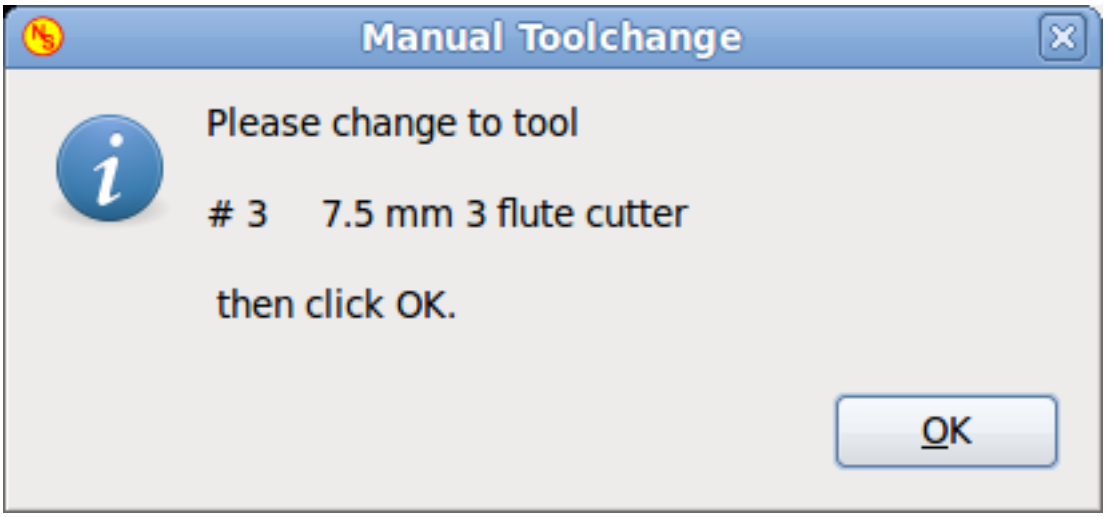


Figura 4.12: Cambio manual de herramienta

- gmoccapy.toolchange-number, HAL_S32, El número de la herramienta que se va a cambiar
- gmoccapy.toolchange-change, HAL_BIT, Indica que se debe cambiar una herramienta
- gmoccapy.toolchange-modified, HAL_BIT, Indica que la herramienta ha cambiado

Por lo general, se conectan así para un cambio de herramienta manual:

```
net tool-change gmoccapy.toolchange-change <= iocontrol.0.tool-change
net tool-changed gmoccapy.toolchange-changed <= iocontrol.0.tool-changed
net tool-prep-number gmoccapy.toolchange-number <= iocontrol.0.tool-prep-number
net tool-prep-loop iocontrol.0.tool-prepare <= iocontrol.0.tool-prepared
```

Pines de offsets de herramientas Estos pines le permiten mostrar los valores de offset de la herramienta activa para X y Z en el cuadro de información de la herramienta. Solo están activos después de G43.

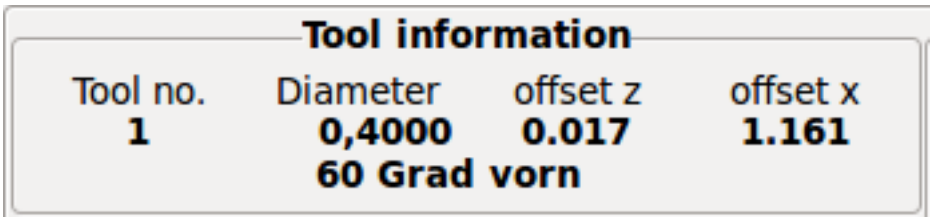


Figura 4.13: Informacion de herramienta

- gmoccapy.tooloffset-x
- gmoccapy.tooloffset-z

Conéctelos en su postgui hal.

nota
la línea tooloffset-x no es necesaria en una fresadora, y no se mostrará en una fresadora con cinemática trivial.

```
net tooloffset-x gmoccapy.tooloffset-x <= motion.tooloffset.x
net tooloffset-z gmoccapy.tooloffset-z <= motion.tooloffset.z
```

Tenga en cuenta que gmoccapy se encarga de actualizar los offsets, enviando un G43 después de cualquier cambio de herramienta, **pero no en modo automático!**

importante
Al escribir un programa, usted es responsable de incluir un G43 después cada cambio de herramienta!

4.2.6. Medición automática de herramientas

Gmoccapy ofrece una medición automática integrada de herramientas. Para usar esta característica, usted tendrá que hacer algunos ajustes adicionales y es posible que desee utilizar el pin hal que se ofrece para obtener valores en su propio procedimiento ngc de remapeo.

[IMPORTANT] Antes de comenzar la primera prueba, no olvide ingresar altura y velocidades de la sonda en la página de configuración! Ver página de configuración de medición de Herramienta

También podría ser una buena idea echar un vistazo a la herramienta de medición en video: ver videos relacionados con la medición de herramientas

La medición de herramientas en gmoccapy se realiza de forma un poco diferente a muchas otras GUI. Debe seguir estos pasos:

- toque de su pieza en X e Y
- mida la altura de su bloque desde la base donde está ubicado su interruptor de herramienta a la cara superior del bloque (incluida mordaza, etc.)
- Presione el botón de altura del bloque e ingrese el valor medido
- Ir al modo automático y comenzar su programa

Aquí hay un pequeño boceto:

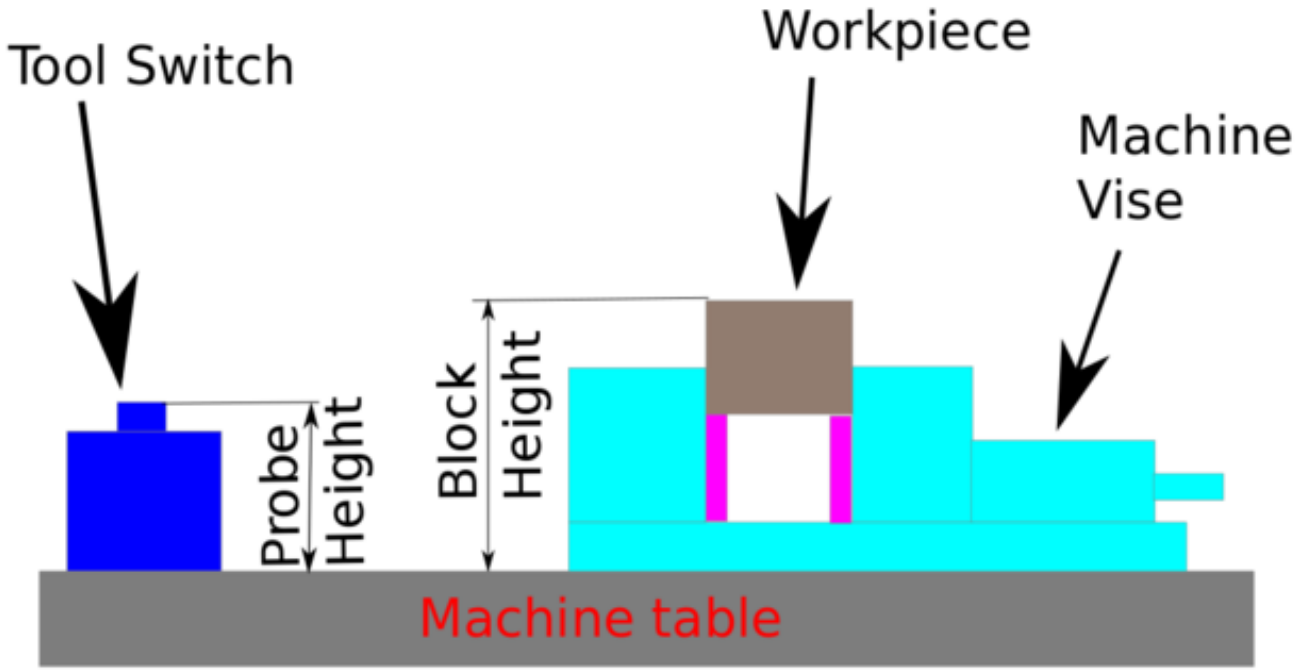


Figura 4.14: Datos para medicion de herramientas

Con el primer cambio de herramienta, la herramienta se medirá y el offset se configura automáticamente para adaptarse a la altura del bloque. La ventaja de gmoccapy es que no se necesita una herramienta de referencia.

nota
¡Su programa debe contener un cambio de herramienta al principio! La herramienta será medida, incluso se ha utilizado antes, por lo que no hay peligro si la altura del bloque ha cambiado. Hay varios videos que muestran la manera de hacerlo en Youtube.

4.2.6.1. Pines de medición de herramientas

Gmoccapy ofrece 5 pines para fines de medición de herramientas. Los pines se utilizan principalmente para leer desde una subrutina de gcode, para que el código pueda reaccionar a diferentes valores.

- gmoccapy.toolmeasurement, HAL_BIT, permitir o no medición de herramientas
- gmoccapy.blockheight, HAL_FLOAT, el valor medido de la cara superior de la pieza de trabajo
- gmoccapy.probeheight, HAL_FLOAT, la altura del interruptor de la sonda
- gmoccapy.searchvel, HAL_FLOAT, la velocidad para buscar el interruptor de la sonda de herramienta
- gmoccapy.probevel, HAL_FLOAT, la velocidad para sondear la longitud de la herramienta

4.2.6.2. Modificaciones de archivos INI en medición de herramienta

Modifique su archivo INI para incluir lo siguiente:

En la sección [RS274NGC]

```
[RS274NGC]
# Habilita lectura de valores INI y HAL desde gcode
FEATURES=12

# sub que se llama cuando ocurre un error durante el cambio de herramienta
ON_ABORT_COMMAND=O <on_abort> call

# El código de remapeo
REMAP=M6  modalgroup=6 prolog=change_prolog ngc=change epilog=change_epilog
```

La seccion del sensor de herramienta Todos los valores de la posición del sensor de la herramienta y la posición inicial del movimiento de sondeo son coordenadas absolutas, excepto MAXPROBE, que debe darse en movimiento relativo.

```
[TOOLSENSOR]
X = 10
Y = 10
Z = -20
MAXPROBE = -20
```

La sección de cambio de posición Esto no se ha llamado TOOL_CHANGE_POSITION a propósito - **canon usa ese nombre e interferiria.**. Es la posición a donde mover la máquina antes de dar el comando para cambiar la herramienta. Todos los valores están en coordenadas absolutas.

```
[CHANGE_POSITION]
X = 10
Y = 10
Z = -2
```

La Sección de Python. Los complementos de Python sirven como intérprete y tarea.

```
[PYTHON]
# La ruta para iniciar una búsqueda de módulos de usuario.
PATH_PREPEND = python
# El punto de inicio para todos.
TOPLEVEL = python/toplevel.py
```

4.2.6.3. Archivos Necesarios

Debe copiar los siguientes archivos a su directorio de configuración

Primero cree un directorio *python* en su carpeta de configuración. Desde *su_directorio_linuxcnc-dev/configs/sim/gmoccapy/python* copie: *toplevel.py*, *remap.py* y *stdglue.py* a su carpeta *config_dir / python*.

Desde *su_directorio_linuxcnc-dev/configs/sim/gmoccapy/macros* copie: *on_abort.ngc* y *change.ngc* al directorio especificado como SUBROUTINE_PATH. Vea Seccion RS274NGC.

Abra *change.ngc* con un editor y descomente las siguientes líneas (49 y 50):

```
F #<_ hal [gmoccapy.probevel]>
G38.2 Z-4
```

Es posible que desee modificar este archivo para que se ajuste más a sus necesidades.

4.2.6.4. Conexiones Hal necesarias

Conecte la sonda de la herramienta en su archivo hal así:

```
net probe  motion.probe-input <= <your_input_pin>
```

La línea podría verse así:

```
 net probe  motion.probe-input <= parport.0.pin-15-in
```

En su archivo postgui.hal agregue:

```
# Las siguientes líneas solo son necesarias si los pines se habían conectado antes
unlinkp iocontrol.0.tool-change
unlinkp iocontrol.0.tool-changed
unlinkp iocontrol.0.tool-prep-number
unlinkp iocontrol.0.tool-prepared

# enlace a gmoccapy toolchange, para obtener la ventaja de la descripción en el cuadro de ↩
    diálogo de cambio de herramienta
net tool-change gmoccapy.toolchange-change <= iocontrol.0.tool-change
net tool-changed gmoccapy.toolchange-changed <= iocontrol.0.tool-changed
net tool-prep-number gmoccapy.toolchange-number <= iocontrol.0.tool-prep-number
net tool-prep-loop iocontrol.0.tool-prepare <= iocontrol.0.tool-prepared
```

4.2.7. La página de configuración

Para ingresar a la página deberás hacer click en y dar un código de desbloqueo, que es * 123 * por defecto. Si quieres cambiarlo en este momento tendrá que editar el archivo de preferencias oculto. Vea la sección de pantalla para más detalles.

La página se ve así:

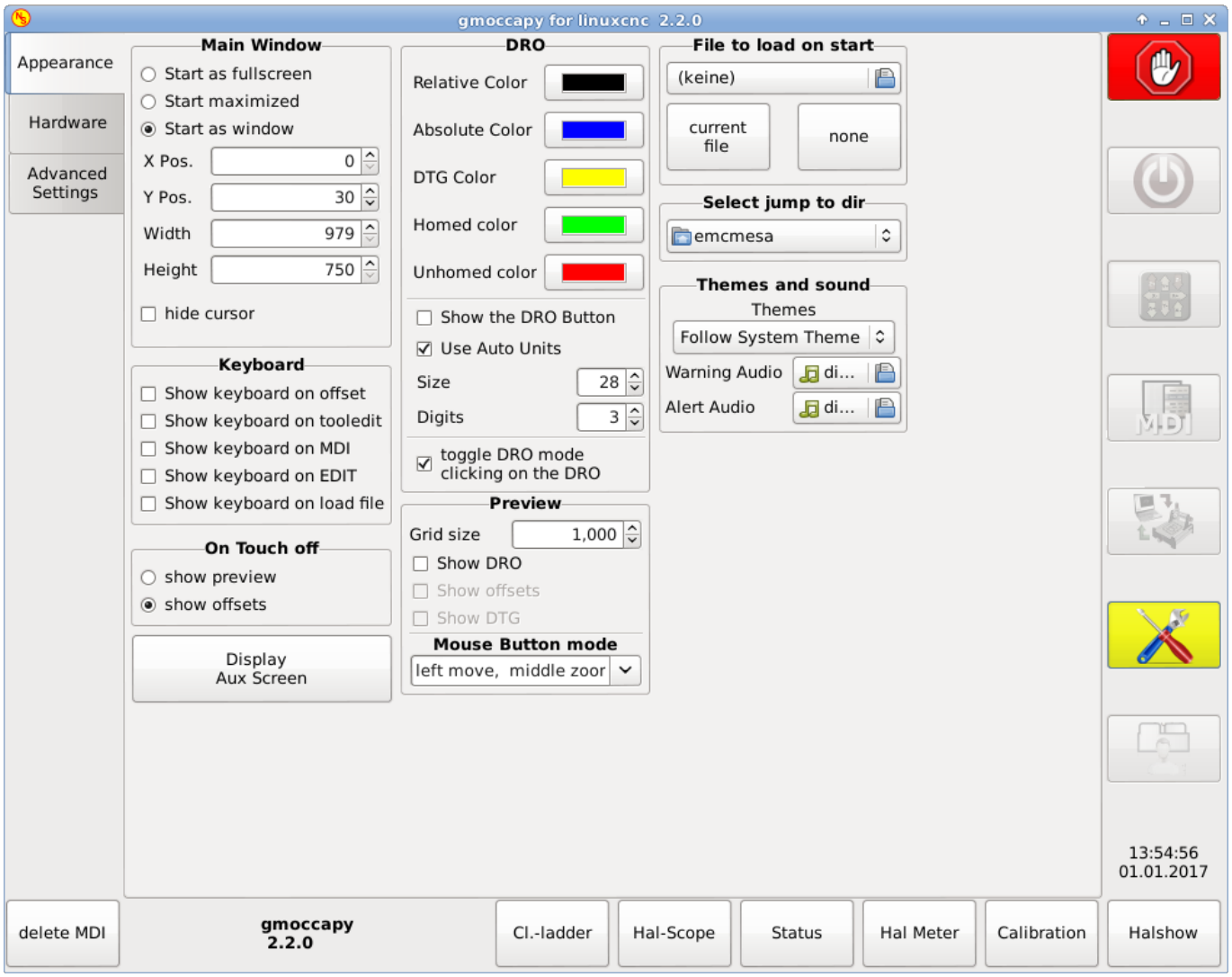


Figura 4.15: Pagina de configuracion

La página está separada en tres pestañas principales:

4.2.7.1. Apariencia

En esta pestaña encontrará las siguientes opciones:

Ventana principal

Aquí puede seleccionar cómo desea que comience la GUI. La razón principal de esto fue el deseo de obtener una forma facil para el usuario de configurar las opciones de inicio sin la necesidad de tocar código.

Tienes tres opciones:

- iniciar como pantalla completa
- iniciar maximizado
- iniciar como ventana
  Si selecciona Iniciar como ventana, se activarán los cuadros para establecer la posición y el tamaño.

Una vez establecidos, la GUI se iniciará cada vez en el lugar y con el tamaño seleccionado. Sin embargo, el usuario puede cambiar el tamaño y la posición con el mouse, pero eso no tiene ninguna influencia en la configuración.

* *Ocultar cursor* * permite ocultar el cursor, lo que es muy útil si utilizar una pantalla táctil.

Teclado

Las casillas de verificación permiten al usuario seleccionar si desea que el teclado integrado se muestre de inmediato, al ingresar al modo MDI, al ingresar a la página de offsets, al widget tooledit o al abrir un programa en el modo EDITAR. El botón del teclado en la lista de botones inferior no se verá afectado por esta configuración, para que pueda mostrar u ocultar el teclado presionando el botón. El comportamiento por defecto será establecido por las casillas de verificación.

Los valores predeterminados son:

---

nota

Si esta sección no es confidencial, no se ha instalado un teclado virtual; + los admitidos son *onboard* y *matchbox-keyboard*.

- Mostrar teclado en offset = Verdadero
- Mostrar teclado en tooledit = False
- Mostrar teclado en MDI = Verdadero
- Mostrar teclado en EDITAR = Verdadero
- Mostrar teclado en carga de = Falso

Si la distribución del teclado no es correcta, es decir, al hacer clic en X se obtiene Z, el diseño no se ha establecido correctamente, en relacion con la configuración regional. Para *onboard* se puede resolver con un pequeño archivo por lotes con el siguiente contenido:

```
#!/bin/bash
setxkbmap -model pc105 -layout XX -variant basic
```

Las letras XX se tendrán que configurar de acuerdo con su configuración regional (para español de España, **es**). Simplemente ejecute este archivo antes de iniciar LinuxCNC. Tambien puede agregar un starter a su carpeta local

```
./config/autostart
```

para que el diseño se establezca automáticamente en el inicio.

Para matchbox-keyboard tendrá que hacer su propio diseño.

On Touch Off

da la opción de mostrar la pestaña de vista previa o de la página de offsets si ingresa en el modo touch off haciendo clic en botón inferior correspondiente.

- Mostrar vista previa
- Mostrar offsets

A medida que se muestran las pestañas, puede cambiar entre ambas vistas en cualquier caso.

Mostrar pantalla auxiliar

Al hacer clic en este botón se abrirá una ventana adicional. Este botón solo es sensible si un archivo llamado *Gmoccapy 3.glade* se encuentra en su carpeta de configuración. Puedes construir la pantalla Aux usando Glade.

aviso
La ventana principal de la pantalla auxiliar debe llamarse window2

Opciones de DRO

Tiene la opción de seleccionar los colores de fondo de los diferentes estados del DRO. Por lo tanto, los usuarios que sufren de protanopia (debilidad roja/verde) pueden seleccionar los colores adecuados

Por defecto los fondos son:

- Modo relativo = negro
- Modo absoluto = azul
- Distancia a recorrer = amarillo

El color de primer plano del DRO se puede seleccionar con:

- homed color = verde
- unhomed color = rojo

mostrar DRO en vista previa
el DRO se mostrará en la ventana de vista previa +

mostrar los offsets
Los offsets se mostrarán en la ventana de vista previa +

mostrar DTG
La distancia a recorrer se mostrará en la ventana de vista previa +

mostrar boton DRO
le permitirá mostrar botones adicionales en el lado izquierdo del DRO.

Se mostrará:
* Un botón para cambiar de coordenadas relativas a absolutas,
* un botón para alternar entre la distancia a recorrer y los otros estados
* y un botón para alternar las unidades de métricas a imperiales y viceversa.

aviso
No se recomienda usar esta opción, porque el usuario puede perder la opción de unidad automática, que alternará las unidades según el gcode activo G20/G21.

nota
Puede cambiar a través de los modos DRO (absoluto, relativo, distancia a recorrer) haciendo clic en el DRO!

Usar unidades automáticas
permite deshabilitar la opción de unidades automáticas de la pantalla, para que pueda ejecutar un programa en pulgadas y ver el

DRO en mm. +

tamaño
permite configurar el tamaño de la fuente DRO; el valor predeterminado es 28; si usa una pantalla más grande, es posible que desee aumentar el tamaño hasta 56. Si utiliza 4 ejes, el tamaño de fuente DRO será 3/4 del valor, debido a razones de espacio. +

dígitos
establece el número de dígitos del DRO de 1 a 5.

nota
Imperial mostrará un dígito más que métrico; si está en unidades de máquinas imperiales y establece el valor de dígito en 1, no obtendrá ningún dígito en métrico.

cambiar modo DRO
Si no está activo, un clic del ratón en el DRO no realizará ninguna acción.
De forma predeterminada, esta casilla de verificación está activa, por lo que cada clic en cualquier DRO cambiará la lectura DRO de real a relativa a DTG (distancia a recorrer). +

Vista previa

Tamaño de cuadrícula Establece el tamaño de cuadrícula de la ventana de vista previa. Desafortunadamente, el tamaño

- debe establecerse en pulgadas *, incluso si las unidades de su máquina son métricas. Esperamos arreglar eso en un futuro lanzamiento.

nota
La cuadrícula no se mostrará en la vista en perspectiva.

Mostrar DRO
Mostrará el DRO también en la ventana de vista previa. Se mostrará automáticamente en vista previa a tamaño completo

Mostrar DTG mostrará también la DTG (distancia al punto final) en la vista previa, solo si *Mostrar DRO* está activo y no es vista previa a tamaño completo.

Mostrar offsets mostrará los offsets en la ventana de vista previa.

nota
Si solo marca esta opción y deja las otras sin marcar, obtendra una vista previa a tamaño completo de la página de offsets

Modo de botón del ratón este cuadro combinado puede seleccionar el comportamiento del botón de ratón para girar, mover o hacer zoom dentro de la vista previa. Las combinaciones posibles, con el orden de botones izquierdo-medio-derecho son:

- girar, mover, zoom
- zoom, mover, girar
- mover, girar, zoom
- zoom, girar, mover
- mover, zoom, girar
- girar, zoom, mover

El valor predeterminado es mover, zoom , girar.

La rueda del ratón seguirá ampliando la vista previa en todos los modos.

sugerencia
Si selecciona un elemento en la vista previa, el elemento seleccionado será tomado como punto central de rotación.

Archivo para cargar en el inicio

Seleccione el archivo que desea cargar en el inicio. En otras GUI, cambiar esto es muy engorroso, porque los usuarios se ven obligados a editar el archivo INI.

Seleccione el archivo que desea cargar en el inicio. Si se carga un archivo, esto puede configurarse presionando el botón actual para evitar que cualquier programa se cargue en Inicia, solo presiona el botón Ninguno.

La pantalla de selección de archivos utilizará los filtros que haya configurado en el archivo INI. Si no hay filtros, solo verá los archivos **ngc**. El camino se establecerá de acuerdo con la configuración de INI en [DISPLAY]PROGRAM_PREFIX

Saltar a dir

Puede configurar aquí el directorio a donde saltar si presiona el botón correspondiente en el cuadro de diálogo de selección de archivos.

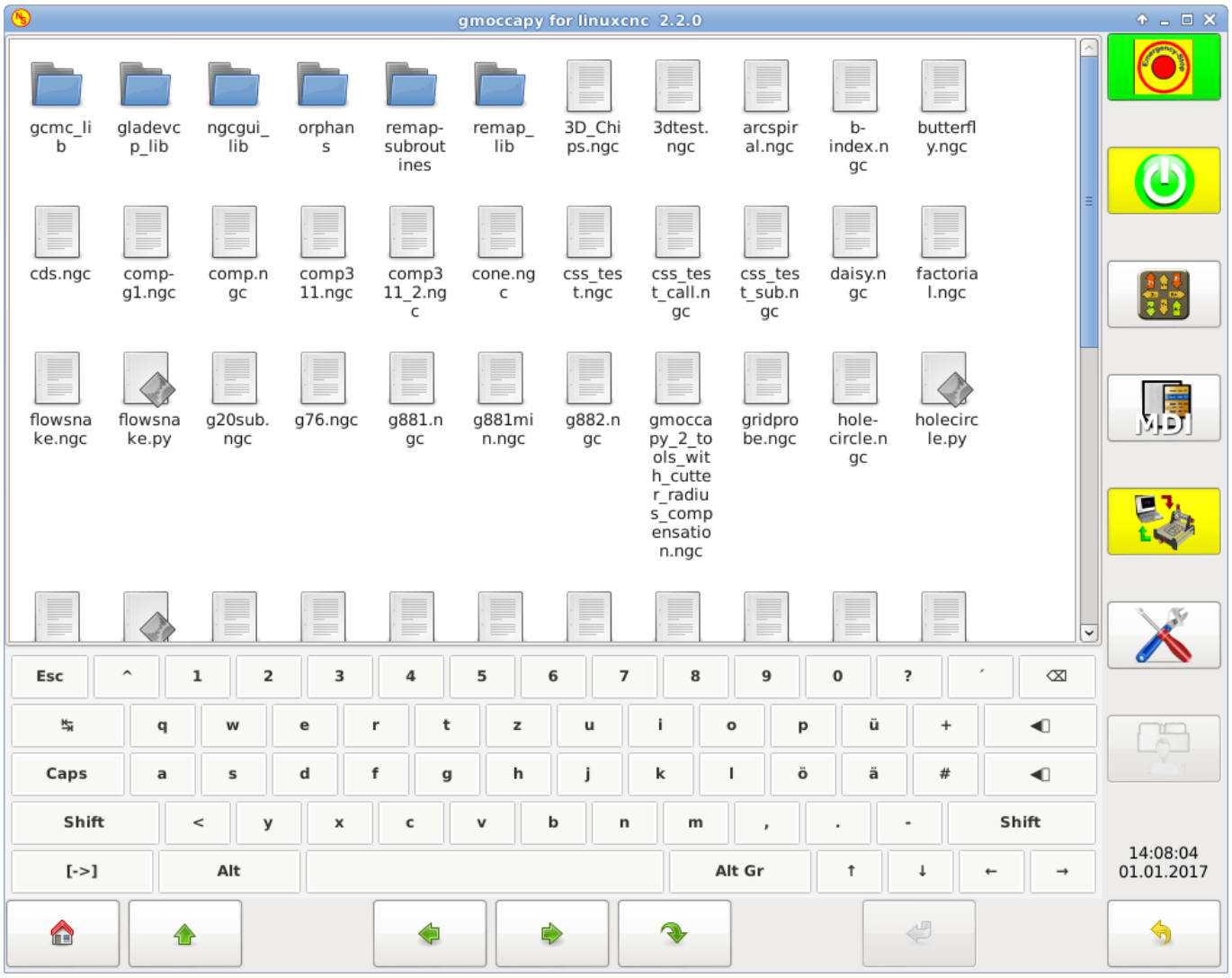
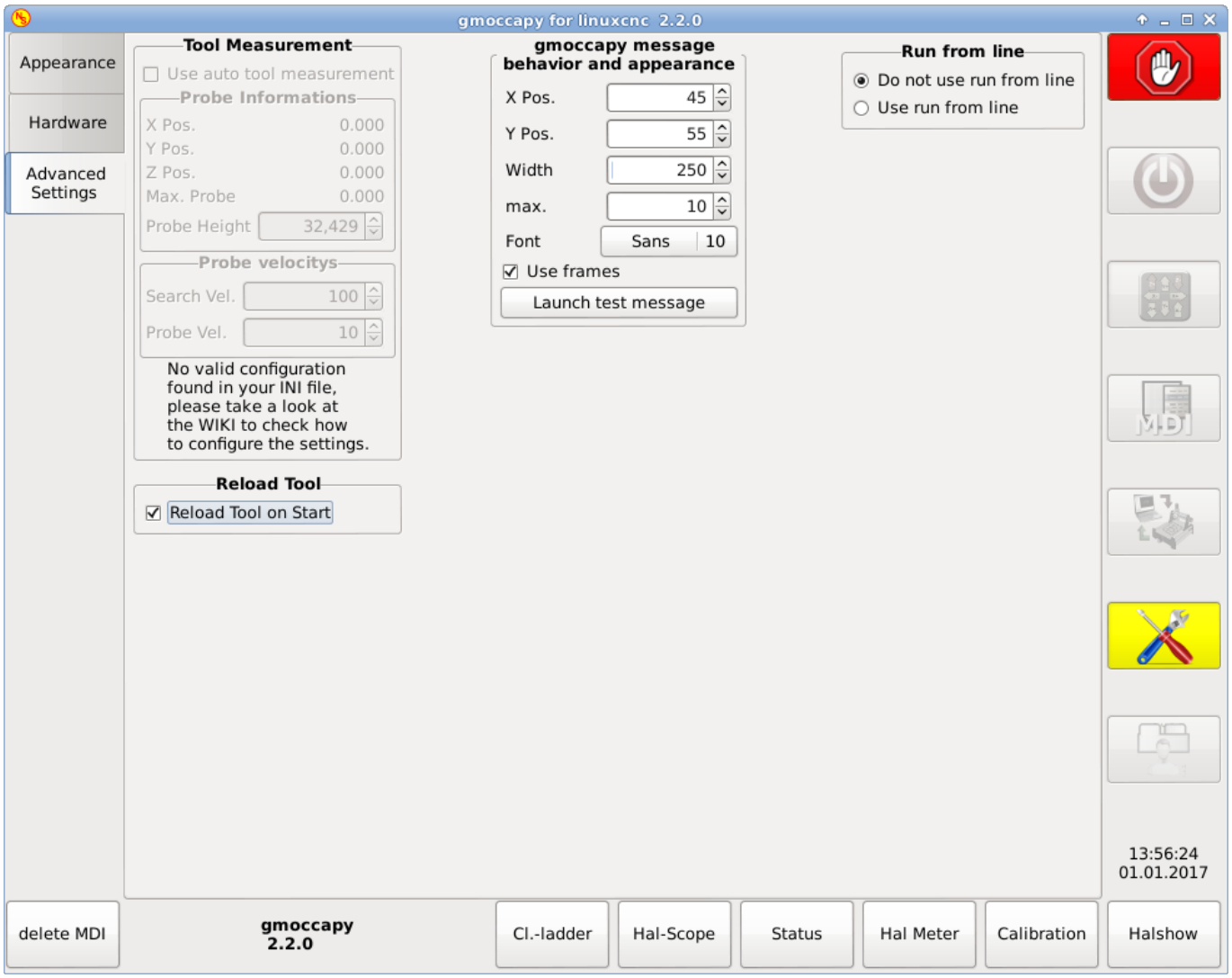


Figura 4.16: Seleccion de directorio

Temas y sonidos

Esto le permite al usuario seleccionar qué tema de escritorio aplicar y qué sonidos de error y mensajes deben reproducirse. Por defecto está establecido "Seguir tema del sistema".

4.2.7.2. Hardware



Escalas de hardware MPG

Para los diferentes pines Hal se conectan a los volantes MPG, puede seleccionar escalas individuales. La razón principal de esto fue mi propia prueba para resolver esto a través de conexiones hal, resultando en un archivo hal muy complejo. Imagine a un usuario que tiene un volante MPG con 100 ipr y quiere reducir la velocidad máxima de 14000 a 2000 mm/min; necesita 12000 impulsos!, lo que da como resultado 120 giros del volante! O a otro usuario que tiene un volante MPG con 500 ipr y quiere configurar el porcentaje del husillo, que tiene limites de 50 a 120 %; de mín a máx solo van 70 impulsos, lo que significa que no llega ni a 1/4 de vuelta.

Por defecto, todas las escalas se establecen utilizando el cálculo:

```
(MAX - MIN)/100
```

Atajos de teclado

Algunos usuarios desean hacer jog usando los botones del teclado y hay otros que nunca lo permitirán. Así que todos pueden elegir si usarlos o no.

Por defecto, se usan los atajos de teclado.

Por favor, tenga cuidado si usa un torno, ya que los atajos serán diferentes. Ver la sección de Torno

- Flecha izquierda o NumPad_Left = X menos
- Flecha derecha o NumPad_Right = X más
- Flecha arriba o NumPad_Up = Y más
- Flecha abajo o NumPad_Down = Y menos
- Page Up o NumPad_Page_Up = Z mas
- Page Down o NumPad_Page_Down = Z menos
- F1 = Estop (funcionará incluso si los atajos de teclado están desactivados)
- F2 = Máquina encendida
- F3 = Modo manual
- F5 = Modo MDI
- ESC = Abortar

Hay teclas adicionales para el manejo de mensajes, ver comportamiento y apariencia del mensaje

- WINDOWS = Borrar el último mensaje
- <CTRL><SPACE> = Borrar todos los mensajes

Desbloquear opciones

Hay tres opciones para desbloquear la página de configuración:

- usar un código de desbloqueo (el usuario debe dar un código para entrar)
- No usar código de desbloqueo (no habrá verificación de seguridad)
- Usar un pin para desbloquear (el pin de hardware debe estar alto para desbloquear la configuración, ver pin de desbloqueo hardware)

El valor predeterminado es usar el código de desbloqueo (predeterminado = * 123 *)

Husillo

RPM establece las rpm que se utilizarán si el husillo se inicia y no se ha establecido ningún valor de S.

---

nota

Este valor será predefinido de acuerdo con su configuración en [DISPLAY]DEFAULT_SPINDLE_SPEED de su INI. Si cambia la configuración en la página de configuración, ese valor será el predeterminado desde ese momento; su archivo INI no sera modificado

---

Con los ajustes MÍN y MÁX, usted establece los límites de la barra de husillo mostrada en el cuadro de información en la pantalla principal. No es un error dar valores erróneos. Si da un máximo de 2000 y su eje llega a 4000 rpm, solo el nivel de barra estara equivocado en velocidades superiores a 2000 rpm.

```
los valores por defecto son
MIN = 0
MAX = 6000
```

Turtle Jog

Esta configuración tendrá influencia en las velocidades de jog.

- *Ocultar boton jog tortuga* ocultará el botón a la derecha del control deslizante de velocidad de jog. Si oculta este botón, tenga cuidado de que se muestre el icono conejo, de lo contrario no podrá hacer jog más rápido que la velocidad de tortuga, que se calcula utilizando el factor jog de tortuga.
- *factor de jog de tortuga* establece la escala para el modo jog tortuga. Si le pone un factor de 20, la velocidad máxima de jog será 1/20 de la velocidad máxima de la máquina si está en modo tortuga (botón presionado, mostrando la tortuga)

nota

Este botón se puede activar usando el pin hal turtle-jog.

4.2.7.3. Configuración avanzada

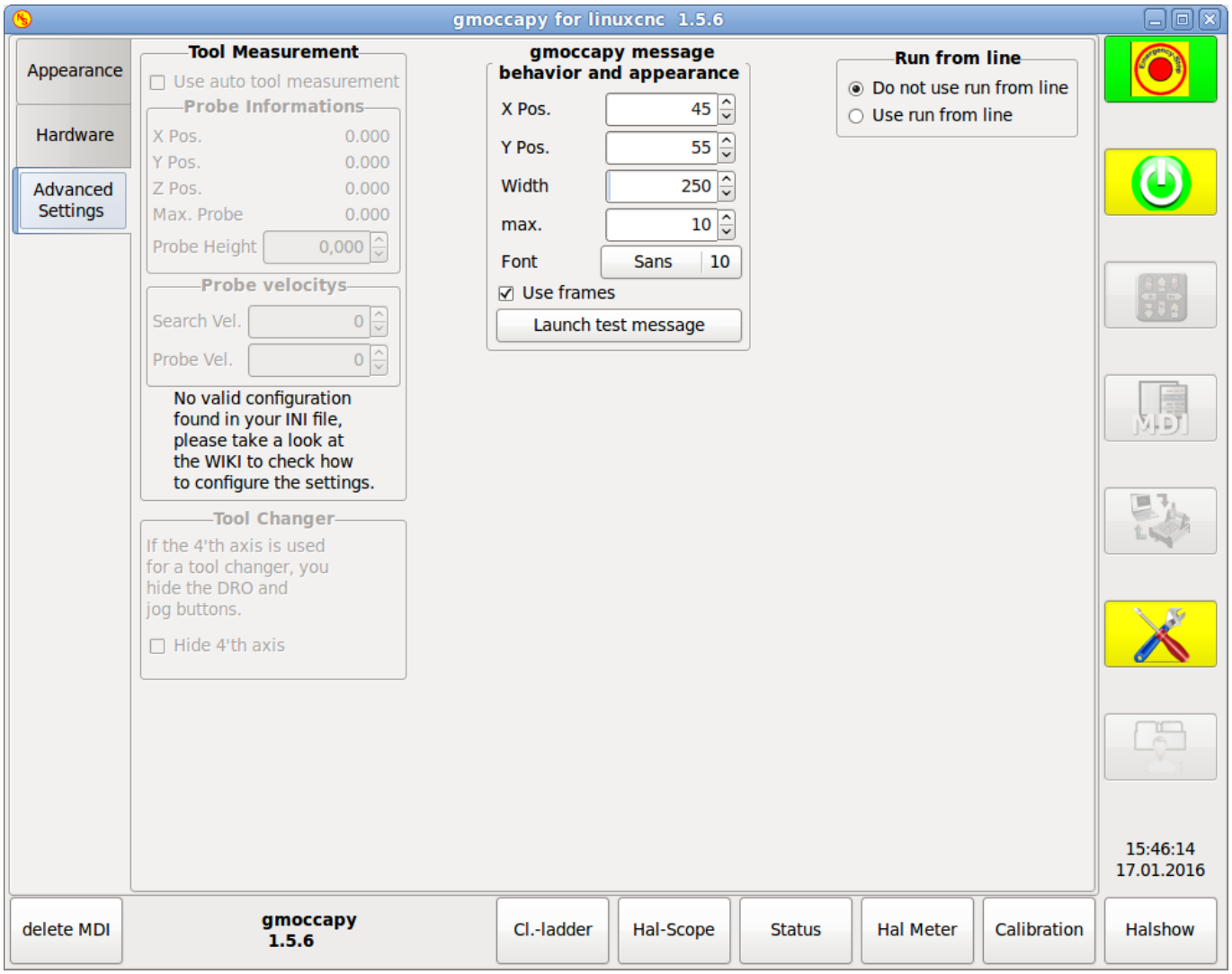


Figura 4.17: Configuracion avanzada

Medición de herramientas

Si esta parte no es sensible, no tiene una configuración de archivo INI válida para utilizar la medición de herramienta.

Verifique medición automática de herramientas

- Usar la medición automática de herramientas: si se marca, después de cada cambio de herramienta se realizará la medición. El resultado se almacenará en la tabla de herramientas y se ejecutará un G43 después del cambio.

Información de sonda Las siguientes informaciones se tomaran de las de su archivo INI y se deben proporcionar en coordenadas absolutas

- X Pos. = La posición X del interruptor de herramienta
- Y Pos. = La posición Y del interruptor de herramienta
- Z Pos. = La posición Z del interruptor de la herramienta, iremos como movimiento rápido a esta coordenada
- Max. Sonda = es la distancia para buscar contacto. Un error será lanzado si no se da contacto dentro de ella. La distancia tiene que ser dada en coordenadas relativas, comenzando el movimiento desde Z Pos., por lo que tiene que dar un valor negativo para bajar!
- Altura sonda = es la altura de su interruptor de sonda; puede medirlo. Solo toque la base donde se encuentra el interruptor de la sonda y ajústelo a cero. Luego haga un cambio de herramienta y observe el valor de tool_offset_z, que es el valor que debe entrar aquí.

LAS VELOCIDADES DE LA SONDA

- Vel. Búsqueda = La velocidad para buscar el interruptor. Después del contacto, la herramienta volverá a subir y luego volverá hacia la sonda con la velocidad de sonda, por lo que obtendrá mejores resultados.
- Vel. Sonda = Es la velocidad para el segundo movimiento hacia el interruptor, que debe ser más lento para obtener mejores resultados. (En modo sim, esto es comentado en macros/change.ngc, de lo contrario el usuario tendría que hacer clic dos veces en el botón de la sonda)

Cambiador de herramientas Si su cuarto eje 'se utiliza en un cambiador de herramientas, es posible que desee ocultar el DRO y todos los demás botones relacionados con ese eje.

Puedes hacerlo marcando la casilla de verificación, que ocultará:

- 4º eje DRO
- 4º eje boton Jog
- 4º eje botón de referencia (home)
- Columna del 4º eje en la página offset.
- Columna del 4º eje en el editor de herramientas.

Si se marca, la herramienta en el husillo se guardará en cada cambio en el archivo de preferencias, lo que hace posible volver a cargar la última herramienta montada en nuevo arranque. La herramienta se cargará después de que todos los ejes estén conectados, porque antes no está permitido ejecutar comandos MDI. Si usa NO_FORCE_HOMING no puedes usar esta característica, porque nunca se emitirá la señal *all_homed* necesaria.

Comportamiento y apariencia de mensajes.

Esto mostrará pequeñas ventanas emergentes que muestran el mensaje o el texto de error. El comportamiento es muy similar al que utiliza AXIS. Puede eliminar un determinado mensaje haciendo clic en el botón Cerrar. Si desea eliminar el último, simplemente presione la tecla WINDOWS en su teclado o borre todos los mensajes con <CTRL><SPACE>.

Puedes configurar algunas opciones:

- X Pos = La posición X de la esquina superior izquierda del mensaje en pixels desde la esquina superior izquierda de la pantalla.
- Y Pos = La posición Y de la esquina superior izquierda del mensaje en pixels desde la esquina superior izquierda de la pantalla.
- Ancho = El ancho del cuadro de mensaje
- max = el máximo de mensajes que desea ver. Si establece esto en 10, el mensaje número 11 eliminará el primero, por lo que solo verá los últimos 10.
- Fuente = la fuente y el tamaño que desea utilizar para mostrar los mensajes
- usar marcos = Si activa la casilla de verificación, se mostrará cada mensaje en un marco, por lo que es mucho más fácil distinguir los mensajes, pero se necesitara un poco más de espacio.
- Lanzamiento de mensaje de prueba solo hará lo que se supone que debe hacer, mostrar un mensaje, para que pueda ver los cambios de su configuración sin la necesidad para generar un error.

La opción Ejecutar desde línea Puede permitir o rechazar la ejecución desde línea. Esto pondra al correspondiente botón insensible (en gris), por lo que el usuario no podrá utilizar esta opción. El valor predeterminado es deshabilitar la ejecución desde la línea.

aviso
No se recomienda usar *Ejecutar desde línea*, ya que LinuxCNC no comprobara ninguna línea anterior en el código antes de la línea de inicio. Así, errores o choques son mas que probables

4.2.8. Sección específica del torno

Si en el archivo INI se da LATHE = 1, la GUI cambiará su apariencia a las necesidades especiales de un torno. Principalmente se ocultará el eje Y y Los botones de jog se ordenarán en un orden diferente.

Figura 4.18: Torno normal (herramienta delantera)

Figura 4.19: Torno con herramienta trasera

Como ve, el R DRO tiene un fondo negro y el de D DRO es gris. Esto puede cambiar de acuerdo con el código G7 o G8 activo. El modo activo es visible por el fondo negro, es decir, en las imágenes mostradas G8 está activo.

La siguiente diferencia a la pantalla estándar es la ubicación del botón Jog. X y Z han cambiado de lugar e Y ha desaparecido. Notese que los botones X+ y X- cambian de lugar según sea torno normal o trasero.

También cambiará el comportamiento del teclado:

Torno normal:

- Flecha izquierda o NumPad_Left = Z menos
- Flecha derecha o NumPad_Right = Z más
- Flecha arriba o NumPad_Up = X menos
- Flecha abajo o NumPad_Down = X más

Torno de herramienta trasera:

- Flecha izquierda o NumPad_Left = Z menos

- Flecha derecha o NumPad_Right = Z más
- Flecha arriba o NumPad_Up = X más
- Flecha abajo o NumPad_Down = X menos

El cuadro de información de la herramienta mostrará no solo el desplazamiento Z, sino también el offset X y la tabla de herramientas muestra toda la información relevante del torno.

4.2.9. Sección específica de plasma

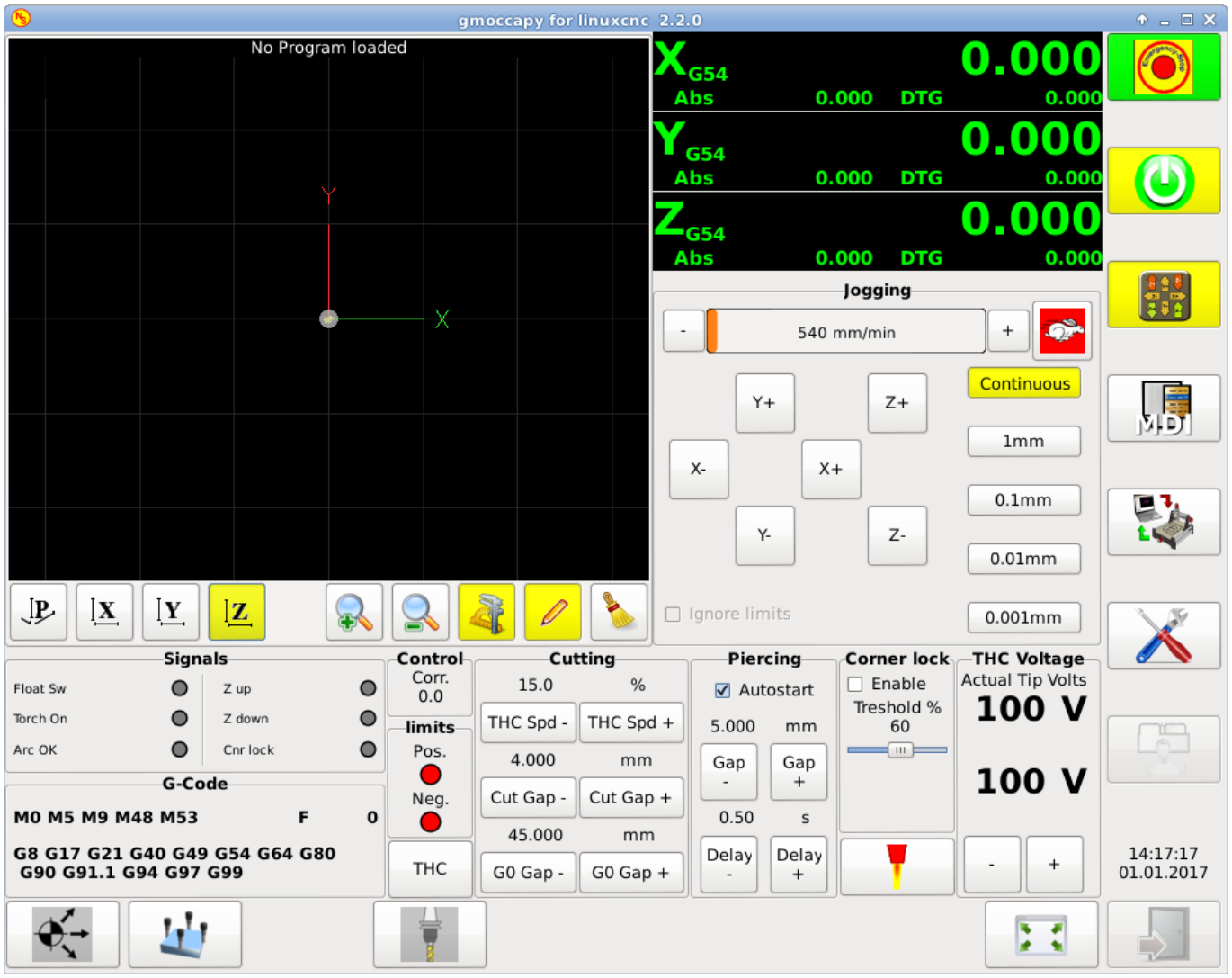


Figura 4.20: GUI plasma

Hay un muy buen WIKI, que en realidad está creciendo, mantenido por Marius. vea página de la wiki

4.2.10. Video en You Tube

Estos videos muestran gmoccapy en acción. Desafortunadamente los videos no muestra la última versión de gmoccapy, pero la forma de usarlo no cambiará mucho en el futuro. Intentaré actualizar los videos lo antes posible.

4.2.10.1. Uso básico

https://www.youtube.com/watch?v=O5B-s3uiI6g

4.2.10.2. Volantes Jog simulados

http://youtu.be/ag34SGxt97o

4.2.10.3. Página de configuración

https://www.youtube.com/watch?v=AuwhSHRJoiI

4.2.10.4. Botón de hardware simulado

Aleman = http://www.youtube.com/watch?v=DTqhY-MfzDE

Inglés = http://www.youtube.com/watch?v=ItVWJBK9WFA

4.2.10.5. Pestañas de usuario

http://www.youtube.com/watch?v=rG1zmeqXyZI

4.2.10.6. Videos de Medicion de Herramientas

Simulacion de medicion Auto = http://youtu.be/rrkMw6rUFdk

Pantalla de medición automática de herramienta = http://youtu.be/Z2ULDj9dzvk

Máquina de medición automática de herramientas = http://youtu.be/1arucCaDdX4

4.2.11. Problemas conocidos

4.2.11.1. Números extraños en el área de información

Si obtienes números extraños en el área de información de gmoccapy como:

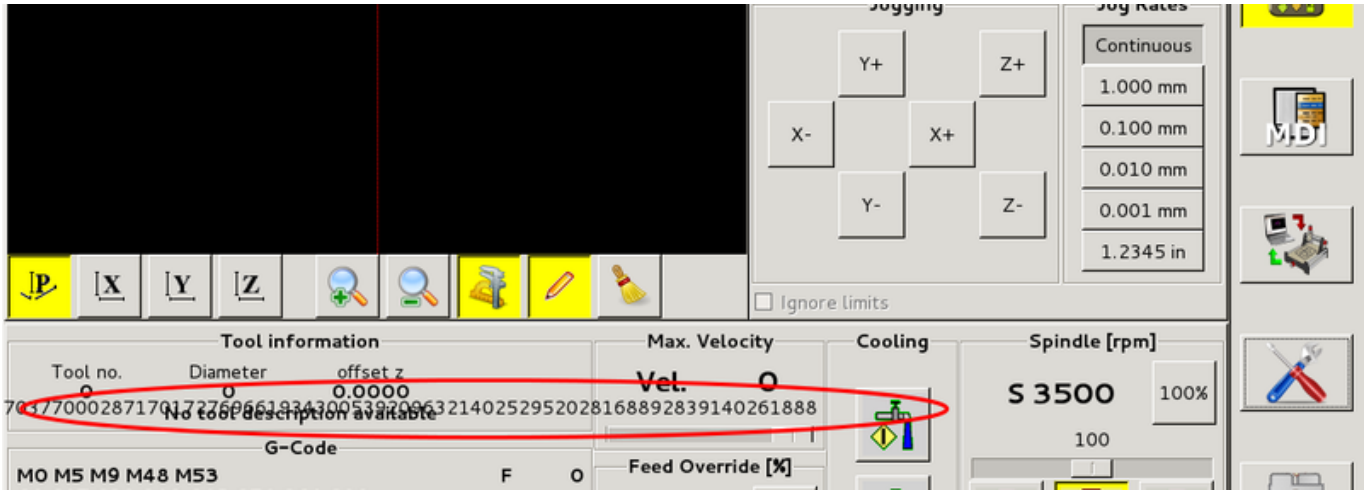


Figura 4.21: Numeros extraños

ha creado su archivo de configuración con una versión anterior de StepConfWizard. Este ha hecho una entrada incorrecta en el archivo INI bajo [TRAJ]MAX_LINEAR_VELOCITY = xxx. Cambia esa entrada a MAX_VELOCITY = xxx

4.2.11.2. Macro no finalizada

Si usas una macro sin movimiento, como esta:

```
 o<zeroxy> sub

G92.1
G92.2
G40

G10 L20 P0 X0 Y0

o<zeroxy> endsub
m2
```

gmoccapy no verá el final de la macro, porque el intérprete necesita cambie su estado a IDLE, pero la macro ni siquiera configura el intérprete a un nuevo estado. Para evitar eso, simplemente agregue una línea G4 P0.1 para obtener la señal necesaria. La macro correcta sería:

```
 o<zeroxy> sub

G92.1
G92.2
G40

G10 L20 P0 X0 Y0

G4 P0.1

o<zeroxy> endsub
m2
```

4.3. NGCGUI

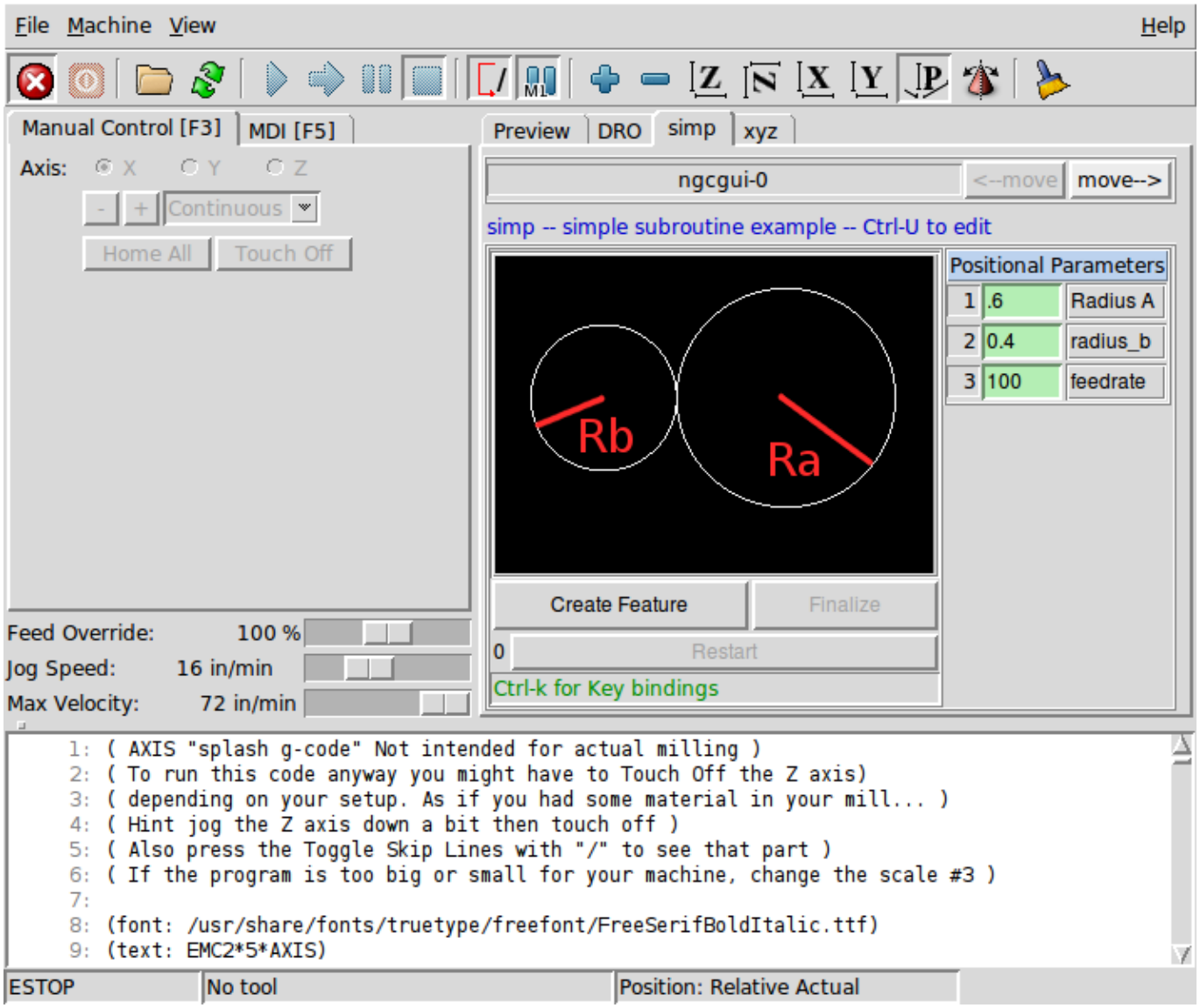


Figura 4.22: NGCGUI incrustado en Axis

4.3.1. Descripción general

- *NGCGUI* es una aplicación Tcl para trabajar con subrutinas. Permite tener una interfaz conversacional con LinuxCNC. Puede organizar las subrutinas en el orden en que las necesite para ejecutarlas y concatenarlas en un archivo para completar un programa.
- *NGCGUI* puede ejecutarse como una aplicación independiente o puede integrarse en múltiples páginas de pestañas en la GUI Axis
- *PYNGCGUI* es una implementación Python alternativa de ngcgui.
- *PYNGCGUI* puede ejecutarse como una aplicación independiente o puede integrarse como una página de pestañas (con su propio conjunto de múltiples pestañas de subrutinas) en cualquier GUI que admita incrustación de aplicaciones; gladevcp axis, touchy, gscreen y gmoccapy.

Usando NGCGUI o PYNGCGUI:

- Se proporcionan pestañas para cada subrutina especificada en el archivo INI
- Se pueden agregar nuevas pestañas de subrutinas sobre la marcha usando pestañas personalizadas
- Cada pestaña de subrutina proporciona cuadros de entrada para todos los parámetros de la subrutina
- Los cuadros de entrada pueden tener un valor predeterminado y una etiqueta que se identifican por comentarios especiales en el archivo de subrutina
- Las invocaciones de subrutina se pueden concatenar juntas para formar un programa de múltiples pasos
- Se puede usar cualquier subrutina de código G de un solo archivo que se ajuste a las convenciones ngcgui
- Se pueden usar cualquier programa gcmc (G code-meta-compiler) que se ajuste a las convenciones ngcgui para etiquetar variables (el ejecutable gcmc debe estar instalado por separado, ver: http://www.vagrearg.org/content/gcmc)

nota
NGCGUI y PYNGCGUI implementan las mismas funciones y ambas procesan archivos .ngc y .gcmc que se ajusten a algunas convenciones específicas de ngcgui. En este documento, el término *NGCGUI* generalmente se refiere a cualquier aplicación.

4.3.2. Configuraciones de demostración

Varias configuraciones de demostración se encuentran en el directorio sim de configuraciones de muestra que ofrece el selector de configuración LinuxCNC. El selector de configuración está en el menú principal del sistema:

```
CNC> LinuxCNC
```

Se incluyen ejemplos para Axis, touchy, gscreen y gmoccapy. Estos ejemplos muestran configuraciones cartesianas de 3 ejes (XYZ) (como fresadoras) y configuraciones de torno (XZ). Algunos ejemplos muestran el uso de un teclado emergente para sistemas de pantalla táctil y otros ejemplos muestran el uso de archivos creados por la aplicación gcmc. Los ejemplos de touchy también muestran la incorporación de un visor de backplot gladevcp (gremlin_view).

La aplicación más simple se encuentra como:

```
Configuraciones de muestra/sim/axis/ngcgui/ngcgui_simple
```

Un ejemplo completo que muestra la compatibilidad con gcmc está en:

```
Configuraciones de muestra/sim/axis/ngcgui/ngcgui_gcmc
```

Un ejemplo completo integrado como una aplicación gladevcp y usando gcmc está en:

```
Configuraciones de muestra/sim/gscreen/ngcgui/pyngcgui_gcmc
```

Las configuraciones sim de ejemplo utilizan archivos de biblioteca que proporcionan ejemplo de archivos de subrutina de código G (.ngc) y archivos de metacompilador de código G (.gcmc):

- *nc_files/ngcgui_lib*
  - *arc1.ngc* - arco básico con compensación del radio del cortador
  - *arc2.ngc* - arco dado por centro, offset, ancho, ángulo (llama a arc1)
  - *backlash.ngc* - rutina para medir el backlash de eje con un indicador
  - *db25.ngc*: crea un recorte de conector DB25
  - *gosper.ngc* - una demostración de recursión (Flownake)

- *helix.ngc* - corte de hélice o con agujero en D
- *helix_rtheta.ngc* - hélice o agujero en D colocado por radio y ángulo
- *hole_circle.ngc* - agujeros igualmente espaciados en un círculo
- *ihex.ngc* - hexágono interno
- *iquad.ngc* - cuadrilátero interno
- *ohex.ngc* - fuera del hexágono
- *oquad.ngc* - cuadrilátero exterior
- *qpex_mm.ngc* - demostración de qpockets (basado en mm)
- *qpex.ngc* - demostración de qpockets (basado en pulgadas)
- *qpocket.ngc* - bolsillo cuadrilátero
- *rectangle_probe.ngc* - sondea un área rectangular
- *simp.ngc* - un ejemplo de subrutina simple que crea dos círculos
- *slot.ngc* - ranura desde la conexión de dos puntos finales
- *xyz.ngc* - ejercitador de máquina limitado a una forma de caja

- *nc_files/ngcgui_lib/lathe*
  - *g76base.ngc* - interfaz gráfica de usuario para g76 threading
  - *g76diam.ngc* - subprocesos especificados por diámetros mayores y menores
  - *id.ngc* - perfora el diámetro interior
  - *od.ngc* - gira el diámetro exterior
  - *taper-od.ngc* - gira un cono en el diámetro exterior
- *nc_files/gcmc_lib*
  - *drill.gcmc*: perfora agujeros en el patrón rectangular
  - *square.gcmc*: demostración simple de etiquetas variables para archivos gcmc
  - *star.gcmc*: demostración de gcmc que ilustra funciones y matrices
  - *wheels.gcmc*: demostración de gcmc de patrones complejos

Para probar una demostración, seleccione una configuración sim y comience linuxCNC.

Si utiliza la interfaz gráfica de usuario Axis, presione *E-Stop* luego *Encender Máquina* luego *Home Todo*. Elija una pestaña ngcgui, complete cualquier espacio en blanco vacío con valores razonables y presione *Crear función* y luego *Finalizar*. Finalmente presione *Ejecutar* para ver cómo se ejecuta. Experimentar creando múltiples características y características de diferentes páginas de pestañas.

Para crear varias subrutinas concatenadas en un solo archivo, vaya a cada pestaña complete los espacios en blanco, presione *Crear función* y luego, con las teclas de flecha, mueva las pestañas necesarias para ponerlas en orden. Ahora presione *Finalizar* y responda el mensaje para crear.

Otras guis tendrán una funcionalidad similar pero los botones y nombres pueden ser diferentes.

Notas

Las configuraciones de demostración crean páginas con pestañas solo para algunos ejemplos. Cualquier interfaz gráfica de usuario con una pestaña personalizada puede abrir cualquiera de las subrutinas de bibliotecas de ejemplo o cualquier archivo de usuario si está en el path de la subrutina linuxCNC.

Para ver las combinaciones de teclas especiales, haga clic dentro de una página de pestaña ngcgui para obtener foco y luego presione Control-k.

Las subrutinas de demostración deben ejecutarse en configuraciones de máquina simuladas incluidas en la distribución. El usuario siempre debe comprender el comportamiento y el propósito de un programa antes de correrlos en una máquina real.

4.3.3. Ubicaciones de la biblioteca

En las instalaciones de linuxCNC desde paquetes deb, las configuraciones de simulación ngcgui usan enlaces simbólicos a bibliotecas LinuxCNC que no se pueden escribir por el usuario para:

- *nc_files/ngcgui_lib* subfiles compatibles con ngcgui
- *nc_files/ngcgui_lib/lathe* subfiles de torno compatibles con ngcgui
- *nc_files/gcmc_lib* programas compatibles con ngcgui-gcmc
- *nc_files/ngcgui_lib/utilitysubs* Subrutinas de ayuda
- *nc_files/ngcgui_lib/mfiles* Archivos M de usuario

Estas bibliotecas están ubicadas por elementos de archivo ini que especifican las rutas de búsqueda utilizadas por linuxCNC (y ngcgui):

```
[RS274NGC]
SUBROUTINE_PATH = ../../nc_files/ngcgui_lib:../../nc_files/gcmc_lib:../../nc_files/ ↩
    ngcgui_lib/utilitysubs
USER_M_PATH = ../../nc_files/ngcgui_lib/mfiles
```

nota
Estas son líneas largas (no continúan en varias líneas) que especifican los directorios utilizados en un path de búsqueda. Los nombres de directorio están separados por dos puntos (:). No hay espacio entre los nombres de directorio.

Un usuario puede crear nuevos directorios para sus propias subrutinas y M-files y agréguelos a la(s) ruta(s) de búsqueda.

Por ejemplo, un usuario podría crear directorios desde la terminal con los comandos:

```
mkdir /home/myusername/mysubs
mkdir /home/myusername/mymfiles
```

Y luego crear o copiar los archivos proporcionados por el sistema a estos directorios que el usuario puede escribir. Por ejemplo, un usuario puede crear un subfile compatible con ngcgui llamado:

```
/home/myusername/mysubs/example.ngc
```

Para usar archivos en nuevos directorios, el archivo ini debe editarse para incluir los nuevos subarchivos y para aumentar la(s) ruta(s) de búsqueda. Para este ejemplo:

```
[RS274NGC]
...
SUBROUTINE_PATH = /home/myusername/mysubs:../../nc_files/ngcgui_lib:../../nc_files/gcmc_lib ↩
    :../../nc_files/ngcgui_lib/utilitysubs
USER_M_PATH = /home/myusername/mymfiles:../../nc_files/ngcgui_lib/mfiles

[DISPLAY]
...
NGCGUI_SUBFILE = example.ngc
...
```

LinuxCNC (y ngcgui) usan el primer archivo encontrado al buscar en directorios en la ruta de búsqueda. Con este comportamiento, puede reemplazar un subfile ngcgui_lib colocando un subfile con un nombre idéntico en un directorio que se encuentre anteriormente en la ruta de busqueda. Se puede encontrar más información en el capítulo INI del Manual de integradores.

4.3.4. Uso independiente

4.3.4.1. NGCGUI independiente

Para su uso, escriba un terminal:

```
ngcgui --help
Uso:
  ngcgui --help | -?
  ngcgui [Opciones] -D nc_files_directory_name
  ngcgui [Opciones] -i LinuxCNC_inifile_name
  ngcgui [Opciones]

  Opciones:
       [-S subrutina_archivo]
       [-p preámbulo]
       [-P archivo_epílogo]
       [-o archivo_salida]
       [-a autosend_file]            (autosend al eje predeterminado: auto.ngc)
       [--noauto]                    (sin autosend al eje)
       [-N | --nom2]                 (sin terminador m2 (use%))
       [--font [big|small|fontspec]] (predeterminado: "Helvetica -10 normal")
       [--horiz | --vert]            (predeterminado: --horiz)
       [--cwidth comment_width]      (ancho del campo de comentario)
       [--vwidth varname_width]      (ancho del campo varname)
       [--quiet]                     (menos comentarios en outfile)
       [--noiframe]                  (predeterminado: el cuadro muestra la imagen)
```

nota
Como aplicación independiente, ngcgui maneja un único archivo de subrutina que se puede invocar varias veces. Se puede iniciar múltiples aplicaciones ngcgui independientes.

4.3.4.2. PYNGCGUI independiente

Para su uso, escriba en un terminal:

```
pyngcgui --help
Uso:
pyngcgui [Opciones] [sub_archivo]
Opciones que requieren valores:
    [-d | --demo] [0|1|2] (0: demo de nivel superior independiente)
                          (1: DEMO incrustar nuevo cuaderno)
                          (2: DEMO incrustado en el cuaderno existente)
    [-S | --subfile sub_filename]
    [-p | - preamble preamble_filename]
    [-P | --postamble postamble_filename]
    [-i | --ini nombre_infiltro]
    [-a | --autofile auto_filename]
    [-t | --test testno]
    [-K | --keyboardfile glade_file] (use el archivo glade popupkeyboard personalizado)
Opciones solas:
    [-v | --verbose]
    [-D | --debug]
    [-N | --nom2] (sin terminador m2 (use%))
    [-n | --noauto] (guardar pero no enviar resultados automáticamente)
    [-k | --keyboard] (use popupkeybaord predeterminado)
    [-s | --sendtoaxis] (envía el archivo ngc generado a la gui axis)
Notas:
```

```
Un conjunto de archivos se compone de un preámbulo, subarchivo y epílogo.
El preámbulo y el epílogo son opcionales.
Se puede especificar un conjunto de archivos desde cmdline.
Se pueden especificar múltiples conjuntos de archivos desde un inifile.
Si --ini NO se especifica:
   busque un linuxCNC en ejecución y use su inifile
```

nota
Como aplicación independiente, pyngcgui puede leer un archivo ini (o una aplicación linuxCNC ejecutandose) para crear páginas con pestañas para múltiples subarchivos.

4.3.5. Incrustar NGCGUI

4.3.5.1. Incrustar NGCGUI en Axis

Los siguientes elementos del archivo INI van en la sección [DISPLAY]. (Ver a continuación secciones adicionales para items adicionales necesarios)

- *TKPKG = Ngcgui 1.0* - el paquete NGCGUI
- *TKPKG = Ngcguittt 1.0*: el paquete True Type Tracer para generar texto para grabar (opcional, debe seguir TKPKG = Ngcgui).
- *TTT = truetype-tracer*: nombre del programa truetype tracer (debe estar en la RUTA del usuario)
- *TTT_PREAMBLE = in_std.ngc* - Opcional, especifica el nombre del archivo para el preámbulo utilizado para subfiles ttt creados. (alternativo: mm_std.ngc)

nota
Los elementos opcionales del trazador truetype se utilizan para especificar una página de pestaña compatible con ngcgui que usa la aplicación truetype-tracer. La aplicación truetype-tracer debe ser instalada de forma independiente y ubicada en la RUTA del usuario.

4.3.5.2. Incrustar PYNGCGUI como una pestaña gladevcp en una GUI

Los siguientes elementos del archivo INI van en la sección [DISPLAY] para usar con Axis, gscreen o touchy. (Consulte las secciones adicionales a continuación para obtener información adicional)

Items EMBED_

```
EMBED_TAB_NAME = Pyngcgui - nombre que aparecerá en la pestaña incrustada
EMBED_TAB_COMMAND = gladevcp -x {XID} pyngcgui_axis.ui - invoca gladevcp
EMBED_TAB_LOCATION = nombre_de_ubicación - donde se encuentra la página incrustada
```

nota
El especificador EMBED_TAB_LOCATION no se usa para la GUI Axis. Mientras Pyngcgui se puede incrustar en Axis, la integración es más completa cuando se usa ngcgui (usando TKPKG = Ngcgui 1.0). Para especificar EMBED_TAB_LOCATION para otras guis, vea Sección DISPLAY del Capítulo de configuración INI .

nota
La gui front-end truetype tracer no está disponible actualmente para aplicaciones gladevcp.

4.3.5.3. Items adicionales del archivo INI necesarios para ngcgui o pyngcgui

Los siguientes elementos del archivo INI van en la sección [DISPLAY] para cualquier gui que incorpore ngcgui o pyngcgui.

- *NGCGUI_FONT = Helvetica -12 normal* - especifica el nombre de la fuente, tamaño, normal|negrita
- *NGCGUI_PREAMBLE = in_std.ngc*: el archivo de preámbulo que se agregará delante del subrutinas. Al concatenar varias invocaciones de subrutinas comunes, este preámbulo solo se agrega una vez. Para máquinas basadas en mm, use mm_std.ngc
- *NGCGUI_SUBFILE = filename1.ngc* - crea una pestaña a partir de la subrutina filename1
- *NGCGUI_SUBFILE = filename2.ngc* - crea una pestaña a partir de la subrutina filename2
- *... etc.*
- *NGCGUI_SUBFILE = gcmcname1.gcmc* - crea una pestaña desde el archivo gcmcname1
- *NGCGUI_SUBFILE = gcmcname2.gcmc* - crea una pestaña desde el archivo gcmcname2
- *... etc.*
- *NGCGUI_SUBFILE = ""* - crea una pestaña personalizada que puede abrir cualquier subrutina en la ruta de búsqueda
- *NGCGUI_OPTIONS = opt1 opt2 ...* - Opciones de NGCGUI
  - *nonew* - no permitir hacer una nueva pestaña personalizada
  - *noremove* - no permitir eliminar ninguna pestaña
  - *noauto* - sin envío automático (use makeFile, luego guarde o envíe manualmente)
  - *noiframe* - sin imagen interna, muestra imágenes en un widget de nivel superior separado
  - *nom2* - Esta opción elimina todos los efectos secundarios de la terminación m2
- *GCMC_INCLUDE_PATH = dirname1:dirname2* - busca directorios para archivos include de gcmc Este es un ejemplo de NGCGUI incrustado en Axis. Las subrutinas deben estar en un directorio especificado por [RS274NGC]SUBROUTINE_PATH. En algun ejemplo las subrutinas usan otras subrutinas, así que asegúrese de tener las dependencias, si las hay, en un directorio SUBROUTINE_PATH. Algunas subrutinas pueden usar archivos M personalizados que deben estar en un directorio especificado por [RS274NGC]USER_M_PATH.

Gcode-meta-compiler (gcmc) puede incluir declaraciones como: include ("filename.inc.gcmc"); Por defecto, gcmc incluye el directorio actual que, para linuxCNC, será el directorio que contiene el archivo ini linuxCNC. Directorios adicionales pueden ser antepuestos al orden de búsqueda de gcmc con el elemento GCMC_INCLUDE_PATH.

Muestra de INI basado en GUI Axis.

```
[RS274NGC]
...
SUBROUTINE_PATH   = ../../nc_files/ngcgui_lib:../../ngcgui_lib/utilitysubs
USER_M_PATH       = ../../nc_files/ngcgui_lib/mfiles

[DISPLAY]
TKPKG             = Ngcgui    1.0
TKPKG             = Ngcguittt 1.0
# Ngcgui debe preceder a Ngcguittt
NGCGUI_FONT       = Helvetica -12 normal
# especificar solo nombres de archivo, los archivos deben estar en [RS274NGC] ↩
    SUBROUTINE_PATH
NGCGUI_PREAMBLE   = in_std.ngc
NGCGUI_SUBFILE    = simp.ngc
NGCGUI_SUBFILE    = xyz.ngc
NGCGUI_SUBFILE    = iquad.ngc
NGCGUI_SUBFILE    = db25.ngc
NGCGUI_SUBFILE    = ihex.ngc
```

```
NGCGUI_SUBFILE    = gosper.ngc
# especificar "" para una pestaña personalizada
NGCGUI_SUBFILE    = ""
#NGCGUI_SUBFILE   = "" se usa cuando se especifica el marco de imagen si
# se requiere abrir otros archivos
# las imágenes se colocarán en una ventana de nivel superior
NGCGUI_OPTIONS    =

#NGCGUI_OPTIONS = opt1 opt2 ...
# opt items:
#   nonew      -- no permitir hacer una nueva pestaña personalizada
#   noremove   -- no permite eliminar ninguna pestaña
#   noauto     -- sin envío automático (makeFile, luego enviar manualmente)
#   noiframe   -- sin imagen interna, imagen en el nivel superior separado
GCMC_INCLUDE_PATH = /home/myname/gcmc_includes

TTT               = truetype-tracer
TTT_PREAMBLE      = in_std.ngc

PROGRAM_PREFIX    = ../../nc_files
```

nota
Lo anterior no es un INI Axis completo - los elementos que se muestran son aquellos utilizado por ngcgui. LinuxCNC requiere muchos elementos adicionales para tener un archivo INI completo.

4.3.5.4. Truetype Tracer

Ngcgui_ttt proporciona soporte para truetype-tracer (v4). Crea una pestaña Axis que permite al usuario crear una nueva página de pestaña ngcgui después de ingresar texto y seleccionar una fuente y otros parámetros. (Truetype-tracer debe estar instalado independientemente).

Para incrustar ngcgui_ttt en Axis, especifique los siguientes elementos además de los elementos ngcgui:

```
Item:    [DISPLAY]TKPKG = Ngcgui_ttt version_number
Example: [DISPLAY]TKPKG = Ngcgui_ttt 1.0
Note:    Obligatorio, especifica la carga de ngcgui_ttt en una página de pestaña  ↩
   de eje llamada ttt.
         Debe seguir el elemento TKPKG = Ngcgui.
Item:    [DISPLAY]TTT = path_to_truetype-tracer
Example: [DISPLAY]TTT = truetype-tracer
Note:    Opcional, si no se especifica, intente usar /usr/local/bin/ truetype- ↩
   tracer.
         Especifique con ruta absoluta o como un nombre ejecutable simple en cuyo  ↩
            caso el entorno PATH del usuario se usará para encontrar el programa.
Item:    [DISPLAY]TTT_PREAMBLE = preamble_filename
Example: [DISPLAY]TTT_PREAMBLE = in_std.ngc
Note:    Opcional, especifica el nombre de archivo para el preámbulo utilizado  ↩
   para los archivos subttt creados.
```

4.3.5.5. Especificaciones de ruta en archivo INI

Ngcgui usa la ruta de búsqueda de linuxCNC.

La ruta de búsqueda comienza con el directorio estándar especificado por:

```
[DISPLAY]PROGRAM_PREFIX=nombre_directorio
```

seguido de múltiples directorios especificados por: [RS274NGC]SUBROUTINE_PATH=directorio1_nombre:directorio1_nombre:direct
...

Los directorios pueden especificarse como rutas absolutas o relativas.

```
Ejemplo: [DISPLAY]PROGRAM_PREFIX = /home/myname/linuxcnc/nc_files
Ejemplo: [DISPLAY]PROGRAM_PREFIX = ~/linuxcnc/nc_files
Ejemplo: [DISPLAY]PROGRAM_PREFIX = ../../nc_files
```

Una ruta absoluta que comienza con "/" especifica una ubicación completa en el sistema de archivos. Una ruta que comienza con u "~/" especifica una ruta que comienza desde el directorio de usuario. Una ruta que comienza con "~nombre de usuario/" especifica una ruta que comienza en el directorio de nombre de usuario.

Caminos relativos Las rutas relativas se basan en el directorio de inicio, que es el directorio que contiene el archivo INI. El uso de rutas relativas puede facilitar la reubicación de configuraciones, pero requiere una buena comprensión de los especificadores de ruta de Linux.

```
./d0 es lo mismo que d0, esto es, un directorio llamado d0 en el directorio de ↩
   inicio
../d1 se refiere a un directorio d1 en el directorio padre
../../d2 se refiere a un directorio d2 en el padre del directorio padre
../../../d3 etc.
```

Se pueden especificar varios directorios con [RS274NGC]SUBROUTINE_PATH separándolos con dos puntos. El siguiente ejemplo ilustra el formato para múltiples directorios y muestra el uso de rutas relativas y absolutas.

Ejemplo de directorios múltiples:

```
[RS274NGC]SUBROUTINE_PATH = ../../nc_files/ngcgui_lib:../../nc_files/ngcgui_lib/utilitysubs ↩
   :/tmp/tmpngc`
```

Esta es una línea larga, no continúa en varias líneas. Cuando linuxCNC y/o ngcgui busca archivos, se utiliza el primer archivo encontrado en la búsqueda.

LinuxCNC (y ngcgui) debe poder encontrar todas las subrutinas, incluidas las rutinas auxiliares que se llaman desde subfiles ngcgui. Es conveniente colocar las utilidades subs en un directorio separado como se indica en el ejemplo anterior.

La distribución incluye el directorio ngcgui_lib y los archivos de demostración para preámbulos, subarchivos, epíñogos y archivos auxiliares. Para modificar el comportamiento de los archivos, puede copiar cualquier archivo y colocarlo en una parte anterior de la ruta de búsqueda El primer directorio buscado es [DISPLAY]PROGRAM_PREFIX. Puede usar este directorio pero es mejor crear uno dedicado y colócarlo al comienzo de [RS274NGC]SUBROUTINE_PATH.

En el siguiente ejemplo, los archivos en /home/myname/linuxcnc/mysubs se encontrarán antes que archivos en ../../nc_files/ngcgui_lib.

Ejemplo de adición de directorio de usuarios:

```
[RS274NGC]SUBROUTINE_PATH=/home/myname/linuxcnc/mysubs:../../nc_files/ngcgui_lib:../../ ↩
   nc_files/ngcgui_lib/utilitysubs`
```

Los nuevos usuarios pueden intentar inadvertidamente usar archivos que no están estructurados para ser compatibles con los requisitos de ngcgui. Ngcgui probablemente informará numerosos errores si los archivos no están codificados con las convenciones. Las buenas prácticas sugieren que los subarchivos compatibles con ngcgui deben colocarse en un directorio dedicado a ese propósito y que los archivos de preámbulo, epilogo y auxiliar deben estar en directorio(s separados) para desalentar los intentos de usarlos como subarchivos. Archivos no destinados para usar como subarchivos puede incluir un comentario especial: "(not_a_subfile)" para que ngcgui los rechace automáticamente con un mensaje relevante.

4.3.5.6. Resumen de los detalles de elementos del archivo INI para usar NGCGUI

```
Elemento: [RS274NGC]SUBROUTINE_PATH = dirname1:dirname2:dirname3 ...
Ejemplo:  [RS274NGC]SUBROUTINE_PATH = ../../nc_files/ngcgui_lib:../../nc_files/ ↩
   ngcgui_lib/utilitysubs
```

```
Nota:     opcional, pero muy útil para organizar subarchivos y archivos de  ↩
   utilidad

Elemento: [RS274NGC]USER_M_PATH = dirname1:dirname2:dirname3 ...
Ejemplo:  [RS274NGC]USER_M_PATH = ../../nc_files/ngcgui_lib/mfiles
Nota:     Opcional, necesario para localizar archivos de usuario personalizados


Elemento: [DISPLAY]EMBED_TAB_NAME = nombre para mostrar en la página de pestaña  ↩
   incrustada
Ejemplo:  [DISPLAY]EMBED_TAB_NAME = Pyngcgui
Nota:     Las entradas: EMBED_TAB_NAME, EMBED_TAB_COMMAND, EMBED_TAB_LOCATION
          definen una aplicación integrada para varias guis linuxCNC

Elemento: [DISPLAY]EMBED_TAB_COMMAND = nombre del programa seguido de argumentos
Ejemplo:  [DISPLAY]EMBED_TAB_COMMAND = gladevcp -x {XID} pyngcgui_axis.ui
Nota:     Para aplicaciones gladevcp, vea el <<cha:glade-vcp,Capítulo GladeVCP>>

Elemento: [DISPLAY]EMBED_TAB_LOCATION = nombre_de_ubicación
Ejemplo:  [DISPLAY]EMBED_TAB_LOCATION = notebook_main
Nota:     Vea archivos INI de ejemplo para posibles ubicaciones.
          No requerido para Axis


Elemento: [DISPLAY]PROGRAM_PREFIX = dirname
Ejemplo:  [DISPLAY]PROGRAM_PREFIX = ../../nc_files
Nota:     Obligatorio y necesario para numerosas funciones de linuxCNC
          Es el primer directorio utilizado en la búsqueda de archivos.


elemento: [DISPLAY]TKPKG = Ngcgui número_versión
Ejemplo:  [DISPLAY]TKPKG = Ngcgui 1.0
Nota:     Solo se requiere para la incrustación en Axis, especifica la carga de
          las pestañas de Axis


Elemento: [DISPLAY]NGCGUI_FONT = font_descriptor
Ejemplo:  [DISPLAY]NGCGUI_FONT = Helvetica -12 normal
Nota:     Opcional, font_descriptor es un especificador de fuente compatible
          con tcl con elementos para fonttype -fontsize fontweight
          El valor predeterminado es: Helvetica -10 normal
          Los tamaños de fuente más pequeños pueden ser útiles para pantallas
          pequeñas. Los tamaños de fuente más grandes pueden ser útiles para
          aplicaciones de pantalla táctil

Elemento: [DISPLAY]NGCGUI_SUBFILE = subfile_filename
Ejemplo:  [DISPLAY]NGCGUI_SUBFILE = simp.ngc
Ejemplo:  [DISPLAY]NGCGUI_SUBFILE = square.gcmc
Ejemplo:  [DISPLAY]NGCGUI_SUBFILE = ""
Nota:     Use uno o más elementos para especificar compatible con ngcgui
          subarchivos o programas gcmc que requieren una página de pestañas
          al inicio.
          Se creará una pestaña "Personalizada" cuando el nombre de archivo sea  ↩
             "".
          Un usuario puede usar una pestaña "Personalizada" para explorar el
          sistema de archivos e identificar archivos de preámbulo, subarchivo y
          epílogo.
```

```
Elemento: [DISPLAY]NGCGUI_PREAMBLE = preamble_filename
Ejemplo:  [DISPLAY]NGCGUI_PREAMBLE = in_std.ngc
Nota:     Opcional, cuando se especifica, el archivo se antepone a un subarchivo.
          Los archivos creados con pestañas "Personalizadas" usan el preámbulo
          especificado con la pagina

Elemento: [DISPLAY]NGCGUI_POSTAMBLE = postamble_filename
Ejemplo:  [DISPLAY]NGCGUI_POSTAMBLE = bye.ngc
Nota:     Opcional, cuando se especifica, el archivo se agrega a un subarchivo.
          Los archivos creados con pestañas "Personalizadas" usan el postámbulo
          especificado con la pagina

Elemento: [DISPLAY]NGCGUI_OPTIONS = opt1 opt2 ...
Ejemplo:  [DISPLAY]NGCGUI_OPTIONS = nonew noremove
Nota:     las opciones múltiples están separadas por espacios en blanco.
          Por defecto, ngcgui configura páginas de pestañas para que:
            1) un usuario puede hacer nuevas pestañas
            2) un usuario puede eliminar pestañas (excepto la última restante)
            3) los archivos finalizados se envían automáticamente a linuxCNC
            4) un marco de imagen (iframe) está disponible para mostrar
               una imagen para el subarchivo (si se proporciona una imagen)
            5) el archivo de resultados ngcgui enviado a linuxCNC finaliza con
               un m2 (e incurre en efectos secundarios de m2)

         Las opciones nonew, noremove, noauto, noiframe, nom2 respectivamente
         deshabilitan estos comportamientos predeterminados.

         Por defecto, si un archivo de imagen (.png, .gif, jpg, pgm)
         se encuentra en el mismo directorio que el subarchivo, La imagen
         se muestra en el iframe. Especificando la opción noiframe pone a
         disposición botones adicionales para seleccionar un preámbulo, subarchivo
         y epílogo y casillas de verificación adicionales. Selecciones de las
         casillas de verificación siempre están disponibles con teclas especiales:
           Ctrl-R Toggle "Retener valores en la lectura de subarchivo"
           Ctrl-E Alternar "Expandir subrutina"
           Ctrl-a Alternar "Envío automático"
           (Ctrl-k enumera todas las teclas y funciones)

         Si se especifica noiframe y se encuentra un archivo de imagen,
         la imagen se muestra en una ventana separada y todas las funciones están
         disponibles en la página de pestañas.

         Las NGCGUI_OPTIONS se aplican a todas las pestañas ngcgui, excepto que ↩
            las
         opciones nonew, noremove y noiframe no son aplicables para pestañas
         "Personalizadas". No use pestañas "Personalizadas" si quiere limitar la
         capacidad del usuario para seleccionar subarchivos o crear pestañas
         adicionales.

Elemento: [DISPLAY]GCMC_INCLUDE_PATH = dirname1: dirname2: ...
Ejemplo:  [DISPLAY]GCMC_INCLUDE_PATH =/home/myname/gcmc_includes:/home/myname/ ↩
   gcmc_includes2
Nota:     Opcional, cada directorio se incluirá cuando se invoque gcmc
          usando la opción: --incluir dirname
```

4.3.6. Requisitos de archivo para compatibilidad NGCGUI

4.3.6.1. Requisitos de subrutina de Gcode de un solo archivo (.ngc)

Un subfile compatible con NGCGUI contiene una única definición de subrutina. El nombre de la subrutina debe ser el mismo que el nombre del archivo (sin incluir el sufijo .ngc). LinuxCNC admite subrutinas con nombre o numeradas, pero solo las subrutinas con nombre son compatibles con NGCGUI. Para más información ver el Capítulo códigos O.

La primera línea sin comentarios debe ser una declaración sub. La última línea sin comentarios debe ser una declaración endsub.

examp.ngc:

```
(información: info_text_to_appear_at_top_of_tab_page)
; línea de comentario que comienza con punto y coma
( línea de comentario usando paréntesis)
o<examp> sub
  BODY_OF_SUBROUTINE
o<examp> endsub
; línea de comentario que comienza con punto y coma
( línea de comentario usando paréntesis)
```

El cuerpo de la subrutina debe comenzar con un conjunto de declaraciones que definen parámetros nombrados locales para cada parámetro posicional esperado para la llamada de subrutina Estas definiciones deben ser consecutivas comenzando con #1 y terminando con el último número de parámetro utilizado. Se deben proporcionar definiciones para cada uno de estos parámetros (sin omisiones).

Numeración de parámetros

```
#<xparm> = #1
#<yparm> = #2
#<zparm> = #3
```

LinuxCNC considera que todos los parámetros numerados en el rango #1 al #30 son parámetros de llamada por lo que ngcgui proporciona cuadros de entrada para cualquier ocurrencia de parámetros en este rango. Es una buena práctica evitar el uso de parámetros numerados #1 a #30 en cualquier otro lugar de la subrutina. El uso de parámetros locales con nombre esta recomendado para todas las variables internas.

Cada declaración de definición puede incluir opcionalmente un comentario especial y un valor predeterminado para el parámetro.

Prototipo de declaración

```
#<vname> = #n (=default_value)
o
#<vname> = #n (comentario_texto)
o
#<vname> = #n (=default_value comment_text)
```

Ejemplos de parámetros

```
#<xparm> = #1 (=0.0)
#<yparm> = #2 (Ystart)
#<zparm> = #3 (=configuración de inicio de Z 0.0)
```

Si se proporciona un valor predeterminado, se ingresará en el cuadro de entrada para el parámetro al inicio.

Si se incluye comment_text, se usará para identificar la entrada en lugar del nombre del parámetro.

Parámetros con nombre global Notas sobre parámetros globales con nombre y ngcgui:

(los parámetros globales con nombre tienen un guión bajo en el nombre, como #<_someglobalname>)

Como en muchos lenguajes de programación, el uso de globales es poderoso pero a menudo puede conducir a consecuencias inesperadas. En LinuxCNC, los parámetros globales con nombre existentes serán válidos en la ejecución de subrutinas y las subrutinas pueden modificar o crear parámetros cun nombre globales.

Se desaconseja pasar información a subrutinas utilizando parámetros con nombre globales dado que dicho uso requiere el establecimiento y mantenimiento de un bien definido contexto global que es difícil de mantener. Usando los parámetros numerados de #1 a #30 como entradas de subrutina debería ser suficiente para satisfacer una amplia gama de requerimientos de diseño.

Si bien no se recomiendan los parámetros con nombre global de entrada, las subrutinas linuxCNC deben usar parámetros globales con nombre para devolver resultados. Los subarchivos ngcgui-compatible están destinados al uso de GUI; los valores de retorno no son un requisito común. Sin embargo, ngcgui es útil como herramienta de prueba para subrutinas que devuelven parámetros con nombre global y es común que los subfiles compatibles con ngcgui llamen archivos de subrutina de utilidad que devuelven resultados con parámetros globales con nombre.

Para admitir estos usos, ngcgui ignora los parámetros globales con nombre que incluyen dos puntos (:) en su nombre. El uso de los dos puntos (:) en el nombre evita que ngcgui haga cuadros de entrada para estos parámetros.

Parámetros con nombre global

```
o<examp> sub
...
# <_ examp: result> = # 5410 (devuelve el diámetro actual de la herramienta)
...
o<helper> call [#<x1>] [#<x2>] (llamar a una subrutina)
#<xresult> = #<_helper:answer> (localice inmediatamente el resultado global del helper)
#<_helper:answer> = 0.0 (anula el parámetro global con nombre utilizado por la subrutina)
...
o<examp> endsub
```

En el ejemplo anterior, la subrutina de la utilidad se encontrará en un archivo separado llamado helper.ngc. La rutina auxiliar devuelve un resultado en un parámetro global con nombre llamado #<_helper:answer>.

Para una buena práctica, el subarchivo de llamada localiza inmediatamente el resultado para su uso en otro lugar del subarchivo y el parámetro global con nombre utilizado para devolver resultado se anula en un intento de mitigar su uso involuntario en otros lugares del contexto global. (Un valor de anulación de 0.0 puede no siempre ser buena elección).

Ngcgui admite la creación y concatenación de múltiples funciones para un subfile y para múltiples subfiles. A veces es útil para subarchivos determinar su orden en tiempo de ejecución para que ngcgui inserte un parámetro global especial que se puede testear dentro de las subrutinas. El parámetro se llama # <_feature:>. Su valor comienza con un valor 0 y se incrementa para cada característica agregada.

Características adicionales Se puede incluir un comentario especial de *información* en cualquier lugar de un subarchivo ngcgui-compatible. El formato es:

```
(info: info_text)
```

Info_text se muestra cerca de la parte superior de la página de la pestaña ngcgui en Axis.

Los archivos que no están destinados a usarse como subarchivos pueden incluir un comentario especial para que ngcgui los rechace automáticamente con un mensaje relevante.

```
(not_a_subfile)
```

Un archivo de imagen opcional (.png, .gif, .jpg, .pgm) puede acompañar a un subarchivo. Los archivos de imagen puede ayudar a aclarar los parámetros utilizados por el subarchivo. El archivo de imagen debe estar en el mismo directorio que el subarchivo y tener el mismo nombre con un sufijo de imagen apropiado, p.ej. el subfile example.ngc podría ir acompañado de un archivo de imagen example.png. Ngcgui intenta cambiar el tamaño de las imágenes grandes submuestreando a un tamaño con un ancho máximo de 320 x 240 píxeles.

Ninguna de las convenciones requeridas para hacer un subfile compatible con ngcgui excluye su uso como archivo de subrutina de propósito general para LinuxCNC.

La distribución LinuxCNC incluye una biblioteca (directorio ngcgui_lib) que incluye archivos de utilidad y subfiles compatibles con ngcgui de ejemplo para ilustrar las características de las subrutinas LinuxCNC y el uso de ngcgui. Otra biblioteca (gcmc_lib) proporciona ejemplos de archivos de subrutina para el metacompilador Gcode (gcmc)

Se pueden encontrar subrutinas adicionales para usuarios en el Foro, en la sección de subrutinas.

4.3.6.2. Requisitos del archivo Gcode-meta-compiler (.gcmc)

Ngcgui lee los archivos de Gcode-meta-compiler (gcmc) y crea cuadros de entrada para variables etiquetadas en el archivo. Cuando una característica del archivo finaliza, ngcgui pasa el archivo como entrada al compilador gcmc y, si la compilación es exitosa, el archivo gcode resultante se envía a linuxCNC para su ejecución. El archivo resultante está formateado como subrutina de un solo archivo; Los archivos .gcmc y los archivos .ngc se pueden mezclar en ngcgui.

Las variables identificadas para su inclusión en ngcgui están etiquetadas con líneas que parecerán comentarios al compilador gcmc.

Ejemplos de formatos de etiquetas variables.

```
//ngcgui: varname1 =
//ngcgui: varname2 = value2
//ngcgui: varname3 = value3, label3;
```

Ejemplos:

```
//ngcgui: zsafe =
//ngcgui: feedrate = 10
//ngcgui: xl = 0, x limit
```

Para estos ejemplos, el cuadro de entrada para varname1 no tendrá un valor predeterminado, el cuadro de entrada para varname2 tendrá un valor predeterminado de value2, y el el cuadro de entrada para varname 3 tendrá un valor predeterminado de 3 y una etiqueta label3 (en lugar de varname3). Los valores predeterminados deben ser números.

Para facilitar la modificación de líneas válidas en un archivo gcmc, alterne formatos de línea de etiqueta aceptados. Los formatos alternativos ignoran los punto y coma finales (;) y marcadores de comentarios finales (//). Con esta disposición, a menudo permite agregar la etiqueta //ngcgui: a líneas existentes en un archivo .gcmc.

Formatos de etiquetas variables alternativas

```
//ngcgui: varname2 = value2;
//ngcgui: varname3 = value3; //, etiqueta3;
```

Ejemplos:

```
//ngcgui: feedrate = 10;
//ngcgui: xl = 0; //, límite x
```

Una línea de información que aparecerá en la parte superior de una pestaña puede ser opcional incluido con una línea etiquetada como:

Etiqueta de información

```
//ngcgui: info: text_to_appear_at_top_of_tab_page
```

Cuando sea necesario, las opciones se pueden pasar al compilador gcmc con una línea etiquetada:

Formato de etiqueta de línea de opción

```
//ngcgui: -option_name [ [=] option_value]
```

Ejemplos:

```
//ngcgui: -I
//ngcgui: --imperial
//ngcgui: --precisión 5
//ngcgui: --precision = 6
```

Las opciones para gcmc están disponibles con el comando de terminal:

```
gcmc --help
```

Un programa gcmc por defecto usa el modo métrico. El modo puede establecerse en pulgadas con la opción:

```
//ngcgui: --imperial
```

Un archivo de preámbulo, si se usa, puede establecer un modo (g20 o g21) que entra en conflicto con el modo utilizado por un archivo gcmc. Para asegurar que el modo de programa gcmc está en vigor, incluya la siguiente declaración en el archivo .gcmc:

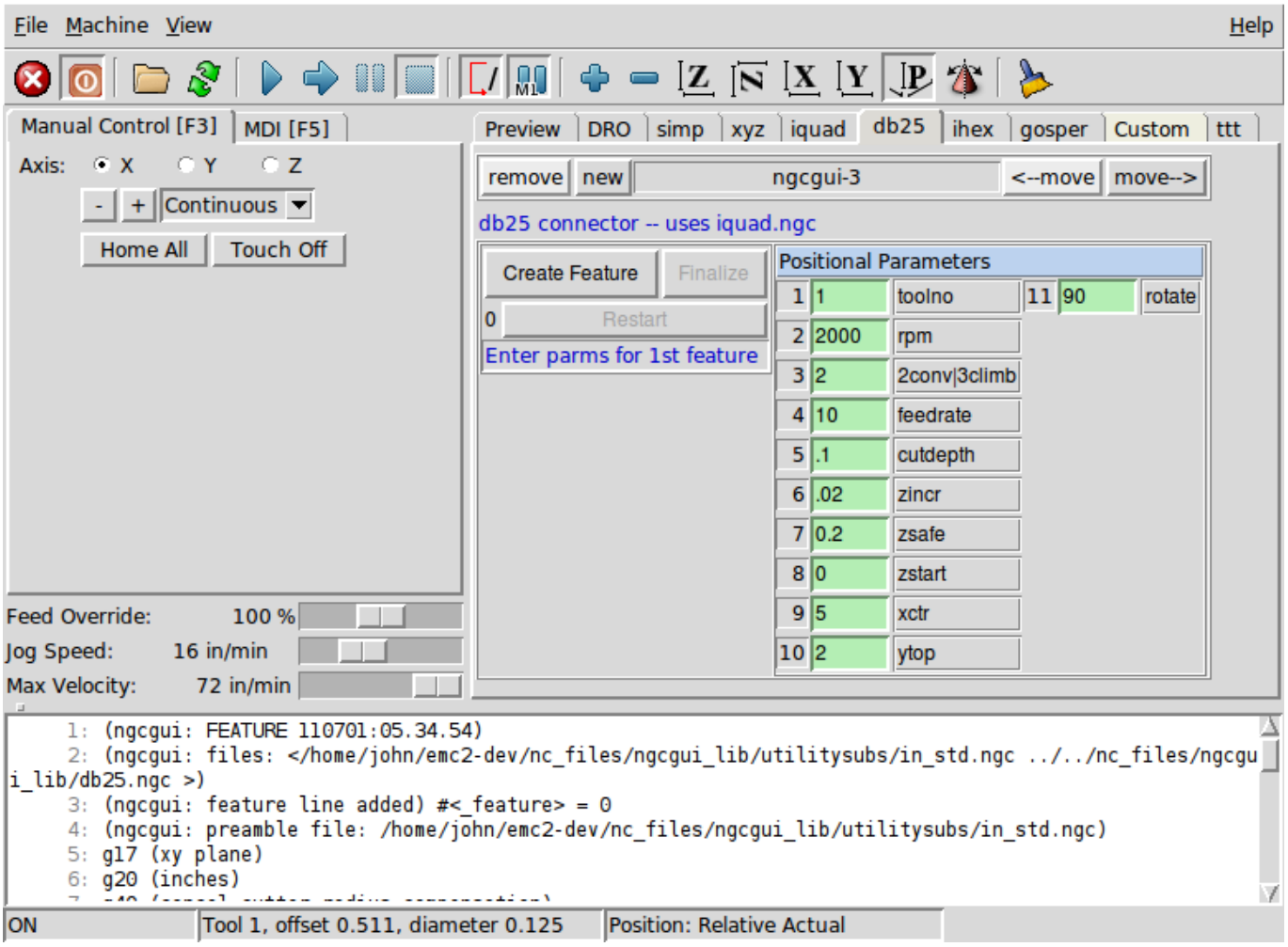
```
include ("sure_mode.gcmc")
```

y proporcione una ruta adecuada para gcmc include_files en el archivo ini, por ejemplo:

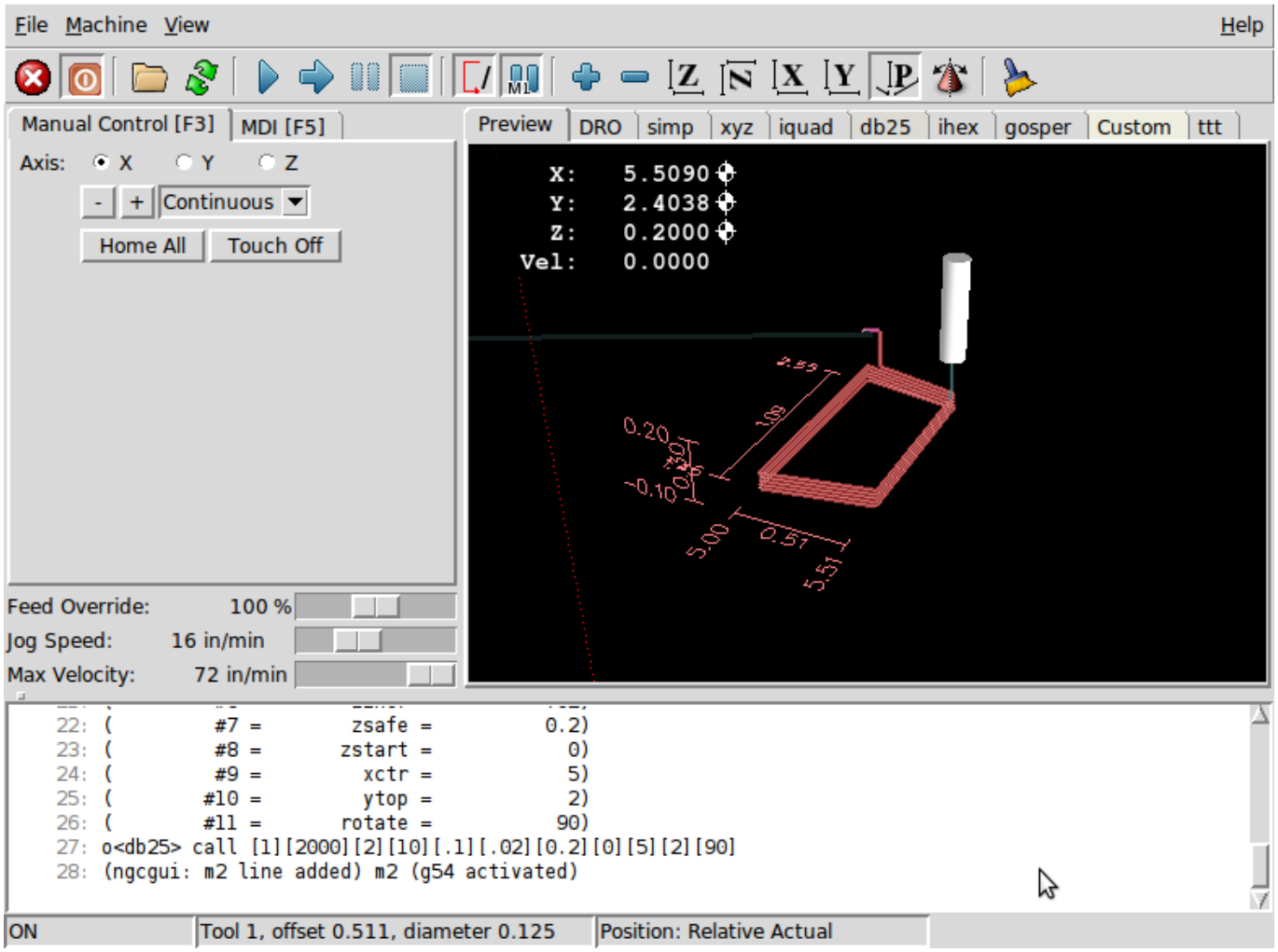
```
[DISPLAY]
GCMC_INCLUDE_PATH = ../../nc_files/gcmc_lib
```

4.3.7. DB25 Ejemplo

A continuación se muestra la subrutina DB25. En la primera imagen se muestra donde completar los espacios en blanco para cada variable.



Esta imagen muestra el plot de la subrutina DB25.



Esta imagen muestra el uso del nuevo botón y la pestaña personalizada para crear tres recortes DB25 en un programa.

images/ngcgui-db25-3.png

4.4. GUI táctil

Touchy es una interfaz de usuario para LinuxCNC diseñada para su uso en paneles de control de máquinas, y por lo tanto no requiere teclado ni mouse.

Está diseñado para usarse con una pantalla táctil y funciona en combinación con un volante/MPG y algunos botones e interruptores.

La pestaña *Volante* tiene botones radio para seleccionar entre las funciones de *Porcentaje de alimentación*, *Porcentaje de husillo*, *Velocidad máxima* y *Jogging* para la entrada del volante/MPG. También están previstos botones radio para la selección de eje e incrementos para jog

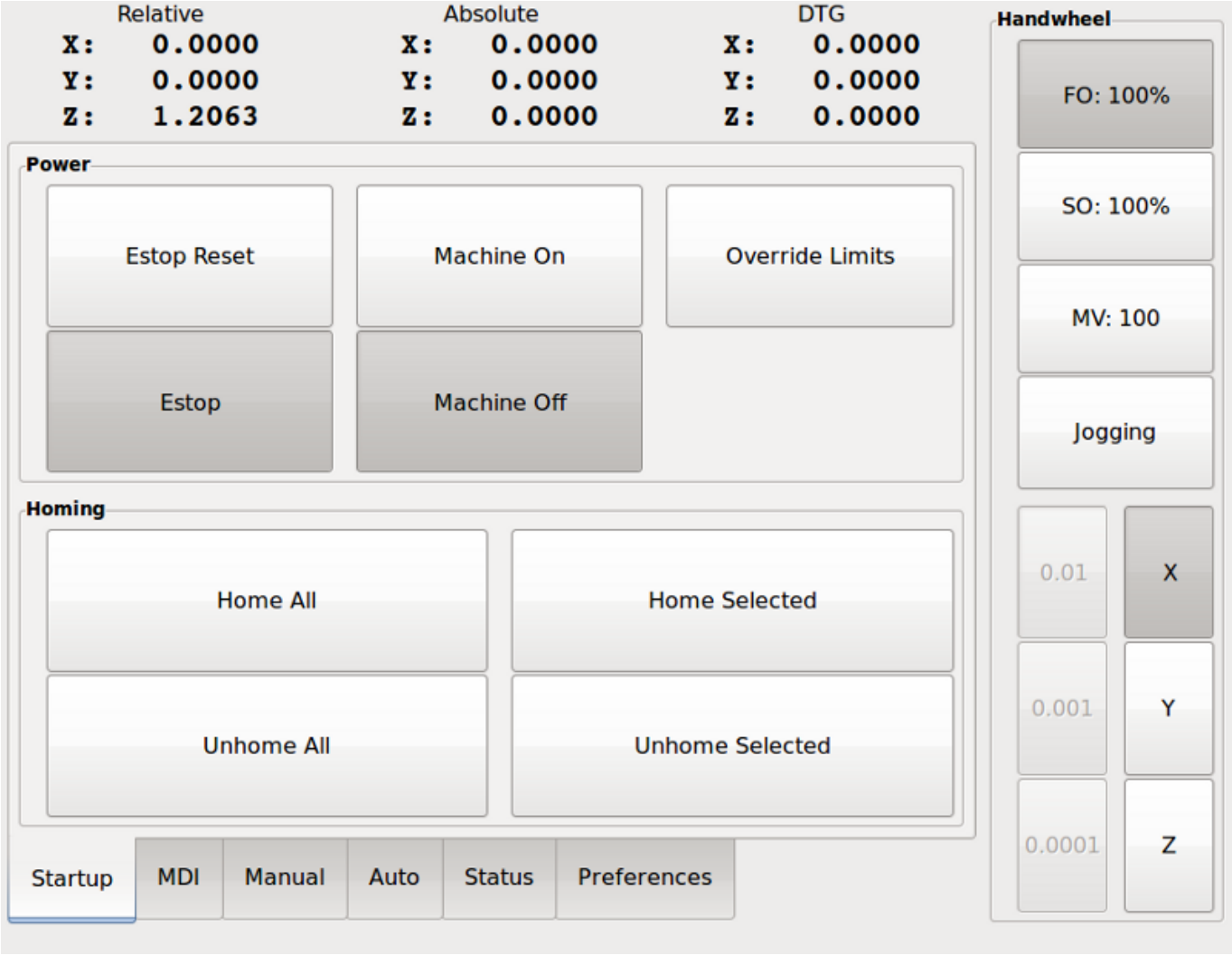


Figura 4.23: Touchy

4.4.1. Configuración del panel

4.4.1.1. Conexiones HAL

Touchy requiere que cree un archivo llamado *touchy.hal* en su directorio de configuración (el directorio en el que se encuentra su archivo ini) para conectar sus controles. Touchy ejecuta los comandos HAL en este archivo después de que haya hecho disponibles sus propios pines para la conexión.

Para obtener más información sobre los archivos HAL y el comando net, consulte el Referencia básica de HAL.

Touchy tiene varios pines de salida que deben conectarse al controlador de movimiento para controlar el jogging de volante:

- *touchy.jog.wheel.increment*, que debe conectarse al pin *axis.N.jog-scale* de cada eje N.
- *touchy.jog.wheel.N*, que se debe conectar a *axis.N.jog-enable* para cada eje N.

[NOTE] *N* representa el número de eje 0-8.

- Además de estar conectado a *touchy.wheel-count*, la cuenta del volante también debe estar conectada a *axis.N.jog-count* para cada eje N. Si usa el componente HAL *ilowpass* para suavizar el jog con el volante, asegúrese de suavizar solo *axis.N.jog-count* y no *touchy.wheel-count*.

CONTROLES REQUERIDOS.

- Botón de cancelación (contacto momentáneo) conectado al pin HAL *touchy.abort*
- Botón de inicio de ciclo (contacto momentáneo) conectado a *touchy.cycle-start*
- Rueda / MPG, conectada a *touchy.wheel-count* y pines de movimiento como se describe anteriormente
- Bloque único (interruptor de palanca) conectado a *touchy.single-block*

CONTROLES OPCIONALES.

- Para jog continuo, una palanca momentánea bidireccional centrada (o dos botones momentáneos) para cada eje, enganchados a *touchy.jog.continuous.x.negative*, *touchy.jog.continuous.x.positive*, etc.
- Si se desea un botón momentaneo para mover Z a la parte superior del recorrido a maxima velocidad , debe conectarse a *touchy.quill-up*.

LÁMPARAS DE PANEL OPCIONALES.

- *touchy.jog.active* muestra cuando los controles de desplazamiento del panel están activos.
- *touchy.status-indicator* está encendido cuando la máquina está ejecutando código G, y parpadea cuando la máquina está ejecutando pero está en pausa/feedhold.

4.4.1.2. Recomendado para cualquier configuración

- Botón de parada Estop conectado en la cadena de parada

4.4.2. Configuración

4.4.2.1. Habilitar Touchy

Para usar Touchy, en la sección *[DISPLAY]* de su archivo ini cambie la línea del selector a *DISPLAY = touchy*

4.4.2.2. Preferencias

Cuando inicie Touchy la primera vez, verifique la pestaña Preferencias. Si usa una pantalla táctil, para mejores resultados elija la opción de ocultar el puntero.

La ventana de estado tiene una altura fija, establecida por el tamaño de una fuente fija. Esto puede verse afectado por el DPI de Gnome, configurado en Sistema/Preferencias/Apariencia/Fuentes/Detalles. Si la parte inferior de la pantalla está cortada, reducir la configuración de DPI.

Todos los demás tamaños de fuente se pueden cambiar en la pestaña Preferencias.

4.4.2.3. Macros

Touchy puede invocar macros O-word usando la interfaz MDI. Para configurar esto, en la sección *[TOUCHY]* del archivo ini, agregue una o más líneas *MACRO*. Cada una debe tener el formato

MACRO = increment xinc yinc

En este ejemplo, increment es el nombre de la macro y acepta dos parámetros, llamados xinc e yinc.

Ahora, coloque la macro en un archivo llamado *increment.ngc*, en el Directorio *PROGRAM_PREFIX* o cualquier directorio en *SUBROUTINE_PATH*.

Debería verse así:

```
O<increment> sub
G91 G0 X#1 Y#2
G90
O<increment> endsub
```

Observe que el nombre del sub coincide con el nombre del archivo y el nombre de la macro exactamente, incluyendo el caso.

Cuando invoque la macro presionando el botón Macro en el MDI de Touchy, puede ingresar valores para xinc e yinc. Estos son pasados a la macro como *#1* y *#2* respectivamente. Los parámetros en blanco se pasan como valor 0.

Si hay varias macros diferentes, presione el botón Macro repetidamente para recorrerlos.

En este ejemplo simple, si ingresa -1 para xinc y presiona arranque de ciclo, se invocará un movimiento rápido *G0*, moviendo una unidad a la izquierda.

Esta capacidad macro es útil para el sondeo de bordes/agujeros y otras tareas de configuracion, así como quizás fresado de agujeros u otras operaciones simples que se puede hacer desde el panel sin necesidad de escribir especialmente programas de gcode.

4.5. Gscreen

4.5.1. Introducción

Gscreen es una infraestructura para mostrar una pantalla personalizada para controlar LinuxCNC. Gscreen se apoya en GladeVCP. GladeVCP usa el editor de widgets GTK GLADE para construir paneles de control virtual (VCP) mediante raton. Gscreen combina esto con la programación en Python para crear una pantalla GUI y manejar una máquina CNC.

Gscreen es personalizable, admitiendo diferentes botones y LED de estado. Gscreen soporta GladeVCP, que se usa para agregar controles e indicadores. Para personalizar Gscreen se usa el editor de Glade. Gscreen no se limita a añadir una personalizacion. El panel de la derecha o una pestaña personalizada es completamente editable.

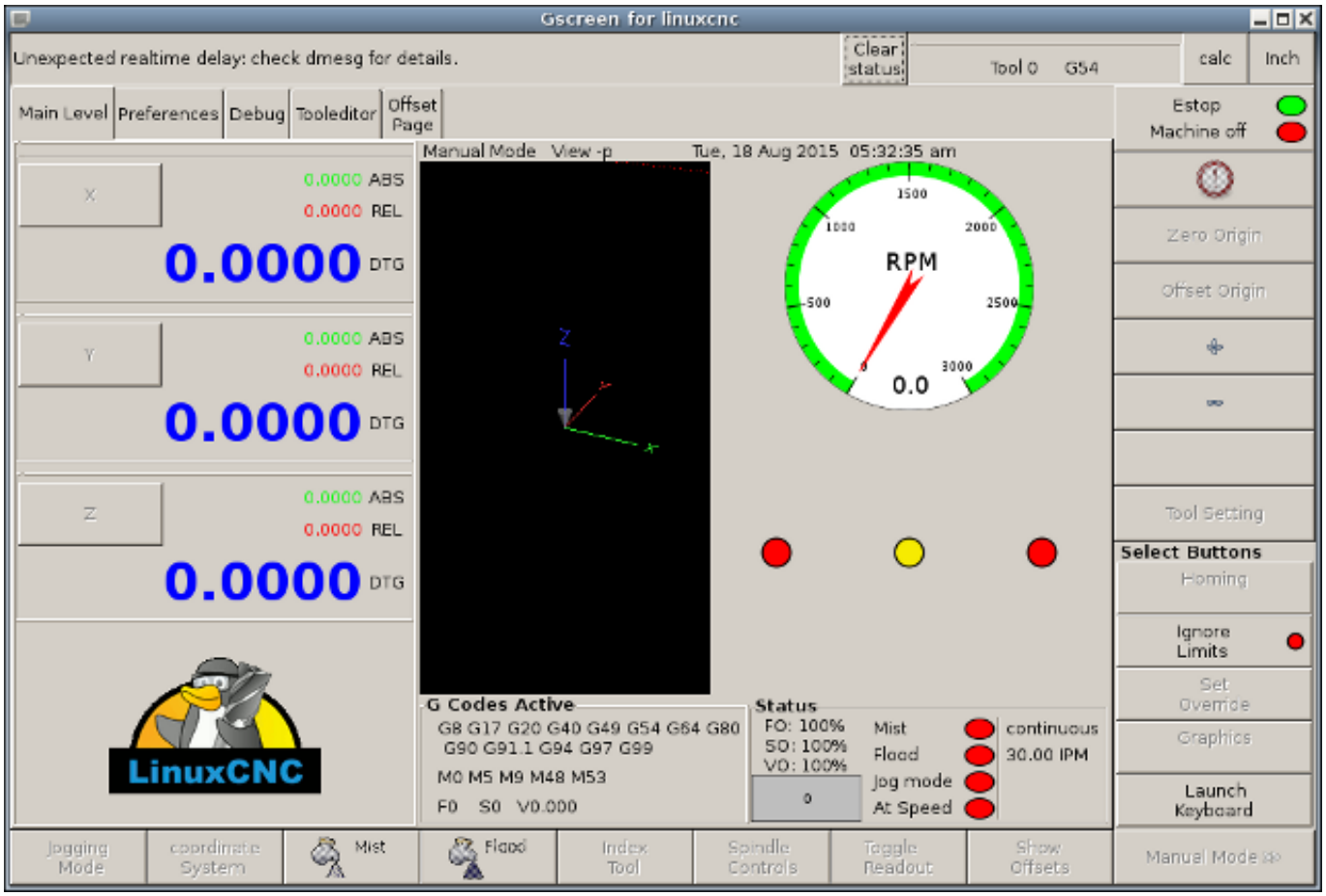


Figura 4.24: Pantalla por defecto Gscreen

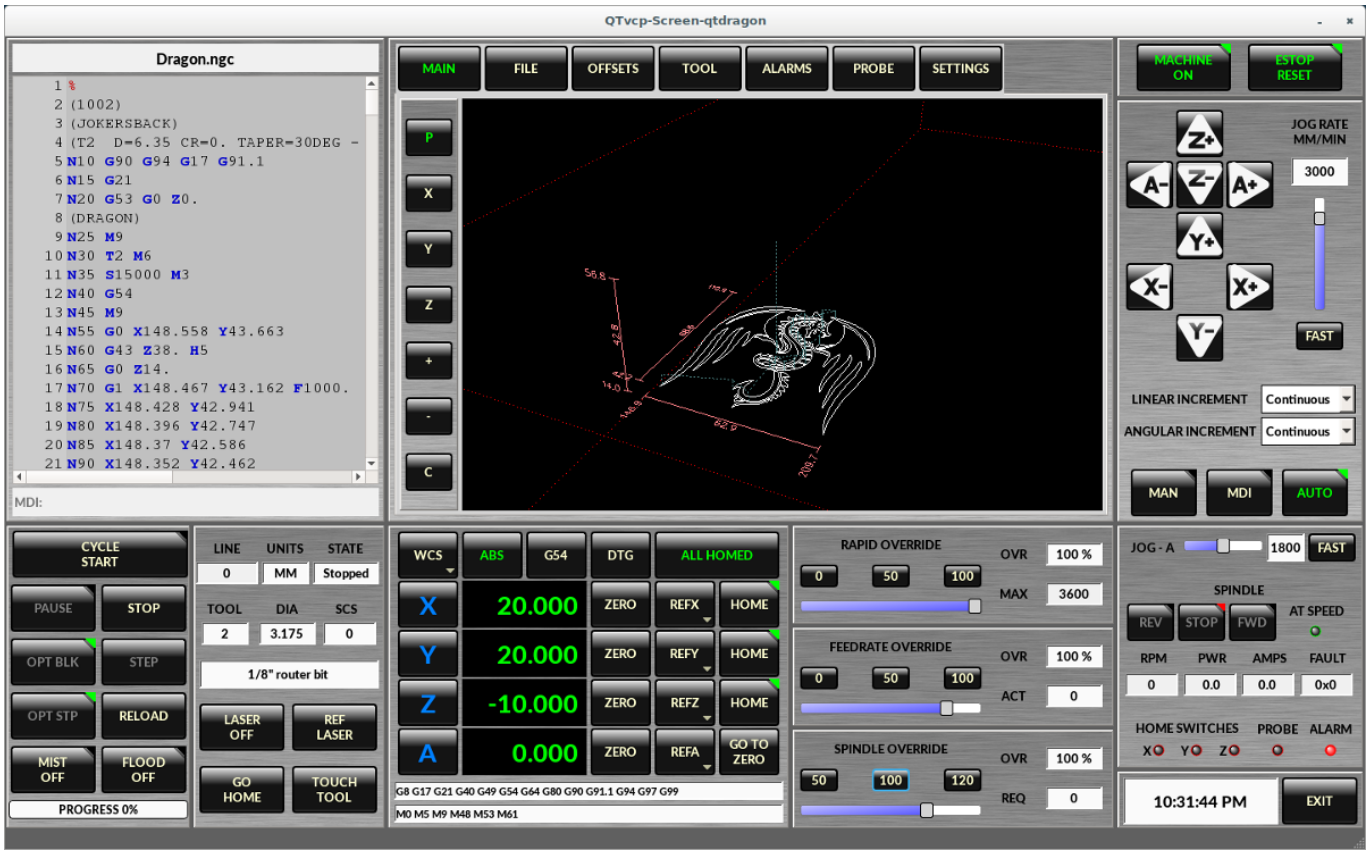


Figura 4.25: Silverdragon

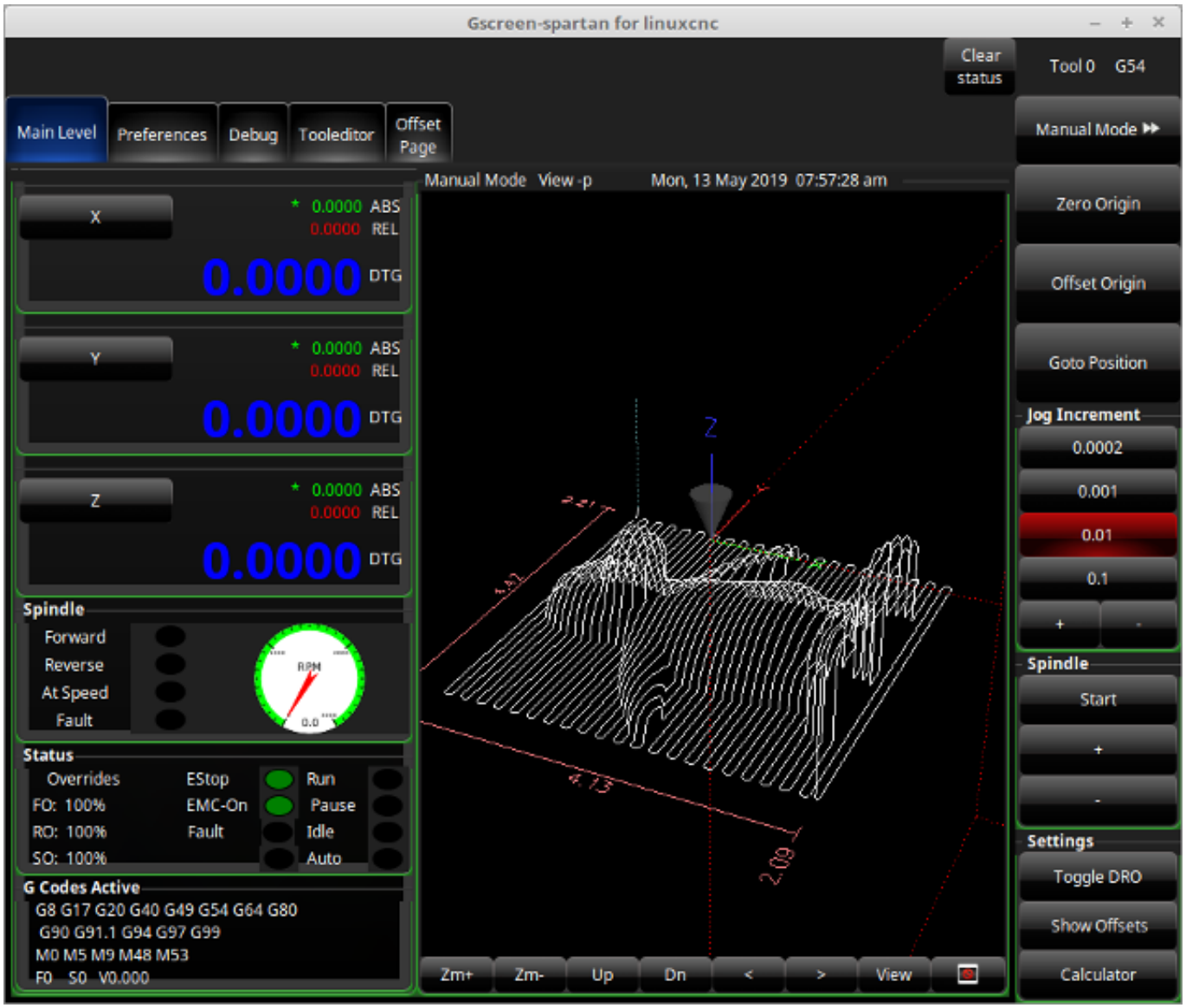


Figura 4.26: Spartan

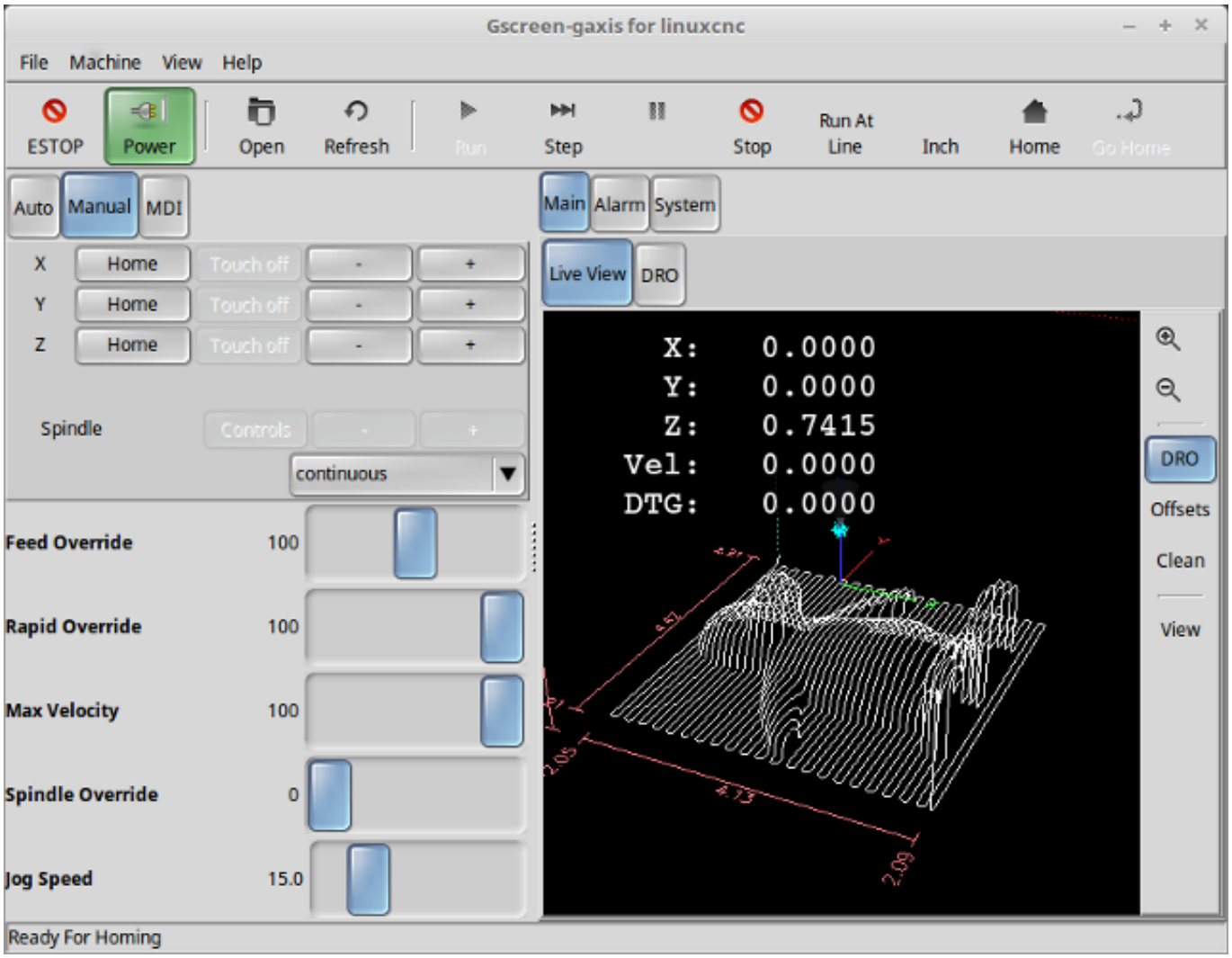


Figura 4.27: Gaxis

Figura 4.28: Industrial

Gscreen se basa en *Glade* (el editor), *PyGTK* (el kit de herramientas widgets), y *GladeVCP* (conexión de LinuxCNC a Glade y PyGTK). GladeVCP tiene algunos widgets especiales y acciones agregadas solo para LinuxCNC. Un widget es solo el nombre genérico utilizado para los botones, controles deslizantes, etiquetas, etc. del kit de herramientas PyGTK.

4.5.1.1. Archivo Glade

Un archivo Glade es un archivo de texto en XML estandar, que describe el diseño y los widgets de la pantalla. PyGTK utiliza este archivo para mostrar y reaccionar a esos widgets. El editor de Glade hace que sea relativamente fácil construir y editar este archivo. Debe usar el editor Glade 3.8.0 que usa widgets GTK2. Las versiones más nuevas de Linux usan el nuevo editor Glade que usa widgets GTK3. En ese caso debes descargar el editor Glade anterior de su repositorio.

4.5.1.2. PyGTK

PyGTK es el enlace de python a GTK. GTK es el *kit de herramientas* de widgets visuales, programado en C. PyGTK usa Python para *enlazar* con GTK.

4.5.2. GladeVCP

GladeVCP enlaza LinuxCNC, HAL, PyGTK y Glade, todos juntos. LinuxCNC requiere algunos widgets especiales; GladeVCP los suministra. Muchos son solo extensiones HAL a los widgets PyGTK existentes. GladeVCP crea los pines HAL para los

widgets especiales descritos en el archivo Glade. GladeVCP también permite agregar comandos Python para interactuar con los widgets, de forma que hagan cosas que no estan disponibles en su forma predeterminada. Si puede construir un panel GladeVCP, puede personalizar Gscreen!

4.5.2.1. Descripción general

Hay dos archivos que se pueden usar, individualmente o en combinación, para agregar personalizaciones; archivos de glade locales y archivos manejadores (handler). Normalmente, Gscreen utiliza el archivo Glade de serie y posiblemente un archivo handler (si se usa un *skin*). Puede especificar Gscreen para usar Glade y archivos handler *local*. Gscreen mira en la carpeta que contiene todos los archivos de configuración para la configuración que ha seleccionado.

Archivos Glade Locales Si están presentes, se cargarán los archivos glade locales en la carpeta de configuración en lugar de los archivos de Glade de serie. Los archivos locales de Glade le permiten usar sus diseños personalizados en lugar de las pantallas por defecto. Hay un interruptor en el archivo INI para establecer el nombre base: *-c nombre* para que Gscreen busque MYNAME.glade y MYNAME_handler.py.

Puedes decirle a Gscreen que solo cargue el archivo Glade y no conecte sus señales internas. Esto permite a gscreen cargar cualquier archivo Glade guardado por GTK. Esto significa que puede mostrar una pantalla completamente personalizada, pero también requiere que utilice un archivo manejador. Gscreen usa el archivo Glade para definir los widgets, por lo que puede mostralos e interactuar con ellos. Muchos de ellos tienen nombres específicos, otros tienen nombres genéricos Glade. Si el widget debe mostrarse pero nunca se cambiará, el nombre genérico estára bien. Si se necesita controlar o interactuar con el widget, entonces se da un nombre con un propósito determinado (todos los nombres deben ser únicos). Los widgets también pueden tener señales definidas para ellos en el editor de GLADE. Esto define que señal se da y qué método llamar.

Modificación de Skins de serie. Si cambia el nombre de un widget, Gscreen no podrá encontrarlo. Si se hace referencia a este widget desde el código Python, en el mejor de los casos el widget no funcionara y, en el peor de los casos, causará un error al cargar las pantallas predeterminadas de Gscreen que no utilizan muchas señales definidas en el editor, sino que las definen en el código python. Si mueve (corta y pega) un widget con señales, las señales no serán copiadas. Debe agregarlas de nuevo manualmente.

Archivos Handler Un archivo handler es un archivo que contiene código python que Gscreen agrega a su rutinas por defecto. Un archivo handler permite modificar valores predeterminados o agregar lógica a una skin Gscreen sin tener que modificar Gscreen propiamente dicho. Puede combinar nuevas funciones con funciones de Gscreen para modificar el comportamiento a su gusto. Usted puede omitir por completo todas las funciones de Gscreen y hacer que funcionen de forma completamente diferente. Si esta presente un archivo handler llamado gscreen_handler.py (o MYNAME_handler.py si usa el interruptor INI) se cargará y registrará. Gscreen solo permite buscar un archivo handler; si lo encuentra, tomara nota de los nombres de funciones específicas y las llamara en lugar de las predeterminadas. Si se agregan widgets, puede configurar llamadas de señal desde el editor de Glade para llamar a rutinas que haya escrito en el archivo handler. De esta forma podrás tener un comportamiento personalizado. Las rutinas del handler pueden llamar a las rutinas predeterminadas de Gscreen, ya sea antes o después de ejecutar su propio código. De esta manera se puede añadir comportamiento extra, como añadir un sonido. Por favor vea el capitulo GladeVCP para entender los fundamentos de los archivos handler GladeVCP. Gscreen utiliza una técnica muy similar.

Temas Gscreen usa el kit de herramientas PyGTK para mostrar la pantalla. Pygtk es el enlace del lenguaje Python con GTK. GTK soporta *temas*. Los temas son una forma de modificar la apariencia de los widgets en la pantalla. Por ejemplo, el color o el tamaño de los botones y los controles deslizantes se pueden cambiar utilizando temas. Hay muchos temas GTK2 disponibles en la web. Los temas también se pueden personalizar para modificar la visualizacion de widgets particulares. Esto vincula el archivo de tema al archivo de glade más firmemente. Algunas de las máscaras de pantalla de muestra permiten al usuario seleccionar cualquiera de los temas en el sistema. La muestra *gscreen* es un ejemplo. Algunos cargarán el tema que tiene el mismo nombre en el archivo de configuración. La muestra *gscreen-gaxis* es un ejemplo. Esto se hace poniendo la carpeta de temas en la carpeta de configuración que tiene los archivos INI y HAL y nombrandola: SCREENNAME_theme (SCREENNAME es el nombre base de los archivos, por ejemplo, gaxis_theme) Dentro de esta carpeta hay otra llamada llamada gtk-2.0, dentro de la cual están los archivos del tema. Si agrega este archivo, Gscreen se establecerá de forma predeterminada en este tema al inicio. gscreen-gaxis tiene un tema personalizado de muestra que busca ciertos widgets con nombre y cambia el comportamiento visual de esos widgets específicos. Estop y los botones de máquina ON utilizan colores diferentes que el resto de los botones para que se destaquen. Esto se hace en el archivo handler dándoles nombres específicos y agregando comandos específicos en el archivo gtkrc del tema. Para obtener información sobre la temática GTK2 (el tema de muestra utiliza el motor de tema de mapa de píxeles). Vea:

Temas GTK

Motor de Temas Pixmap

4.5.2.2. Construir un Panel GladeVCP

Gscreen es solo un gran y complicado panel GladeVCP, con código python para controlarlo. Para personalizarlo necesitamos el archivo Glade cargado en el editor de Glade.

LinuxCNC Instalado Si tiene LinuxCNC 2.6+ instalado en Ubuntu 10.04, simplemente inicie el editor Glade Desde el menú de aplicaciones o desde el terminal. Las nuevas versiones de Linux requieren que instale Glade 3.8.0 - 3.8.6 (posiblemente deberá compilarlo)

Comandos en compilados RIP Usando un compilado de la versión fuente de LinuxCNC, abra un terminal y cambie de directorio a la parte superior de la carpeta LinuxCNC. Configure el entorno entrando *. ./scripts/rip-environment ' y ahora ingrese 'glade*. Usted vera un montón de advertencias en el terminal que puede ignorar y el editor debe abrirse. El archivo de Gscreen Glade está en: src/emc/usr_intf/gscreen/. Las skins de muestra están en /share/gscreen/skins/. Esto debe ser copiado a una carpeta de configuración, o puede hacer un archivo Glade limpio guardándolo en una carpeta de configuración.

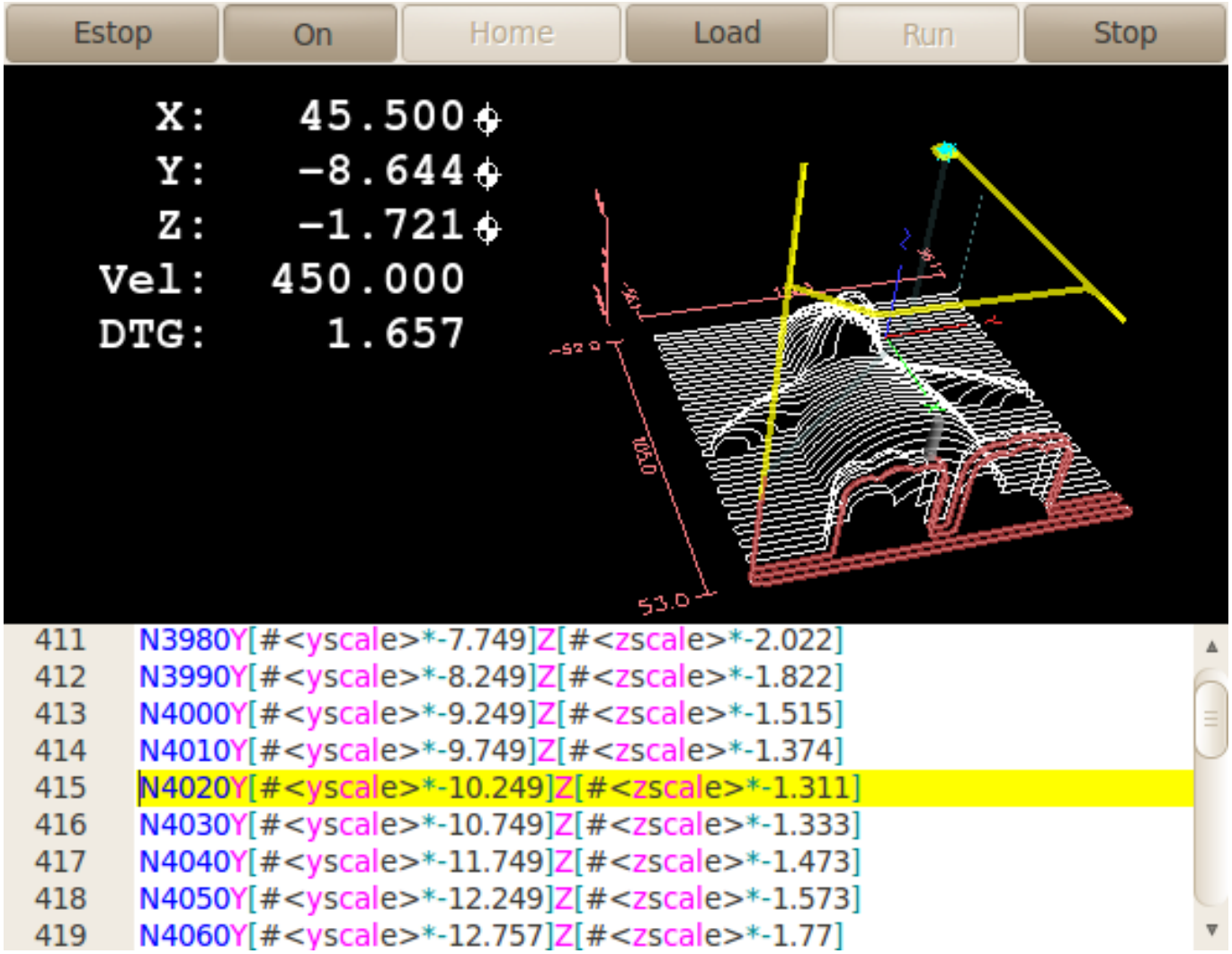
De acuerdo. Ha cargado el archivo Glade de serie y ahora puede editarlo. La primera cosa que notara es que el editor no se parece a lo que muestra. Gscreen usa algunos trucos, como esconder todas las cajas de botones excepto uno y lo cambia dependiendo del modo. Lo mismo ocurre con los cuadernos; algunas pantallas usan cuadernos con pestañas que no se muestran. Para cambiar páginas en el editor necesita mostrar temporalmente esas pestañas.

Al realizar cambios, es mucho más fácil agregar widgets y luego restar widgets y aún así, la pantalla funciona correctamente, lo que hace que los objetos "no sean visibles" es una forma de cambiar la pantalla sin obtener errores. Esto no siempre funcionará con algunos widgets que se establecerá visible de nuevo. Cambiar los nombres de los widgets regulares de Gscreen probablemente no va a funcionar bien sin cambiar el código de Python, pero mover un Widget manteniendo el nombre suele ser viable.

Gscreen aprovecha los widgets de GladeVCP tanto como sea posible, para evitar agregar código python. Aprender acerca de los widgets de GladeVCP es un requisito previo. Si los widgets existentes le brindan la función que desea o necesita, entonces no es necesario agregar código Python; solo guarde el archivo Glade en su carpeta de configuración. Si necesita algo más personalizado, entonces debe hacer algo de programación en Python. El nombre de la ventana principal debe ser window1. Gscreen asume este nombre.

Recuerde, si utiliza una opción de pantalla personalizada, USTED es responsable de actualizarla (si es necesario) al actualizar LinuxCNC.

4.5.3. Construyendo una simple pantalla limpia personalizada

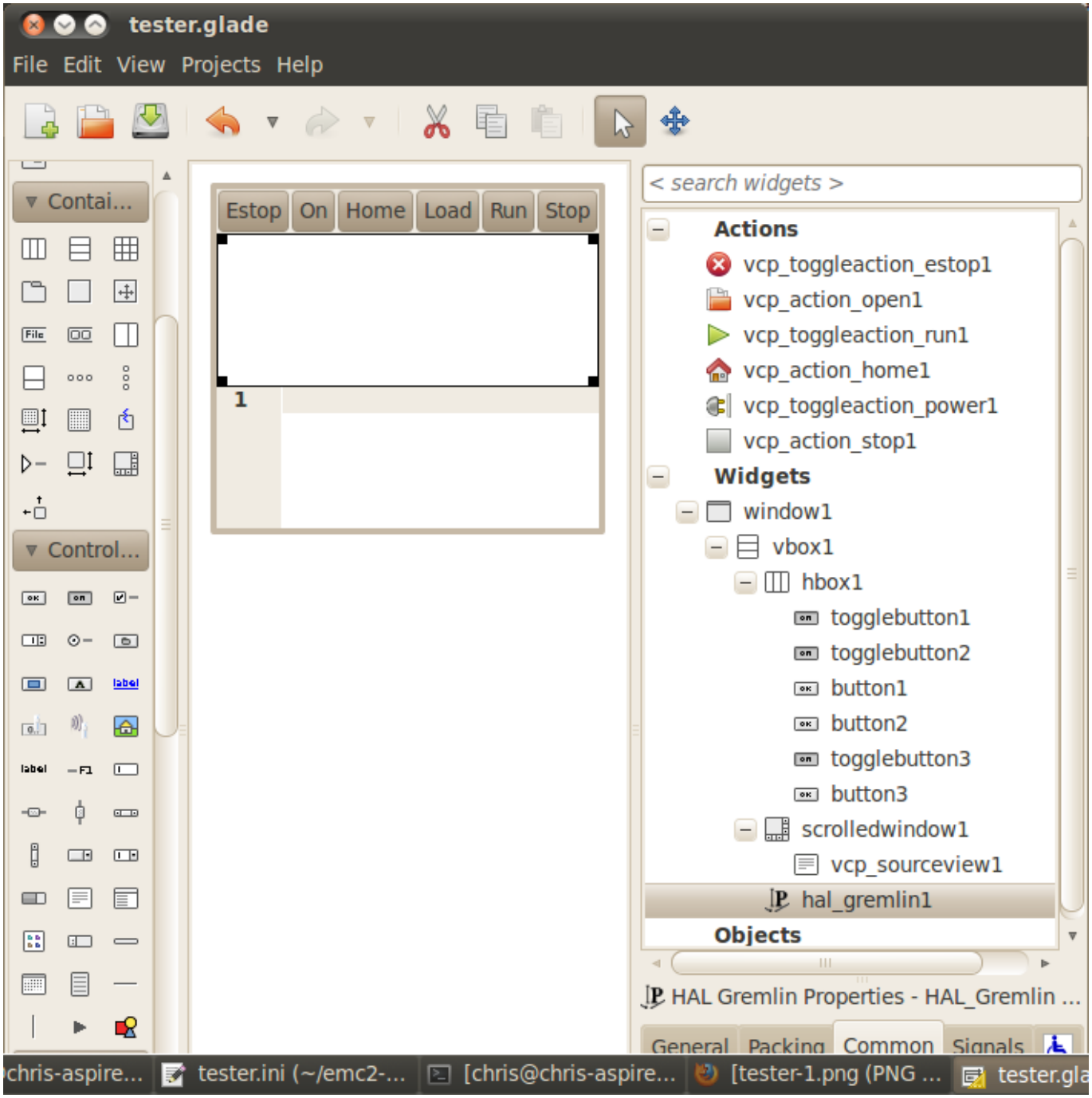


Permite construir una pantalla usable simple. Construya esto en el editor de Glade (si usa un RIP ejecutelo desde un terminal después de usar .scripts/rip-environment).

HAY QUE TENER EN CUENTA QUE:

- La ventana de nivel superior debe llamarse con el nombre predeterminado, *window1* - Gscreen se basa en esto.
- Agregue acciones haciendo clic derecho y luego seleccione *add as toplevel widget* que no agregan nada visual a la ventana pero se agregan a la ventana de lista de acciones de la derecha. Agrega todos los que ves en la parte superior derecha.
- Después de agregar las acciones, debemos vincular los botones a las acciones para que puedan trabajar (ver abajo).
- El widget gremlin no tiene un tamaño predeterminado, por lo que establecer un tamaño es útil (ver más abajo).
- El widget SourceView intentará usar toda la ventana por lo que agregarlo a una ventana con scroll cubrirá esto (esto ya se ha hecho en el ejemplo).
- Los botones se expandirán a medida que la ventana se haga más grande, lo que se ve feo, así que configuraremos el cuadro en el que están para que no expanda (ver más abajo).
- Los tipos de botones que se utilizarán dependen de la acción VCP utilizada -eg vcp_toggle_action por lo general requieren botones alternantes (siga el ejemplo por ahora).

- Los botones en este ejemplo son botones normales, no botones HAL. Nosotros no necesitamos los pines HAL.



En esta pantalla estamos usando VCP_actions para comunicar a LinuxCNC las acciones que queremos. Esto nos permite funciones estándar sin agregar código python en el archivo handler. Vamos a vincular el botón de detención Estop a la acción de detención. Seleccione el el botón de paro y debajo de la pestaña general busque "Acción relacionada" y haga clic en el botón junto a él. Ahora seleccione la acción conmutada estop. Ahora el boton se activará y desactivará cuando se haga clic. Debajo de la pestaña general puede cambiar el texto de la etiqueta del botón para describir su función. Haga esto en todos los botones.

Seleccione el widget Gremlin, haga clic en la pestaña común y establezca la altura solicitada en 100 y haga clic en la casilla de verificación junto a él.

Haga clic en el cuadro horizontal que contiene los botones. Haga clic en la pestaña packing y haga clic en *expandir* a *No*.

Guárdelo como tester.glade en la carpeta sim/gscreen/gscreen_custom/. Ahora inicie LinuxCNC y haga clic en sim/gscreen/gscreen_custom/tester e inícielo. Si todo va bien, nuestra pantalla se abrirá y los botones harán su trabajo. tester.ini le dice a gscreen

que busque y cargue tester.glade y tester_handler.py. El archivo tester_handler.py está incluido en esa carpeta y está codificado para mostrar la pantalla y no mucho más. Desde los widgets especiales, que se comunican directamente con LinuxCNC, todavía puede hacer cosas útiles. Si sus necesidades de la pantalla están cubiertas por los widgets especiales disponibles, entonces ya se tiene la forma de construir una pantalla. Si quiere algo más, todavía hay muchos trucos disponibles simplemente añadiendo *llamadas de función* para obtener un comportamiento fijo codificando su propio código Python para personalizar exactamente lo que desea. Pero eso significa aprender acerca de los archivos handler.

4.5.4. Ejemplo de archivo handler

Hay funciones especiales para las que Gscreen verifica el archivo handler. Si los agrega en su archivo handler, Gscreen los llamará en lugar de las funciones internas del mismo nombre de gscreen.

- initialize_preferences(self): puede instalar nuevas rutinas de preferencias.
- initialize_keybindings(self) Puede instalar nuevas rutinas de enlace de teclas. En la mayoría de los casos no querrá hacer esto, querrá reasignar las llamadas de enlaces a teclas individuales. También puede agregar más enlaces de teclas que llamen a una función arbitraria.
- initialize_pins(self): crea / inicializa pines HAL
- connect_signals(self,handlers): si está utilizando un dispositivo completamente diferente pantalla la pantalla de G por defecto que debe agregar esto o la pantalla de g intentará conectarse Señales a los widgets que no están allí. La función por defecto de Gscreen se llama. with self.gscreen.connect_signals (manejadores) Si desea simplemente agregar extra envía señales a su pantalla, pero aún así desea que las predeterminadas llamen a esto primero y luego añadir más señales. Si las señales son simples (no se pasan datos de usuario), entonces También puede utilizar la selección de señal de Glade en el editor de Glade.
- initialize_widgets(self): puedes usar esto para configurar cualquier widget. Gscreen normalmente llama a *self.gscreen.initialize_widget* que en realidad llama Muchas funciones separadas. Si deseas incorporar alguno de esos widgets. entonces simplemente llame a esas funciones directamente. o agregar self.gscreen.init_show_windows () para que solo se muestren los widgets. Entonces sí deseado, inicialice / ajuste sus nuevos widgets.
- initialize_manual_toolchange (self): permite una renovación completa del manual sistema de cambio de herramientas.
- set_restart_line(self.line):
- timer_interrupt (self): permite que uno complete de nuevo la interrupción rutina Esto se usa para llamar a periodic () y verificar errores de linuxcnc.status.
- check_mode(self): se usa para verificar en qué modo se encuentra la pantalla. Devuelve una lista [] 0 -manual 1- mdi 2- auto 3- jog.
- on_tool_change(self, widget): puede usar esto para anular la herramienta manual diálogo de cambio: esto se llama cuando *gscreen.tool-change* cambia de estado.
- dialog_return(self, dialog_widget, displaytype, pinname): use esto para anular Cualquier mensaje de usuario o cuadro de diálogo de cambio de herramienta manual. Llamado cuando el diálogo es cerrado.
- periódico (auto): Esto se denomina cada milisegundos (predeterminado 100). Usalo para actualiza tus widgets / HAL pins. Puedes llamar a Gscreen regularmente. después también, self.gscreen.update_position() o simplemente agregar pass to no actualizar cualquier cosa El update_position() de Gscreen en realidad llama a muchos por separado funciones Si desea incorporar algunos de esos widgets, simplemente llame esas funciones directamente.

También puede agregar sus propias funciones para llamarlas en este archivo. Usualmente tu agregaría una señal a un widget para llamar a su función.

4.5.4.1. Agregar funciones de combinación de teclas

Nuestro ejemplo de probador sería más útil si respondiera a los comandos del teclado.
Hay una función llamada keybindings () que intenta configurar esto.
Si bien puede anularlo completamente, no lo hicimos, pero asume algunas cosas.
Asume que el botón de alternancia de parada se llama *button_estop* y que la tecla F1 lo controla.
Asume que el botón de encendido se llama *button_machine_on* y la tecla F2 lo controla.
Estos se pueden corregir fácilmente cambiando el nombre de los botones en el editor de Glade para que coincidan.
Pero en cambio, vamos a anular las llamadas estándar y agregar las nuestras.

Agregue estos comandos al archivo de controlador:

```
# Ajustar funciones de Gscreen
# llamadas de combinacion de teclas
def on_keycall_ESTOP(self,state,SHIFT,CNTRL,ALT):
  if state: # only if pressed, not released
    self.widgets.togglebutton1.emit('activate')
    self.gscreen.audio.set_sound(self.data.alert_sound)
    self.gscreen.audio.run()
    return True # stop progression of signal to other widgets
def on_keycall_POWER(self,state,SHIFT,CNTRL,ALT):
  if state:
    self.widgets.togglebutton2.emit('activate')
    return True
def on_keycall_ABORT(self,state,SHIFT,CNTRL,ALT):
  if state:
    self.widgets.button3.emit('activate')
    return True
```

Así que ahora hemos anulado las llamadas de función de Gscreen del mismo nombre y las tratamos en nuestro archivo de manejador.
Ahora hacemos referencia a los widgets por el nombre que usamos en el editor de Glade.
También agregamos una función gscreen incorporada para hacer un sonido cuando cambia Estop.
Tenga en cuenta que llamamos a las funciones integradas de Gscreen que debemos usar self.gscreen. [NOMBRE DE LA FUNCIÓN] ()
Si usamos self. [NOMBRE DE LA FUNCIÓN] () llamaría a la función en nuestro archivo de controlador.

Permite agregar otro enlace de teclas que carga el halómetro cuando se presiona F4.
En el archivo del controlador bajo def initialize_widgets (self): cambie a:

```
def initialize_widgets(self):
  self.gscreen.init_show_windows()
  self.gscreen.keylookup.add_conversion('F4','TEST','on_keycall_HALMETER')
```

Luego agregue estas funciones bajo la clase HandlerClass:

```
def on_keycall_HALMETER(self,state,SHIFT,CNTRL,ALT):
  if state:
    self.gscreen.on_halmeter()
    return True
```

Esto agrega una conversión de combinación de teclas que dirige a gscreen a llamar a on_keycall_HALMETER cuando se presiona F4.
Luego agregamos la función al archivo de identificador para llamar a una función incorporada de Gscreen para iniciar el halómetro.

4.5.4.2. Linuxcnc State Status

The module *Gstat* polls linuxcnc's state every 100ms and sends callback messages to user functions when state changes.
You can register messages to act on specific state changes.
As an example we will register to get *file-loaded* messages when linuxcnc loads a new file.
First we must import the module and instantiate it:
In the import section of the handler file add:

```
from hal_glib import GStat
GSTAT = GStat()
```

In the handler file under *def __init__(self):* add:

```
GSTAT.connect('file-loaded', self.update_filepath)
```

Then in the *HandlerClass*, add the function:

```
self.update_filepath(self, obj, path):
    self.widgets.my_path_label.set_text(path)
```

When linuxcnc loads a new file, Gstat will send a callback message to the funciton *update_filepath*.
In this example whe update a label with that path name (assuming there is a label nammed *my_path_label*) in the GLADE file.

4.5.4.3. Teclas de Jogging

No hay widgets especiales para hacer jogging con botones de pantalla, así que debemos hacerlo con el código de Python.
Bajo la función connect_signals agregue este código:

```
        for i in('x','y','z'):
            self.widgets[i+'neg'].connect("pressed", self['jog_'+i],0,True)
            self.widgets[i+'neg'].connect("released", self['jog_'+i],0,False)
            self.widgets[i+'pos'].connect("pressed", self['jog_'+i],1,True)
            self.widgets[i+'pos'].connect("released", self['jog_'+i],1,False)
        self.widgets.jog_speed.connect("value_changed",self.jog_speed_changed)
```

Agregue estas funciones bajo la clase HandlerClass:

```
    def jog_x(self,widget,direction,state):
        self.gscreen.do_key_jog(_X,direction,state)
    def jog_y(self,widget,direction,state):
        self.gscreen.do_key_jog(_Y,direction,state)
    def jog_z(self,widget,direction,state):
        self.gscreen.do_key_jog(_Z,direction,state)
    def jog_speed_changed(self,widget,value):
        self.gscreen.set_jog_rate(absolute = value)
```

Finalmente, agregue dos botones al archivo GLADE para cada eje, uno para positivo, otro para jogging en dirección negativa.
Nombra estos botones xneg, xpos, yneg, ypos zneg, zpos respectivamente.
agregue un widget SpeedControl al archivo GLADE y llámelo jog_speed

4.5.5. Gscreen Start Up

Gscreen es realmente solo una infraestructura para cargar un archivo GladeVCP personalizado y interactuar con ella

1. Gscreen lee las opciones con las que se inició.
2. Gscreen establece el modo de depuración y establece el nombre de la máscara opcional.
3. Gscreen comprueba si hay archivos *locales* de XML, controladores y / o configuración regional en el carpeta de configuración. Los usará en lugar de los predeterminados. (en compartir / gscreen / skins /) (Puede haber dos pantallas separadas mostradas).
4. Se carga la pantalla principal y se configuran las traducciones. Si presente el segundo Se cargará la pantalla y se configurarán las traducciones.
5. El audio opcional se inicializa si está disponible.
6. Lee algo del archivo INI para inicializar las unidades, y el número / tipo de ejes.
7. Inicializa el enlace de Python a HAL para construir un componente de espacio de usuario con el Nombre de pantalla.
8. Se llama a los makepins de GladeVCP para analizar el archivo XML para crear pines HAL para los widgets de HAL y registrar los widgets conectados de LinuxCNC.
9. Busca un archivo de controlador *local* en la carpeta de configuración o usa otro El stock de la carpeta de la piel.
10. Si hay un archivo manejador, gscreen lo analiza y registra la función. llama al espacio de nombres de Gscreen.
11. Glade compara / registra todas las llamadas de señal a las funciones en gscreen y archivo de controlador.
12. Gscreen comprueba el archivo INI para un nombre de archivo de preferencia de opción de lo contrario utiliza *.gscreen_preferences* =.
13. Gscreen comprueba si hay una llamada a la función de preferencia (*initialize_preferences (self)*) en el archivo del controlador, de lo contrario usa el Gscreen stock uno.
14. Gscreen comprueba el componente en tiempo real classicladder.
15. Gscreen comprueba el tema GTK de todo el sistema.
16. Gscreen recopila los incrementos de jogging del archivo INI.
17. Gscreen recopila los incrementos de jogging angular del archivo INI.
18. Gscreen recopila la tasa de jog predeterminada y máxima del INI.
19. Gscreen recopila la velocidad máxima de cualquier eje de la sección TRAJ del INI.
20. Gscreen comprueba si hay ejes angulares y luego recopila los valores predeterminados y Velocidad máxima desde el archivo INI.
21. Gscreen recopila todos los ajustes de anulación del INI.
22. Gscreen comprueba si se trata de una configuración de torno desde el archivo INI.
23. Gscreen encuentra el nombre de la tabla de herramientas, el editor de herramientas y el archivo param de la INI.
24. Gscreen comprueba el archivo de controlador para la función de enlace de teclas (*initialize_keybindings (self)*) o use Gscreen stock one.
25. Gscreen comprueba el archivo del controlador para la función de pines (*initialize_pins (self)*) o bien usar Gscreen stock uno.
26. Gscreen comprueba el archivo del controlador para la función manual_toolchange (*initialize_manual_toolchange (self)*) o use Gscreen stock one.

27. Gscreen comprueba el archivo de controlador para la función connect_signals (*initialize_connect_signals (self)*) o use Gscreen stock one.
28. Gscreen checka el archivo de controlador para la función de widgets (*initialize_widgets (self)*) o use Gscreen stock one.
29. Gscreen seta up mensajes especificados en el archivo INI.
30. Gscreen le dice a HAL que el componente Hscreen de Gscreen ha terminado de hacer pines y está Listo. Si hay un widget de terminal en la pantalla, se imprimirán todos los Pasadores de pantalla a ella.
31. Gscreen establece el tiempo del ciclo de visualización según el archivo INI.
32. Gscreen comprueba el archivo del controlador para la llamada a la función *timer_interupt (self)* de lo contrario, utilice la función de llamada por defecto de Gscreen.

4.5.6. Configuración INI

Bajo [DISPLAY]:

```
DISPLAY = gscreen -c tester
  opciones:
   -d depurando en
   -v verbose depuración en
```

El interruptor -c permite seleccionar un *skin*. Gscreen asume el archivo Glade y El archivo manejador usa este mismo nombre. La segunda pantalla opcional será la mismo nombre con un 2 (por ejemplo, tester2.glade) No se permite un segundo archivo de controlador. Solo se cargará si está presente. Gscreen buscará el LinuxCNC archivo de configuración que se lanzó primero para los archivos, luego en el sistema carpeta de la piel.

4.5.7. Mensajes de diálogo del usuario

Esta función se utiliza para mostrar mensajes de diálogo emergentes en la pantalla.
Estos están definidos en el archivo INI y controlados por los pines HAL.
Texto en negrita es generalmente un título.
texto está debajo de eso y generalmente más largo.
Detalle está oculto a menos que se haga clic en él.
pinname es el nombre base de los pines HAL.
tipo especifica si se trata de un mensaje de sí / no, ok o de estado.
Los mensajes de estado se mostrarán en la barra de estado y en el cuadro de diálogo de notificación.
no requiere intervención del usuario.
Los mensajes ok requieren que el usuario haga clic en ok para cerrar el diálogo.
los mensajes ok tienen un pin HAL para iniciar el diálogo y uno para indicar que está esperando para la respuesta
los mensajes sí / no requieren que el usuario seleccione los botones sí o no para cerrar el cuadro de diálogo.
los mensajes sí / no tienen tres pines hal: uno para mostrar el cuadro de diálogo, otro para esperar,
y uno para la respuesta.

Aquí hay un código INI de muestra. Estaría bajo el encabezado [DISPLAY].

```
# Esto solo se muestra en la barra de estado y en la ventana emergente de notificación del ↩
    escritorio.
MESSAGE_BOLDTEXT = NONE
MESSAGE_TEXT = This is a statusbar test
MESSAGE_DETAILS = STATUS DETAILS
MESSAGE_TYPE = status
MESSAGE_PINNAME = statustest

# Aparecerá un cuadro de diálogo que hace un sí, no una pregunta
```

```
MESSAGE_BOLDTEXT = NONE
MESSAGE_TEXT = This is a yes no dialog test
MESSAGE_DETAILS = Y/N DETAILS
MESSAGE_TYPE = yesnodialog
MESSAGE_PINNAME = yndialogtest

# Aparece un cuadro de diálogo que requiere una respuesta correcta y se muestra en la barra ↩
     de estado y
# la ventana emergente de notificación de Destop.
MESSAGE_BOLDTEXT = This is the short text
MESSAGE_TEXT = This is the longer text of the both type test. It can be longer then the  ↩
    status bar text
MESSAGE_DETAILS = BOTH DETAILS
MESSAGE_TYPE = okdialog status
MESSAGE_PINNAME = bothtest
```

4.5.7.1. Copie el archivo de serie Handler/Glade para su modificación

Si desea utilizar una pantalla de archivo pero modificar su archivo de controlador, debe copie el archivo de stock a su carpeta de archivo de configuración.
Gscreen verá esto y usará el archivo copiado
¿Pero dónde está el archivo original? Si utiliza un linuxcnc RIP, el las máscaras de muestra están en / share / gscreen / skins / *SCREENNAME*
Las versiones instaladas de linuxcnc las tienen en lugares ligeramente diferentes dependiendo en la distribución utilizada.
Una manera fácil de encontrar la ubicación es abrir una terminal e inicie la pantalla sim que desea usar.
En el terminal se imprimirán las ubicaciones de los archivos.
Puede ser útil agregar el interruptor -d a la línea de carga de gscreen en el INI.

Aquí hay una muestra:

```
chris@chris-ThinkPad-T500 ~/emc-dev/src $ linuxcnc
LINUXCNC - 2.7.14
Machine configuration directory is '/home/chris/emc-dev/configs/sim/gscreen/gscreen_custom'
Machine configuration file is 'industrial_lathe.ini'
Starting LinuxCNC...
Found file(lib): /home/chris/emc-dev/lib/hallib/core_sim.hal
Note: Using POSIX non-realtime
Found file(lib): /home/chris/emc-dev/lib/hallib/sim_spindle_encoder.hal
Found file(lib): /home/chris/emc-dev/lib/hallib/axis_manualtoolchange.hal
Found file(lib): /home/chris/emc-dev/lib/hallib/simulated_home.hal
**** GSCREEN WARNING: no audio alerts available - Is python-gst0.10 libray installed?
**** GSCREEN INFO ini: /home/chris/emc-dev/configs/sim/gscreen/gscreen_custom/ ↩
    industrial_lathe.ini
**** GSCREEN INFO: Skin name = industrial

**** GSCREEN INFO:  Using SKIN glade file from /home/chris/emc-dev/share/gscreen/skins/ ↩
    industrial/industrial.glade ****

**** GSCREEN INFO:  No Screen 2 glade file present
**** GSCREEN INFO: handler file path: ['/home/chris/emc-dev/share/gscreen/skins/industrial/ ↩
    industrial_handler.py']
```

La linea:

```
**** GSCREEN INFO: handler file path: ['/home/chris/emc-dev/share/gscreen/skins/industrial/ ↩
    industrial_handler.py']
```

muestra dónde vive el archivo de stock. Copie este archivo a su carpeta de configuración.
Esto funciona igual para el archivo Glade.

4.6. TkLinuxCNC GUI

4.6.1. Introduction

TkLinuxCNC is one of the first graphical front-ends for LinuxCNC. It is written in Tcl and uses the Tk toolkit for the display. Being written in Tcl makes it very portable (it runs on a multitude of platforms). A separate backplot window can be displayed as shown.

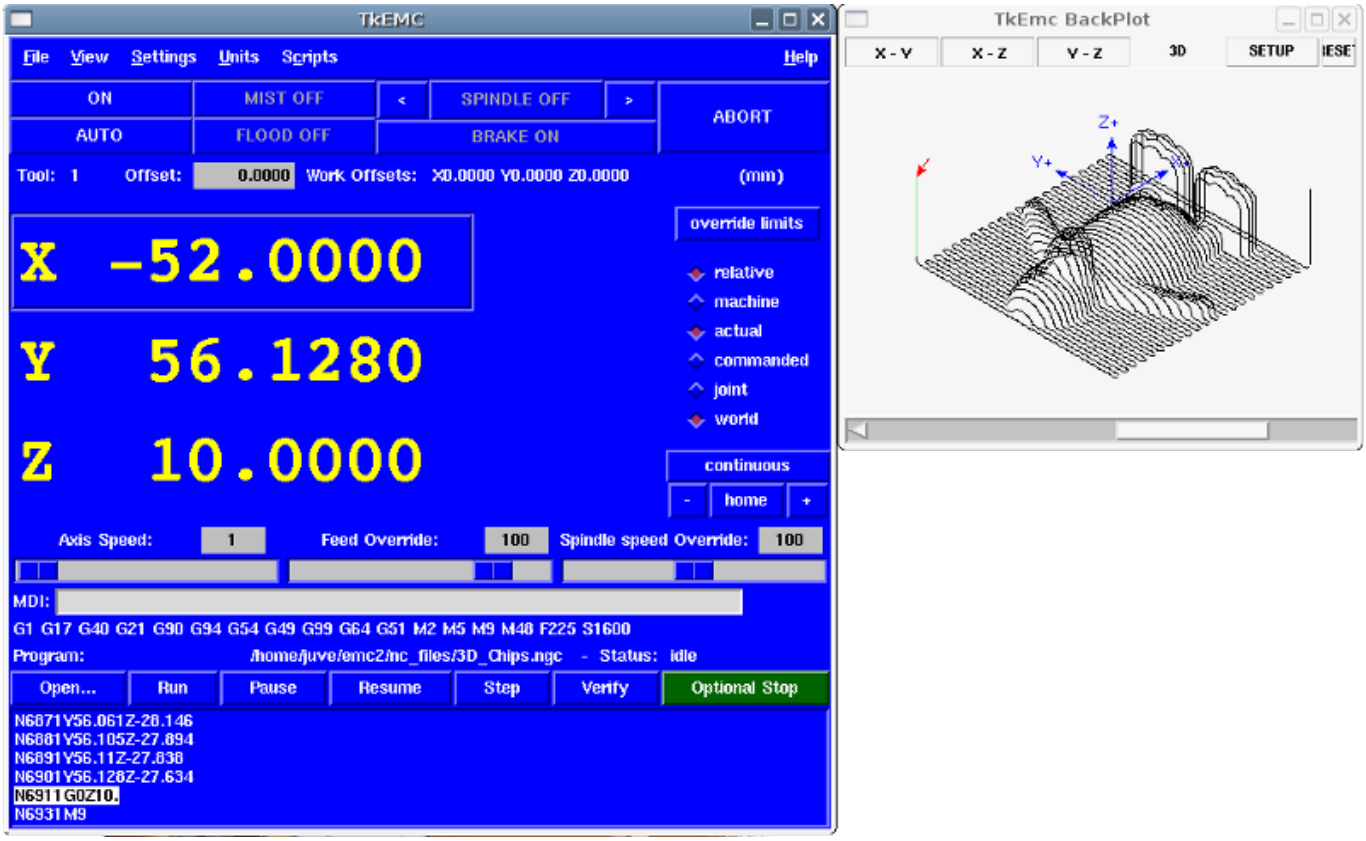


Figura 4.29: TkLinuxCNC Window

4.6.2. Getting Started

To select TkLinuxCNC as the front-end for LinuxCNC, edit the .ini file. In the section *[DISPLAY]* change the *DISPLAY* line to read

```
DISPLAY = tklinuxcnc
```

Then, start LinuxCNC and select that ini file. The sample configuration *sim/tklinuxcnc/tklinuxcnc.ini* is already configured to use TkLinuxCNC as its front-end.

4.6.2.1. A typical session with TkLinuxCNC

1. Start LinuxCNC and select a configuration file.
2. Clear the *E-STOP* condition and turn the machine on (by pressing F1 then F2).
3. *Home* each axis.

4. Load the file to be milled.
5. Put the stock to be milled on the table.
6. Set the proper offsets for each axis by jogging and either homing again or right-clicking an axis name and entering an offset value. [1]
7. Run the program.
8. To mill the same file again, return to step 6. To mill a different file, return to step 4. When you're done, exit LinuxCNC.

4.6.3. Elements of the TkLinuxCNC window

The TkLinuxCNC window contains the following elements:

- A menubar that allows you to perform various actions
- A set of buttons that allow you to change the current working mode, start/stop spindle and other relevant I/O
- Status bar for various offset related displays
- Coordinate display area
- A set of sliders which control *Jogging speed*, *Feed Override* , and *Spindle speed Override* which allow you to increase or decrease those settings
- Manual data input text box *MDI*
- Status bar display with active G-codes, M-codes, F- and S-words
- Interpreter related buttons
- A text display area that shows the G-code source of the loaded file

4.6.3.1. Main buttons

From left to right, the buttons are:

- Machine enable: *ESTOP* > *ESTOP RESET* > *ON*
- Toggle mist coolant
- Decrease spindle speed
- Set spindle direction *SPINDLE OFF* > *SPINDLE FORWARD* . *SPINDLE REVERSE*
- Increase spindle speed
- Abort

then on the second line:

- Operation mode: *MANUAL* > *MDI* > *AUTO*
- Toggle flood coolant
- Toggle spindle brake control

[1] For some of these actions it might be necessary to change the mode LinuxCNC is currently running in.

4.6.3.2. Offset display status bar

The Offset display status bar displays the currently selected tool (selected with Txx M6), the tool length offset (if active), and the work offsets (set by right-clicking the coordinates).

4.6.3.3. Coordinate Display Area

The main part of the display shows the current position of the tool. The color of the position readout depends on the state of the axis. If the axis is unhomed the axis will be displayed in yellow letters. Once homed it will be displayed in green letters. If there is an error with the current axis TkLinuxCNC will use red letter to show that. (for example if an hardware limit switch is tripped).

To properly interpret these numbers, refer to the radio boxes on the right. If the position is *Machine*, then the displayed number is in the machine coordinate system. If it is *Relative*, then the displayed number is in the offset coordinate system. Further down the choices can be *actual* or *commanded*. Actual refers to the feedback coming from encoders (if you have a servo machine), and the *commanded* refers to the position command send out to the motors. These values can differ for several reasons: Following error, deadband, encoder resolution, or step size. For instance, if you command a movement to X 0.0033 on your mill, but one step of your stepper motor is 0.00125, then the *Commanded* position will be 0.0033 but the *Actual* position will be 0.0025 (2 steps) or 0.00375 (3 steps).

Another set of radio buttons allows you to choose between *joint* and *world* view. These make little sense on a normal type of machine (e.g. trivial kinematics), but help on machines with non-trivial kinematics like robots or stewart platforms. (you can read more about kinematics in the Integrator Manual).

Backplot When the machine moves, it leaves a trail called the backplot. You can start the backplot window by selecting View→Backplot.

4.6.3.4. Automatic control

Buttons for control The buttons in the lower part of TkLinuxCNC are used to control the execution of a program: *Open* to load a program, *Verify* to check it for errors, *Run* to start the actual cutting, *Pause* to stop it while running, *Resume* to resume an already paused program, *Step* to advance one line in the program and *Optional Stop* to toggle the optional stop switch (if the button is green the program execution will be stopped on any M1 encountered).

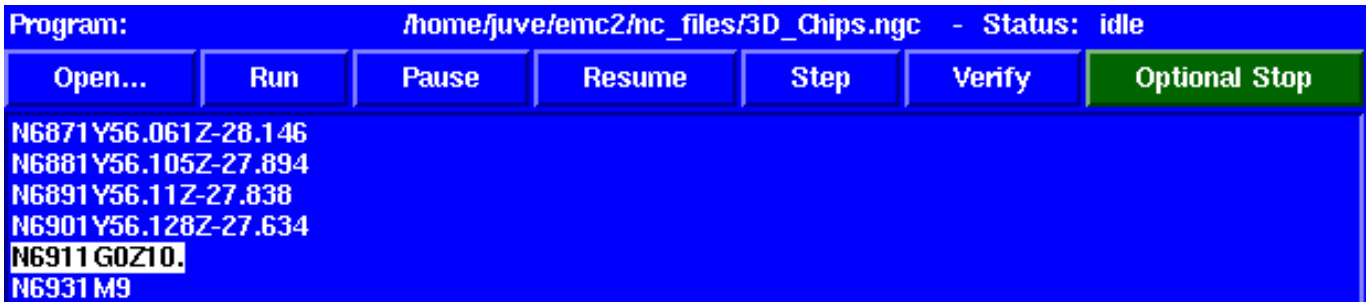


Figura 4.30: TkLinuxCNC Interpreter / program control

Text Program Display Area When the program is running, the line currently being executed is highlighted in white. The text display will automatically scroll to show the current line.

4.6.3.5. Manual Control

Implicit keys TkLinuxCNC allows you to manually move the machine. This action is known as *jogging*. First, select the axis to be moved by clicking it. Then, click and hold the + or - button depending on the desired direction of motion. The first four axes can also be moved by the keyboard arrow keys (X and Y), the PAGE UP and PAGE DOWN keys (Z) and the *[* and *]* keys (A/4th).

If *Continuous* is selected, the motion will continue as long as the button or key is pressed. If another value is selected, the machine will move exactly the displayed distance each time the button is clicked or the key is pressed. The available values are: *1.0000, 0.1000, 0.0100, 0.0010, 0.0001*

By pressing *Home* or the HOME key, the selected axis will be homed. Depending on your configuration, this may just set the axis value to be the absolute position 0.0, or it may make the machine move to a specific home location through use of *home switches*. See the Homing Chapter for more information.

By pressing *Override Limits*, the machine will temporarily be permitted to jog outside the limits defined in the .ini file. (Note: if *Override Limits* is active the button will be displayed using a red color).

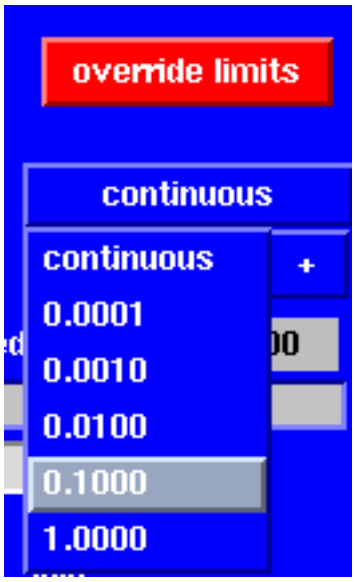


Figura 4.31: TkLinuxCNC Override Limits & Jogging increments example

The Spindle group The button on the first row selects the direction for the spindle to rotate: Counterclockwise, Stopped, Clockwise. The buttons next to it allow the user to increase or decrease the rotation speed. The button on the second row allows the spindle brake to be engaged or released. Depending on your machine configuration, not all the items in this group may have an effect.

The Coolant group The two buttons allow the *Mist* and *Flood* coolants to be turned on and off. Depending on your machine configuration, not all the items in this group may appear.

4.6.3.6. Code Entry

Manual Data Input (also called MDI), allows G-code programs to be entered manually, one line at a time. When the machine is not turned on, and not set to MDI mode, the code entry controls are unavailable.

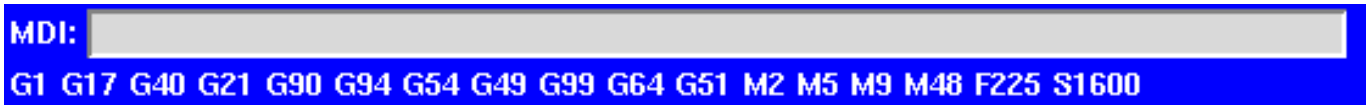


Figura 4.32: The Code Entry tab

MDI: This allows you to enter a g-code command to be executed. Execute the command by pressing Enter.

Active G-Codes This shows the *modal codes* that are active in the interpreter. For instance, *G54* indicates that the *G54 offset* is applied to all coordinates that are entered.

4.6.3.7. Jog Speed

By moving this slider, the speed of jogs can be modified. The numbers above refer to axis units / second. The text box with the number is clickable. Once clicked a popup window will appear, allowing for a number to be entered.

4.6.3.8. Feed Override

By moving this slider, the programmed feed rate can be modified. For instance, if a program requests *F60* and the slider is set to 120%, then the resulting feed rate will be 72. The text box with the number is clickable. Once clicked a popup window will appear, allowing for a number to be entered.

4.6.3.9. Spindle speed Override

The spindle speed override slider works exactly like the feed override slider, but it controls to the spindle speed. If a program requested S500 (spindle speed 500 RPM), and the slider is set to 80%, then the resulting spindle speed will be 400 RPM. This slider has a minimum and maximum value defined in the ini file. If those are missing the slider is stuck at 100%. The text box with the number is clickable. Once clicked a popup window will appear, allowing for a number to be entered.

4.6.4. Keyboard Controls

Almost all actions in TkLinuxCNC can be accomplished with the keyboard. Many of the shortcuts are unavailable when in MDI mode.

The most frequently used keyboard shortcuts are shown in the following table.

Cuadro 4.2: Most Common Keyboard Shortcuts

Keystroke	Action Taken
F1	Toggle Emergency Stop
F2	Turn machine on/off
`, 1 .. 9, 0	Set feed override from 0% to 100%
X, `	Activate first axis
Y, 1	Activate second axis
Z, 2	Activate third axis
A, 3	Activate fourth axis
Home	Send active axis Home
Left, Right	Jog first axis
Up, Down	Jog second axis
Pg Up, Pg Dn	Jog third axis
[,]	Jog fourth axis
ESC	Stop execution

4.7. Guía del usuario de PlasmaC

4.7.1. Licencia

PlasmaC y todo su software relacionado se lanza bajo GPLv2.

4.7.2. Introducción

PlasmaC es una configuración de corte por plasma para LinuxCNC v2.8 o posterior. Incluye un componente HAL más configuraciones de GUI para Axis y Gmoccapy. También hay una opción para que la GUI Axis se muestre en modo vertical, consulte archivo .ini de Axis.

Debe ejecutarse en cualquier hardware que sea compatible con LinuxCNC siempre que haya suficientes pines de E/S para la configuración.

Si es nuevo en el uso del plasma, puede valer la pena leer el documento Plasma CNC Primer que es una introducción genérica al corte por plasma CNC.

PlasmaC utiliza uno de tres modos de operación diferentes.

Modo	Descripción
0	Use voltaje de arco externo para calcular el voltaje de arco y arco OK
1	Use voltaje de arco externo para calcular el voltaje de arco Use un arco externo ok para Arc OK
2	Use el arco externo ok para Arc OK Use arriba/abajo externo para THC

NOTA: Si su cortadora de plasma tiene una salida Arc OK (Transfer), se recomienda usarla para Arc OK en lugar del Arc OK soft del modo 0.

No se requiere movimiento del eje Z en G-Code; de hecho, las configuraciones estándar de PlasmaC comentarán cualquier comando Z durante la carga del programa.

Para un uso confiable de PlasmaC, usted **NO** debe usar ningún offset del eje Z que no sea offset del sistema de coordenadas (G54 ~ G59.3).

Normalmente, es mejor un touch off de su eje Z al límite mínimo del eje Z durante la configuración inicial y luego dejarlo así.

4.7.3. Requisitos de E/S

Esta sección se centra solo en la E/S requerida para PlasmaC; los requisitos de la máquina base, como interruptores de límite, interruptores de inicio, etc., son añadidos.

Name	Modes	Description
Arc Voltage	0,1	Entrada analógica para leer el voltaje de arco Normalmente conectado a una salida de velocidad de un codificador
Float Switch	0,1,2	Entrada digital. Opcional, ver abajo Conectado al interruptor en la cabeza flotante Si no utiliza una sonda óhmica, este es el sensor de sonda principal Si usa una sonda óhmica, este es el sensor de sonda de reserva
Ohmic Probe	0,1,2	Entrada digital. Opcional, ver abajo Conectado a la salida de la sonda Si se usa, este es el sensor de sonda principal
Ohmic Probe Enable	0,1,2	Salida digital. Opcional, solo se requiere si se usa un sondeo óhmico Conectado a la sonda que habilita la entrada Este pin se usa para armar la sonda óhmica
Breakaway Switch	0,1,2	Entrada digital. Opcional, ver abajo Conectado al interruptor de ruptura de la antorcha Detecta si la antorcha se ha desprendido de su cuna.
Torch On	0,1,2	Salida digital Conectado a la antorcha en la entrada de la fuente de alimentación de plasma
Move Up	2	Entrada digital Conectado a la salida ascendente del control externo de THC
Move Down	2	Entrada digital Conectado a la salida baja del control externo de THC

Scribe Arming	0,1,2	Salida digital Conectado al circuito de armado del grabador
Scribe On	0,1,2	Salida digital Conectado al circuito grabador ON

Solo se requiere uno de entre **Float Switch** o **Ohmic Probe**. Si se usan ambos, entonces **Float Switch** será una alternativa si **Ohmic Probe** no se detecta.

Si se utiliza **Ohmic Probe**, se requiere **Ohmic Probe Enable**.

Breakaway Switch no es obligatorio porque **Float Switch** se trata igual que una ruptura cuando no se está sondeando. Si fueran dos interruptores separados, podrían combinarse y conectarse como **Float Switch**

NOTA: El requisito mínimo de E/S para una configuración de PlasmaC es entrada de **Arc Voltage**, entrada de **Float Switch** y salida de **Torch On**. En este caso, PlasmaC trata el interruptor Float como un interruptor de ruptura cuando no está sondeando.

IMPORTANTE: ESTOS PINES SE ENTRARÁN DESPUÉS EN EL CONFIGURADOR, **NO** DEBEN SER USADOS EN EL .HAL DE SU MÁQUINA BASE.

4.7.4. Instalar LinuxCNC

El método preferido para instalar LinuxCNC con PlasmaC es usar una imagen ISO como se describe a continuación.

El requisito mínimo para PlasmaC es LinuxCNC v2.8.

NOTA: Es posible instalar y ejecutar LinuxCNC en una variedad de distribuciones de Linux, pero esto está fuera del alcance de la Guía del usuario de Plasmac. Si desea seguir esta via, deberá instalar su distribución de Linux preferida y luego instalar LinuxCNC v2.8 o posterior junto con las dependencias requeridas; luego puede abrir LinuxCNC y continuar con Test de Instalación LinuxCNC

Todo el desarrollo de PlasmaC se ha realizado en una instalación Stretch ISO.

4.7.4.1. Si no tiene Linux instalado

LinuxCNC requiere un kernel de Linux en tiempo real y actualmente hay dos imágenes ISO disponibles desde descargas de LinuxCNC con diferentes opciones en tiempo real.

1. PREEMPT_RT - la opción preferida si se utiliza una tarjeta de movimiento externa. Bueno para puerto paralelo si su máquina tiene una latencia aceptable. Se requiere si tiene la intención de utilizar una tarjeta Ethernet de Mesa Electronics.
2. RTAI - bueno para el puerto paralelo, normalmente proporciona una mejor latencia que PREEMPT_RT.

Las imágenes ISO descargables son:

- PREEMPT_RT - http://www.linuxcnc.org/testing-stretch-rtpreempt/
- RTAI - http://www.linuxcnc.org/iso/linuxcnc-2.7.14-wheezy.iso

Descargue la imagen elegida y escríbala en un dispositivo de arranque, luego inicie desde ese dispositivo y ejecútelo en modo Live para verificar la latencia de la máquina. La prueba de latencia se encuentra en el menú principal en CNC. Realmente debería tener Max Jitter menor de 50000, cuanto más bajo, mejor. Cuando esté satisfecho con el rendimiento, reinicie y seleccione instalar en el menú de inicio y continúe instalando el sistema.

Esto le dará una máquina con la rama estable actual (v2.7) de LinuxCNC que se actualizará a 2.8 en la siguiente sección.

4.7.4.2. Si tiene LinuxCNC v2.7

El método descrito aquí se denomina Instalación Buildbot, ya que utiliza paquetes preconstruidos de LinuxCNC Buildbot para instalar y actualizar LinuxCNC.

NOTA: Existe otro método llamado Run In Place que requiere que el usuario compile LinuxCNC localmente. Este método se usa normalmente para el trabajo de desarrollo y está más allá del alcance de este manual. Para obtener más información sobre una instalación Run In Place, consulte Construir LinuxCNC

Para usar los paquetes buildbot en su computadora, primero agregue la clave de firma del archivo buildbot a su keychain apt ejecutando este comando desde una terminal:

```
sudo apt-key adv --keyserver hkp://keys.gnupg.net --recv-key E0EE663E
```

Si obtiene un error como GPG: la recepción del servidor de claves falló: No dirmngr

entonces ingrese lo siguiente en la terminal:

```
sudo apt-get install dirmngr
```

El paquete dirmngr que falta se instalará.

Si aún recibe un error al descargar la clave, puede deberse a estar detrás de un firewall o un servidor proxy. En este caso, puede intentar especificar el puerto 80 en lugar de usar el valor predeterminado 11371:

```
sudo apt-key adv --keyserver hkp://keys.gnupg.net: 80 --recv-key E0EE663E
```

Cierre la terminal y, desde el Menú principal, abra el Administrador de paquetes Synaptic.

Seleccione Buscar en la barra de menú y realice una búsqueda utilizando **linuxcnc** como término de búsqueda. Haga clic derecho en cualquier resultado de linuxcnc y luego haga clic en Marcar para eliminación. Cuando se seleccionan todos los resultados instalados, haga clic en Aplicar en la barra de menú y se eliminarán.

Haga clic en Configuración en el menú y seleccione Repositorios.

A continuación, puede editar o eliminar entradas existentes de LinuxCNC de la lista de repositorios. Debe terminar con el siguiente conjunto de estrofas como las únicas referencias habilitadas para LinuxCNC:

```
deb     http://buildbot.linuxcnc.org/ <distro> <branch> - <rt_type>
deb-src http://buildbot.linuxcnc.org/ <distro> <branch> - <rt_type>
```

Dónde:

- <distro> es stretch o wheezy
- <rama> es 2.8 o master
- <rt_type> es rtpreempt o rt

Por tanto, si instaló la versión **stretch** y deseaba la version **2.8** en una máquina **PREEMPT_RT**, entonces tendría:

```
deb     http://buildbot.linuxcnc.org/ stretch 2.8-rtpreempt
deb-src http://buildbot.linuxcnc.org/ stretch 2.8-rtpreempt
```

Cuando esto se haya completado, Synaptic debería hacer una recarga del paquete, o bien hacer una recarga manual desde la barra de menú.

Ahora haga una búsqueda con **linuxcnc** como término de búsqueda y el resultado debería ser una lista de los paquetes LinuxCNC recientemente disponibles

Haga clic derecho en el paquete que desea instalar y luego seleccione Marcar para la instalación.

Para PREEMPT_RT seleccione linuxcnc-uspace y para RTAI, linuxcnc.

También puede seleccionar la documentación si desea instalarla localmente y/o el paquete -dev si tiene la intención de construir nuevos componentes en tiempo real y/o front-end alternativos para linuxcnc.

Cuando haya hecho todas sus selecciones, haga clic en Aplicar en la barra de menú y se instalarán los paquetes seleccionados.

Vaya a probando la instalación de linuxCNC

NOTA: Para obtener más información sobre LinuxCN Buildbot, consulte la página home de LinuxCNC Buildbot

4.7.4.3. Prueba de instalación de LinuxCNC

El Selector de configuración de LinuxCNC debería abrirse.

Vaya a Configuraciones de muestra> by_machine> plasmac.

Seleccione Axis o Gmoccapy.

Seleccione métrico o imperial.

La configuración de simulación de PlasmaC seleccionada se abrirá y se podrá probar a voluntad.

Ahora tiene todos los requisitos para ejecutar PlasmaC, el siguiente paso es crear una configuración de trabajo para su máquina.

4.7.4.4. Hacer una configuración de trabajo de máquina base

NOTA: La configuración de la máquina base significa un sistema de trabajo completo sin conexiones de plasma. Todos los ejes deben estar funcionando y ajustados para un mejor rendimiento y todos los interruptores de inicio y de límite, si están instalados, deben funcionar correctamente.

IMPORTANTE: **NO** AGREGUE NINGUNA CONEXIÓN DE PLASMA COMO TORCH_ON O ARC_OK ETC. AQUÍ. ESTOS SERÁN AGREGADOS MÁS TARDE DURANTE LA CONFIGURACIÓN PlasmaC.

- Cabezal flotante: se recomienda que se use un cabezal flotante y que tenga suficiente movimiento para permitir rebasamientos durante el sondeo. El exceso puede calcularse utilizando la siguiente fórmula:

```
o = 0.5 $\times$ a $\times$ (v $\div$ a)^2
```

donde: o = rebasamiento, a = aceleración en udes/seg2 y v = velocidad en udes/seg

Para una máquina métrica con MAX_ACCELERATION en eje Z de 600 mm/s^2 y MAX_VELOCITY de 60 mm/s, el rebasamiento sería de 3 mm

Para una máquina imperial con MAX_ACCELERATION en eje Z de 24 in/s^2 y MAX_VELOCITY de 2.4 in/s, el rebasamiento sería 0.12 in

También se recomienda que incluso si tiene la intención de utilizar el sondeo óhmico como su sistema de sondeo primario, también tenga un interruptor en el cabezal flotante como respaldo en caso de falla de la sonda óhmica debido a superficies sucias.

Las configuraciones recomendadas para el eje Z son:

- Z MINIMUM_LIMIT debe estar justo debajo de la parte superior de los listones con márgenes para float_switch_travel y tolerancia. Por tanto, si su interruptor float tomó 4 mm (0.0157") en activarse, configure su Z mínimo 5 mm (0.096) debajo del listón más bajo.
- Z MAXIMUM_LIMIT debe ser lo más alto que desea que se mueva el eje Z (no debe ser inferior a Z HOME_OFFSET)
- Z HOME debe estar justo por debajo del límite máximo.

IMPORTANTE: DEBE PODER PONER EN HOME SU MÁQUINA, HACER TOUCH OFF, JOG A LÍMITES SOFT SIN CHOQUES Y EJECUTAR ALGUNOS CÓDIGOS G DE PRUEBA SIN ERRORES.

Cuándo y SOLO CUANDO esta etapa se ha completado con éxito, puede configurar PlasmaC en su máquina ya funcional.

Puede crear una máquina base manualmente o puede usar ayudantes de configuración existentes:

NOTA: Es mejor mantener simple la configuración de la máquina base hasta que la haya probado y ajustado completamente. Si está utilizando Stepconf o Pncconf, anule la selección de Panel VCP, husillo, cambio manual de herramientas y las opciones de classic ladder; estos pueden agregarse manualmente más adelante si los necesita.

Si usa una placa Mesa Electronics, use el Asistente pncconf:

```
$ pncconf
```

Si usa un puerto paralelo, use el Asistente stepconf:

```
$ stepconf
```

Si usa una placa de Pico Systems:
Este hilo del foro LinuxCNC puede ayudar.

Si tiene una configuración de pórtico de doble motor, es posible que deba editar manualmente su configuración para adaptarla a:
https://forum.linuxcnc.org/49-basic-configuration/33079-how-to-2-or-more-motors-on-one-axis-gantry-linuxcnc-2-8-master?start=0
[Este hilo del foro LinuxCNC] puede ayudar.

PRECAUCIÓN: **NO** CONTINÚE HASTA QUE LA MÁQUINA BASE ESTÉ CONFIGURADA Y FUNCIONAL.

4.7.5. Agregar una configuración de PlasmaC a la máquina base

NECESITA tener una configuración de máquina base totalmente probada y en funcionamiento sin ninguna conexión a equipos específicos de plasma antes de continuar.

PRECAUCIÓN: **NO** PROCEDA HASTA QUE SE HAYA HECHO

NOTA: SI UTILIZA UNA TARJETA **Mesa Electronics THCAD** PARA MEDIR EL VOLTAJE DEL ARCO, CONSULTE Mesa THCAD ANTES DE CONTINUAR

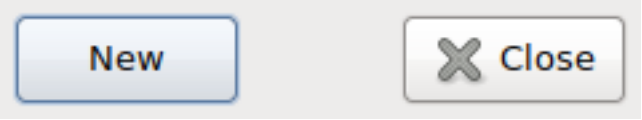
La instalación de una configuración de Plasmac se realiza desde el Configurador.

Desde una terminal:

```
$ python /usr/share/doc/linuxcnc/examples/sample-configs/by_machine/plasmac/configurator.py
```

4.7.5.1. Configurar

La ventana de selección ahora es visible:



Seleccione **Nuevo** en la ventana de selección, esto muestra un cuadro de diálogo de información. Seleccione **Continuar** y se mostrará la ventana Nueva configuración.

NOTA: Habrá diferentes campos visibles dependiendo del modo que seleccione.

Mandatory Settings

Machine Name:

INI file in existing working config:

HAL file in existing working config:

Use arc voltage for both Arc-OK and THC

Mode: 0 Mode: 1 Mode: 2

Arc Voltage HAL pin: (float in)

Torch On HAL pin: (bit out)

Run Frame is a tab behind preview

Run Tab Run Panel

Optional Settings

Float Switch HAL pin: (bit in)

Breakaway Switch HAL pin: (bit in)

Ohmic Probe HAL pin: (bit in)

Ohmic Probe Enable HAL pin: (bit out)

Scribe Arming HAL pin: (bit out)

Scribe On HAL pin: (bit out)

Powermax Com Port: (e.g. /dev/ttyUSB0)

Create Back

Los ejemplos de conexión HAL en la siguiente tabla muestran un ejemplo para una configuración de puerto paralelo y una configuración Mesa 7i96. Cambie estos nombres de pin para adaptarse a su configuración.

Campo	Descripción	Ejemplos
Nombre de la máquina	el nuevo nombre de su máquina. Esto creará un directorio ~/linuxcnc/<name> y su archivo .ini será <name>.ini	plasma_table
Ini File	este es el archivo .ini que creó para su máquina base	base.ini
Archivo HAL	este es el archivo .hal que creó para su máquina base	base.hal
Modo	seleccione el modo que necesita 0 si usa una señal de voltaje de arco para Arc Voltage y Arc OK 1 si usa arco ok externo para Arc OK 2 si se usa arriba/abajo desde un THC externo	1
Arc Voltage	Modos 0 y 1 solamente el pin HAL al que ha conectado su señal de voltaje de arco	encoder.0.velocity hm2_7i96.0.encoder.00.velocity
Torch On	el pin HAL al que ha conectado su antorcha	parport.0.pin-16-out hm2_7i96.0.ssr.00.out-00
Arc OK	Modos 1 y 2 solamente el pin HAL al que ha conectado su señal de arco ok	parport.0.pin-10-in-not hm2_7i96.0.gpio.008.in

Ohmic Probe	solo se requiere si se utiliza sondeo óhmico el pin HAL al que ha conectado su sonda óhmica	parport.0.pin-11-in hm2_7i96.0.gpio.007.in
Ohmic Probe Enable	solo se requiere si se usa el sondeo ohmic el pin HAL al que ha conectado la habilitacion de su sonda ohmica	parport.1.pin-01-out hm2_7i96.0.ssr.00.out-01
Float Switch	solo si se usa un interruptor flotante el pin HAL al que ha conectado su interruptor flotante	parport.0.pin-12-in hm2_7i96.0.gpio.006.in
Interruptor de Ruptura	solo si se usa un interruptor de ruptura el pin HAL al que ha conectado su interruptor de ruptura	parport.0.pin-13-in hm2_7i96.0.gpio.005.in
Move Up	Modo 2 solamente el pin HAL al que ha conectado su señal de movimiento ascendente	parport.1.pin-10-in hm2_7i96.0.gpio.004.in
Move Down	Modo 2 solamente el pin HAL al que ha conectado su señal de movimiento hacia abajo	parport.1.pin-11-in hm2_7i96.0.gpio.003.in
Run Panel	Ejecutar pestaña = el marco de ejecución es una pestaña detrás de la pestaña de vista previa Panel de ejecución = el marco de ejecución es un panel en el lateral de la GUI	Pestaña Ejecutar
Scribe Arming	solo se requiere si se usa un grabador el pin HAL para armar el grabador	parport.1.pin-16-out hm2_7i96.0.ssr.00.out-02
Scribe On	solo es necesario si se usa un grabador el pin HAL para activar el grabador	parport.1.pin-16-out hm2_7i96.0.ssr.00.out-03
Powermax Comms	solo es necesario si se utilizan comunicaciones Powermax el puerto serie utilizado para las comunicaciones	/dev/ttyUSB0

NOTA: Si no está seguro del nombre completo de un pin HAL, inicie LinuxCNC para su máquina base y ejecute **HalShow** para obtener una lista completa de todos los pines HAL.

Complete las entradas requeridas para adaptarse al cableado de su máquina, haga clic en **Crear** y debería terminar con una configuración de PlasmaC en funcionamiento en ~/linuxcnc/configs/<machine_name>

Esto se puede ejecutar de la siguiente manera:

```
$ linuxcnc ~/linuxcnc/configs/<nombre de máquina>/<nombre de máquina>.ini
```

Después de crear una nueva configuración, se requiere algunos ajustes iniciales.

4.7.5.2. Ajuste inicial

LinuxCNC ahora debería ejecutarse con los paneles PlasmaC visibles.

Abra el Panel de configuración y asegúrese de que su máquina cumpla cada una de estos ajustes.

Consulte Panel de configuración

1. MIMIMUM_LIMIT del eje Z debe establecerse por debajo de los listones la cantidad de recorrido del interruptor flotante más una pequeña tolerancia. Digamos que su interruptor flotante se mueve 4 mm (0.0157") antes de que se active el interruptor, entonces debe establecer el límite mínimo del eje Z en 5 mm (0.096) por debajo de la posición más baja del listón.
2. MAXIMUM_LIMIT del eje Z debe ser lo más alto que desea para el recorrido del eje Z; no debe ser inferior a HOME_OFFSET Z pero no necesita estar en la parte superior del recorrido.
3. HOME del eje Z debe estar en o justo debajo de MAXIMUM_LIMIT del eje Z.
4. Home para el eje Z.
5. Asegúrese de que no haya nada debajo de la antorcha, luego mueva Z hacia abajo hasta que se detenga en MIMIMUM_LIMIT del eje Z y luego toque el eje Z con el botón Touchoff en offset cero.

6. Haga home Z de nuevo.

Si usa un interruptor flotante, deberá establecer el offset en el Panel de configuración.

1. Verifique que la velocidad de la sonda y la altura de la sonda en el Panel de configuración sean correctas. PlasmaC puede sondear a la velocidad máxima del eje Z siempre que su máquina tenga suficiente movimiento en el interruptor flotante para absorber cualquier rebasamiento. Si su máquina es adecuada, puede establecer la altura de la sonda cerca del mínimo del eje Z y hacer todas las pruebas a toda velocidad.
2. Haga home de la máquina si aún no lo tiene o no está en la posición home.
3. Coloque algo de material sobre las tablillas debajo de la antorcha.
4. Presione el botón Probe Test.
5. El eje Z sondeará hacia abajo, encontrará el material y luego se moverá hacia la altura de perforación. Esperará allí el tiempo establecido en el archivo ini, el valor predeterminado es 30 segundos (puede cambiar esto si lo desea). Después de esto, se elevará a la altura inicial.
6. Mida la altura sobre el material mientras espera en la altura de perforación.
7. Si mide más que la altura de perforación, reduzca el recorrido del flotante el valor de la diferencia; si es menor que la altura de perforación, aumente el recorrido del flotante esa diferencia.
8. Después de que el eje Z se eleve, repita desde el n.° 4 anterior hasta que la altura de perforación sea correcta.

IMPORTANTE: SI UTILIZA UN **THCAD de Mesa Electronics** EL VALOR **Voltage Scale** FUE OBTENIDO MATEMÁTICAMENTE. SI USTED PRETENDE UTILIZAR VOLTAJES DE CORTE DE LOS FABRICANTES DE GENERADOR DE PLASMA, ENTONCES SERÁ RECOMENDABLE HACER MEDICIONES DE VOLTAJES ACTUALES Y SINTONIZAR **Voltage Scale** Y **Voltage Offset**.

PRECAUCIÓN: LOS VOLTAJES DE CORTE DE PLASMA PUEDEN SER LETALES, SI NO TIENE EXPERIENCIA EN ESTAS MEDIDAS, OBTENGA ALGUNA AYUDA CALIFICADA.

4.7.6. Filtro LOWPASS

El componente PlasmaC HAL tiene un filtro de paso bajo incorporado que tiene el mismo comportamiento que el componente HAL lowpass de LinuxCNC. Este filtro, si se usa, se aplica a la salida **plasmac.arc-tension**. El nombre del pin que se usará para este filtro es **plasmac.lowpass-frequency** y, como su nombre indica, usted ingresa la frecuencia de corte que requiere; si la frecuencia es cero, entonces no se aplica ningún filtrado.

Se sugiere que si desea utilizar el filtro, la entrada debe estar en el archivo <machine_name>_connections.hal en su directorio de configuración.

```
setp plasmac.lowpass-frequency 100
```

Daría una frecuencia de corte de 100Hz.

4.7.7. Antirebote de contactos

El rebote de contactos o interferencias externas pueden causar un comportamiento inconsistente de los siguientes interruptores:

- Interruptor flotante
- Sonda óhmica
- Interruptor de ruptura

Esto puede aliviarse cambiando el valor de antirebote en el archivo <machine_name>_connections.hal. En una configuración estándar, este valor se aplica a los tres conmutadores simultáneamente, pero puede separarlos utilizando un grupo exclusivo de eliminación de rebotes para cada uno.

Cada incremento cambia el tiempo de rebote en un ciclo de hilo servo y para los interruptores Float y Ohmic equivale a un aumento de 0.001mm (0.00004") en el resultado de la altura de la sonda.

El antirebote se logra utilizando el componente HAL debounce.

Si necesita eliminar el rebote para otros interruptores como home o límites, etc., cambie la siguiente línea en <machine_name>_connectio

```
loadrt debounce cfg=3
```

a:

```
loadrt debounce cfg=3,n
```

siendo n la cantidad de contactos que necesitan eliminacion de rebote.

Esto creará un nuevo grupo de rebote llamado debounce.1 para su uso, luego agregue las siguientes líneas:

```
setp debounce.1.delay c
addf debounce.1 servo-thread
```

siendo c el número de ciclos servo para el tiempo de debounce.

Luego, puede eliminar el rebote en cualquier pin HAL que necesite en cualquiera de los archivos .hal (excepto postgui.hal) como en el siguiente ejemplo:

```
net home-x-in parport.1.pin-11-in-not => debounce.1.0.in
net home-x-out debounce.1.0.out => joint.0.home-sw-in
```

4.7.8. Lanzador de escritorio

Puede crear un lanzador de escritorio para su configuración haciendo clic derecho en el escritorio y seleccionando Crear iniciador o similar. Aparecerá un cuadro de diálogo para crear un iniciador. Déle un nombre corto, ingrese cualquier cosa para el comando y haga clic en Aceptar.

Después de que aparezca el lanzador en el escritorio, haga clic derecho sobre él y luego abra con el editor de su elección. Edite el archivo para que se vea similar a esto:

```
[Desktop Entry]
Comment=
Terminal=false
Name=LinuxCNC
Exec=sh -c "linuxcnc $HOME/linuxcnc/configs/<machine_name>/<machine_name>.ini"
Type=Application
Icon=/usr/share/pixmaps/linuxcncicon.png
```

Si desea que se abra una ventana de terminal detrás de la ventana de la GUI, cambie la línea de Terminal a:

```
Terminal=true
```

Mostrar un terminal puede ser útil para visualizar mensajes de error y mensajes de información.

4.7.9. Archivos Plasmac

Después de una instalación, se crean los siguientes archivos en el directorio de configuración:

Nombre de archivo	Función
<machine_name>.ini	archivo de configuración
<machine_name>.hal	Conexiones HAL para el componente PlasmaC
<machine_name>_connections.hal	Conexiones HAL a los pines HAL de E / S
postgui.hal	un archivo HAL ejecutado después de carga de la GUI para personalización de usuario

NOTA: Los comandos personalizados están permitidos en <machine_name>_connections.hal y los archivos postgui.hal, ya que no se sobrescriben durante las actualizaciones.

Además de los archivos anteriores, se crean los siguientes enlaces a los archivos en el directorio de origen:

Nombre de archivo	Función
blank.ngc	para cargar el archivo de biblioteca de formas Gmoccapy
configurator.py	nueva configuración PlasmaC o actualiza/reconfigura una existente
xxx_startup.ngc	G-Code inicial para máquinas imperial o métrica
M190	G-Code para el cambio de material
materialverter.py	convertidor de tabla de herramientas y creador de archivos de material
plasmac_axis.py	código python para personalizar la GUI Axis
plasmac_xxx.glade	panel gladevcp para la configuración de PlasmaC
plasmac_xxx.hal	conexiones HAL para el panel
plasmac_xxx.py	código python para el panel
plasmac_G-Code.py	preprocesador para el archivo G-Code cargado
plasmac.tcl	conexiones HAL permanentes para el componente plasmac
pmx_test.py	panel de prueba para comunicaciones powermax
pmx485.py	código python para comunicaciones powermax
tool.tbl	tabla de herramientas para la configuración de PlasmaC
versions.html	lista completa de uddate
wizards	archivos relacionados con la biblioteca de formas
test	archivos para el panel de prueba

Después de ejecutar una nueva configuración la primera vez, se crearán los siguientes archivos en el directorio de configuración:

* Nombre de archivo*	**Función**(<nombre de máquina> = nombre de la máquina en el archivo .ini)
<machine_name>_config.cfg	ajustes de configuración para el Panel de configuración
<nombre_de_máquina>_run.cfg	valores de configuración para el Panel de ejecución
<machine_name>_material.cfg	archivo de material para parámetros de corte
<machine_name>_wizards.cfg	valores de configuración para la biblioteca de formas
plasmac_stats.var	estadísticas guardadas

NOTA: Los archivos .ini están escritos con los requisitos adicionales para estas configuraciones.

NOTA: Los archivos .cfg son texto sin formato y pueden editar con cualquier editor de texto.

4.7.10. Paneles GUI PlasmaC

PlasmaC agrega varios paneles a la GUI; algunos son paneles permanentes en pantalla y otros son pestañas trás la pestaña de vista previa.

Algunas funciones solo se usan para modos particulares y no se muestran si no se requieren.

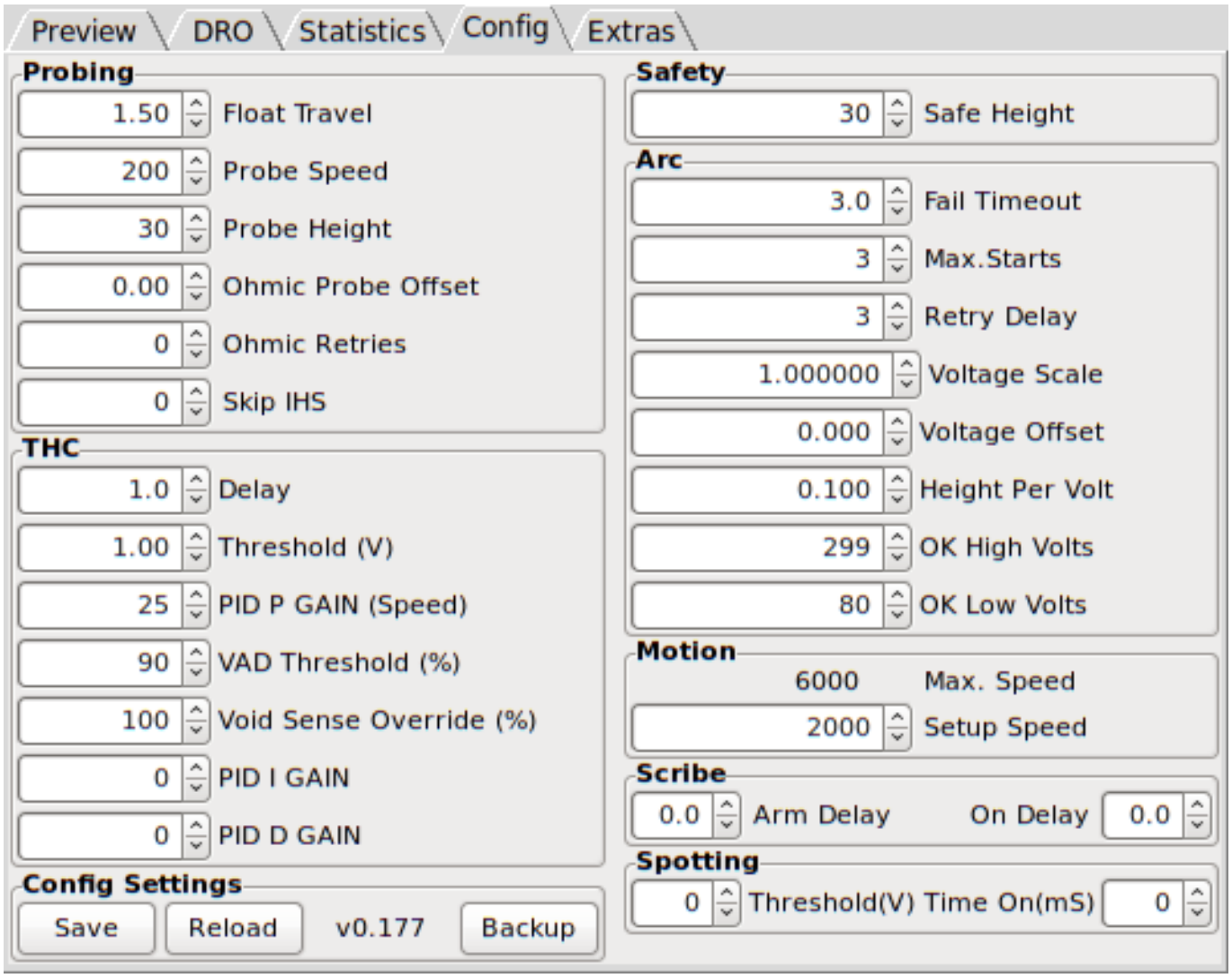
4.7.10.1. Panel de configuración

Este panel es para parámetros de configuración que se modifican con poca frecuencia.

Es posible desactivar este panel para que personal no autorizado no pueda modificar la configuración de la máquina. Esto se logra estableciendo una variable en el archivo .ini:

```
[PLASMAC] CONFIG_DISABLE = 1
```

Un pin HAL llamado **plasmac_config.config-disable** se puede configurar a cero para habilitar el panel. Este pin podría estar vinculado a un interruptor de llave o similar, por lo que solo el personal autorizado podría habilitar el Panel de configuración.



Sondeo

Nombre	Descripción
Recorrido flotante	Cantidad de recorrido en el mecanismo del interruptor flotante. Esto se puede probar con un botón de prueba de sonda, consulte botones de usuario personalizados.
Velocidad de la sonda	Velocidad a la que el eje Z sondeará después de moverse a la altura de la sonda.
Altura de la sonda	Altura por encima del límite inferior del eje Z donde comienza Velocidad de sonda.
Offset de sonda óhmica	Offset Z de la sonda óhmica.
Reintentos óhmicos	Número de veces que se debe volver a intentar una sonda óhmica, si falla, antes de recurrir al interruptor flotante.
Omitir IHS	Distancia utilizada para ver si la altura inicial del sensor (sonda) se puede omitir para el corte actual, consulte Omitir IHS.

NOTA: La altura de la sonda es la altura por encima del límite mínimo del eje Z.

THC

Nombre	Modos	Descripción
Retraso	0,1,2	Retraso desde el inicio del corte hasta que se active el THC.
Umbral	0,1,2	Variación del voltaje objetivo para el THC a la altura correcta.
PID P Gain (Velocidad)	0,1	ganancia PID P para THC, corrección de velocidad de THC.
Umbral VAD	0,1	porcentaje de velocidad de avance de corte que debe ser inferior para bloquear el THC.
Anulación de vacíos	0,1	Los valores más altos necesitan un mayor cambio de voltaje para bloquear el THC
PID I Gain	0,1	ganancia PID I para THC
PID P Gain	0,1	ganancia PID D para THC

Altura segura

Nombre	Descripción
Altura segura	Altura sobre la superficie de trabajo a la que el eje Z se retraerá en movimientos rápidos. Si se establece en cero, la altura máxima del eje Z se utilizará para la altura segura.

Arco

Nombre	Modos	Descripción
Fail Timeout	0,1,2	La cantidad de tiempo de espera desde antorcha ON hasta fallo si no se detecta el arco.
Máx. Inicia	0,1,2	Número de intentos de iniciar un arco.
Retry Delay	0,1,2	Tiempo entre un fallo de arco y otro intento de inicio de arco.
Escala de voltaje	0,1	Valor requerido para escalar la entrada de voltaje de arco para mostrar el voltaje de arco correcto.
Offset de voltaje	0,1	El valor requerido para mostrar cero voltios cuando hay una entrada de voltaje de arco cero. Para la configuración inicial, multiplique el valor de salida de voltaje de arco por -1 e ingrese eso para el offset de voltaje.
Altura por voltio	0,1,2	La distancia que la antorcha necesitaría moverse para cambiar el voltaje del arco en un voltio.
OK Voltios altos	0	Umbral de alto voltaje para Arc OK.
OK Voltios bajos	0	Umbral de bajo voltaje para Arc OK.

Movimiento

Nombre	Descripción
Máx. Velocidad	Muestra solo la velocidad máxima de la que es capaz el eje Z.
Velocidad de configuración	La velocidad del eje Z para movimientos de configuración. p.ej. Altura de perforación, altura de corte, etc.

NOTA: La velocidad de configuración no tiene efecto en la velocidad de THC, que es capaz de Max. Velocidad.

Grabado

Nombre	Descripción
Retardo de armado	Retraso para permitir que el grabador llegue a la superficie del material antes de comenzar a trazar.
On Delay	Retraso para permitir que el grabador comience antes de comenzar el movimiento.

Punteado

Nombre	Descripción
Umbral	El voltaje de arco en el que comenzará el temporizador de retardo. 0V inicia el retraso cuando se activa la señal de antorcha.
Retardo	La cantidad de tiempo que la antorcha está encendida después de alcanzar el umbral (mS).

Botones Guardar y Recargar

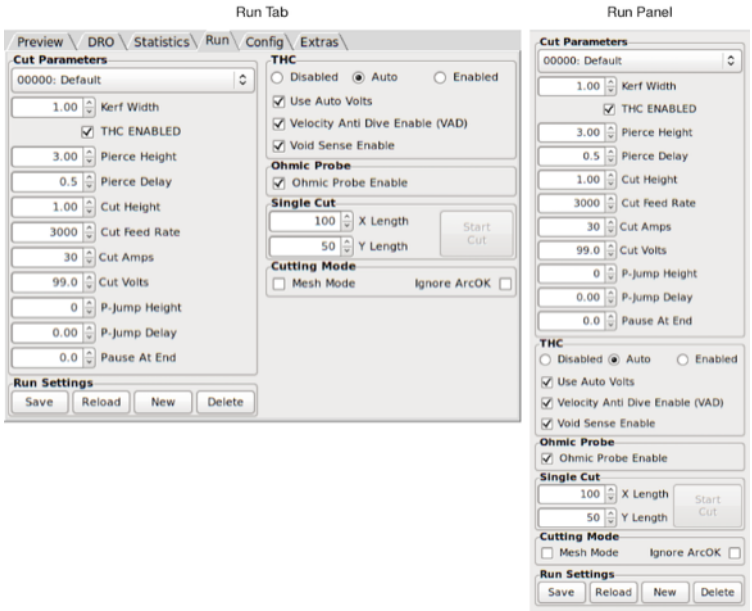
El botón **Guardar** guardará los parámetros que se muestran actualmente.

El botón **Recargar** volverá a cargar todos los parámetros.

4.7.10.2. Ejecutar panel

Este panel muestra los parámetros que están activos para el corte actual.

Hay dos formatos para este panel, una pestaña detrás de la pestaña de vista previa o un panel al costado de la GUI. Los formatos son diferentes pero la funcionalidad es idéntica.



Consulte la sección [DISPLAY] de Axis o la sección [DISPLAY] de Gmoccapy

Parámetros de corte

Nombre	Descripción
Material	Selecciona y muestra el material seleccionado actualmente, si no hay materiales en el archivo de material, solo se mostrará el material predeterminado
Ancho de corte	El ancho de corte para el material seleccionado actualmente
THC Enable	Estado de THC para el material seleccionado actualmente
Altura de perforación	La altura de perforación para el material seleccionado actualmente
Retraso de perforación	El retraso de perforación para el material seleccionado actualmente
Altura de corte	La altura de corte para el material seleccionado actualmente
Velocidad de alimentación de corte	La velocidad de alimentación de corte para el material seleccionado actualmente
Amplificadores de corte	El amperaje de corte para el material seleccionado actualmente Indicador del operador solo a menos que use comunicaciones Powermax
Voltios de corte	El voltaje de corte para el material seleccionado actualmente

Altura de salto P	La altura de salto de charco para el material seleccionado actualmente
P-Jump Delay	El retraso de salto de charco para el material seleccionado actualmente
Pausa al final	Pausa durante n segundos al final del corte con la antorcha encendida antes de apagar la antorcha y levantar la antorcha
Presión de gas	La presión de gas para el material seleccionado actualmente. Solo válido si las comunicaciones PowerMax están operativas
Modo de corte	El modo de corte para el material seleccionado actualmente. Solo válido si las comunicaciones PowerMax están operativas 1 = Normal, 2 = CPA, 3 = Gubia / Marca

THC

Nombre	Modos	Descripción
Estado	0,1,2	Deshabilitado = permanentemente apagado Auto = decidir por THC Enable y G-Code Activado = activado a menos que el Código G lo desactive
Usar voltios automáticos	0,1	Encendido = El voltaje de control del THC se toma de la tensión del arco Apagado = el voltaje de control de THC es de Voltios de corte
VAD Enable	0,1,2	On = Velocity anti-dive is enabled Apagado = La velocidad anti-inmersión está desactivada
Void Sense Enable	0,1	On = La detección de vacío está habilitada Apagado = La detección de vacío está desactivada

Sonda óhmica

Nombre	Descripción
Activación de sonda óhmica	Esto es para activar o desactivar la entrada de sonda óhmica.

NOTA: Si la sonda Ohmic está desactivada, el LED de la sonda Ohmica seguirá mostrando el estado de la entrada de la sonda.

Corte único

Nombre	Descripción
Longitud del eje X	La distancia del eje X a recorrer para un solo corte.
Longitud del eje Y	La distancia del eje Y a recorrer para un solo corte.
Iniciar corte	Presione para comenzar un solo corte.

Corte de malla

Nombre	Descripción
Activar	Activar el modo de malla para cortar metal expandido.
Ignorar OK	No requiere Arc OK para comenzar a cortar en modo de malla.

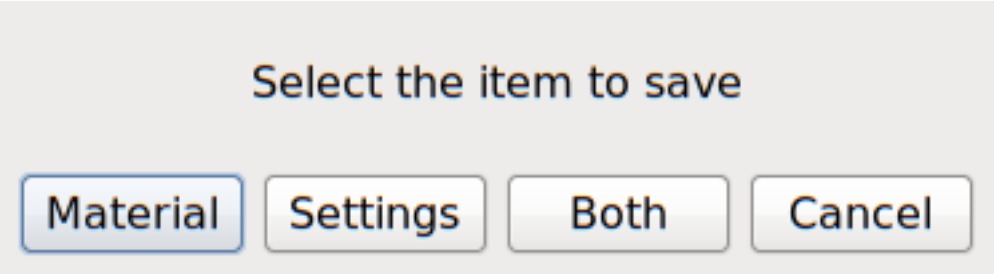
Powermax Comms

Nombre	Descripción
Habilitar	Esto es para habilitar o deshabilitar las comunicaciones a un Powermax.
Estado	Esto mostrará uno de **Conectando**, **Conectado**, **Error de comunicación** o **Código de falla**.

Para obtener más información, consulte la sección Comunicaciones Powermax.

NOTA: Este marco solo es visible si un PM_PORT está configurado en la sección [PLASMAC] del archivo .ini.

Botón Guardar



El botón **Guardar** muestra un cuadro de diálogo con las siguientes opciones:

Material	Guarde los parámetros de corte para el material que se muestra actualmente.
Configuración	Guardar toda la configuración excepto los parámetros de corte.
Ambos	Guarde los parámetros de corte y la configuración.
Cancelar	Cierre el cuadro de diálogo sin guardar.

Botón de recarga

El botón **Recargar** volverá a cargar todas las configuraciones guardadas y el archivo de material y luego volverá a mostrar el material actual. Si se habilitaron las comunicaciones Powermax, ignorará la configuración guardada y la mantendrá habilitada.

Botón Nuevo

El botón **Nuevo** permite agregar un nuevo material en el archivo de material.

Solicitará un número de material y un nombre de material. Cuando se ingresen, creará el nuevo material utilizando los Parámetros de corte del material actualmente activo y luego volverá a cargar el archivo de material con el nuevo material que se muestra.

Realice los cambios necesarios y luego guárdelos con el botón Guardar.

Botón Eliminar

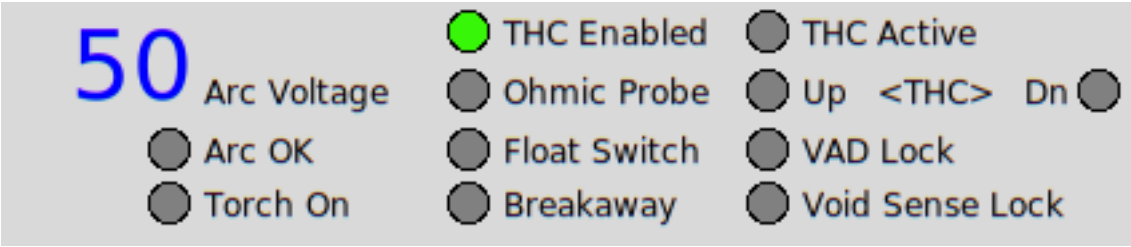
El botón **Eliminar** permite eliminar un material del archivo de material.

Solicitará un número de material y cuando se ingrese, volverá a solicitar para asegurarse de que desea eliminarlo.

Si se confirma la eliminación, eliminará el material del archivo de material y volverá a cargar los materiales con el material predeterminado cargado.

4.7.10.3. Panel Monitor

El Panel monitor se usa solo para visualización; tanto Axis como Gmoccapy tienen Paneles monitor similares.



Nombre	Modos	Descripción
Voltaje de arco	0,1	Muestra el voltaje de arco real
Arc OK	1,2	Indica el estado de la señal de arco ok
Antorcha encendida	0,1,2	La antorcha está activada
THC habilitado	0,1,2	Indica si THC está habilitado
Sonda óhmica	0,1,2	Indica que la sonda ha detectado el material
Interruptor de flotador	0,1,2	Indica que el interruptor de flotador está activado
Breakaway	0,1,2	Indica que el sensor de ruptura está activado
THC activo	0,1,2	Indica que el THC está activo actualmente
THC Up	0,1,2	Indica que el THC está causando que el eje Z se eleve
THC Down	0,1,2	Indica que el THC está causando que el eje Z baje
THC Velocity Lock	0,1,2	Indica que el THC está bloqueado debido a restricciones de velocidad
THC Void Lock	0,1	Indica que el THC está bloqueado debido a que se detecta un vacío

NOTA: El LED de la sonda Ohmica mostrará el estado de la entrada de la sonda independientemente de si la sonda Ohmica está activada o desactivada.

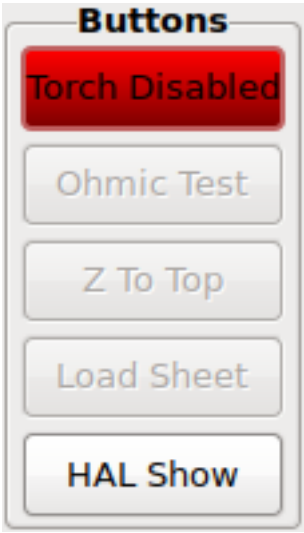
4.7.10.4. Panel de botones

El Panel de botones contiene botones útiles para el funcionamiento de la máquina.

Ambas GUI tienen un botón **Torch Enable** que es permanente, todos los demás botones son programables por el usuario en el archivo .ini. Axis tiene cinco botones de usuario y Gmoccapy tiene cuatro botones de usuario.

Diseño de botonesn de Axis:

Diseño del botones de Gmoccapy:



El botón **Activar antorcha** alterna entre Activado y Desactivado.

Debe estar habilitado para hacer cualquier corte.

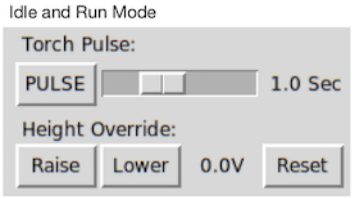
Si está deshabilitado, ejecutar un programa cargado hará que el programa realice su ciclo sin que se active la antorcha.

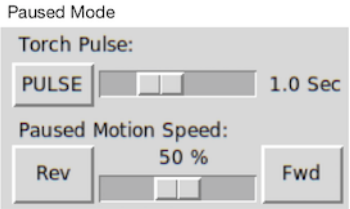
4.7.10.5. Panel de control

El Panel de control permite el control de algunas funciones, ambas GUI son similares, excepto que el Panel de control Gmoccapy está integrado en el marco que contiene alimentación y anulaciones rápidas.

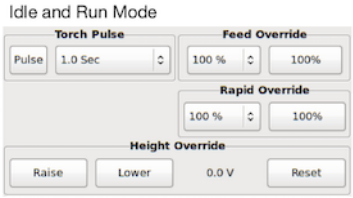
Estas funciones se habilitan y deshabilitan automáticamente según el estado de la máquina.

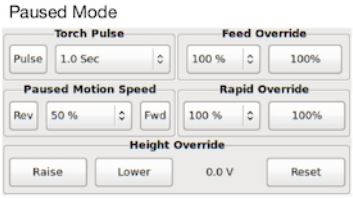
Panel de control de Axis:





Panel de control de Gmoccapy:





Antorcha Pulso

Esto enciende la antorcha durante el tiempo establecido en el control deslizante correspondiente.

Movimiento en pausa

Cuando un programa está en pausa, esto permite que el movimiento x/y siga la ruta programada en la dirección inversa o hacia adelante al porcentaje de la velocidad de alimentación actual que se muestra en el ajuste.

Cuando se invierte, el movimiento se detendrá cuando llegue a un comando husillo ON.

Al reversa, el movimiento puede continuar hasta el final del camino.

Cambio de altura

Cada vez que presione Subir o Bajar, la altura de la antorcha cambiará según el valor establecido en Altura por voltio en el marco de arco del Panel de configuración. Restablecer cancelará cualquier anulación de altura.

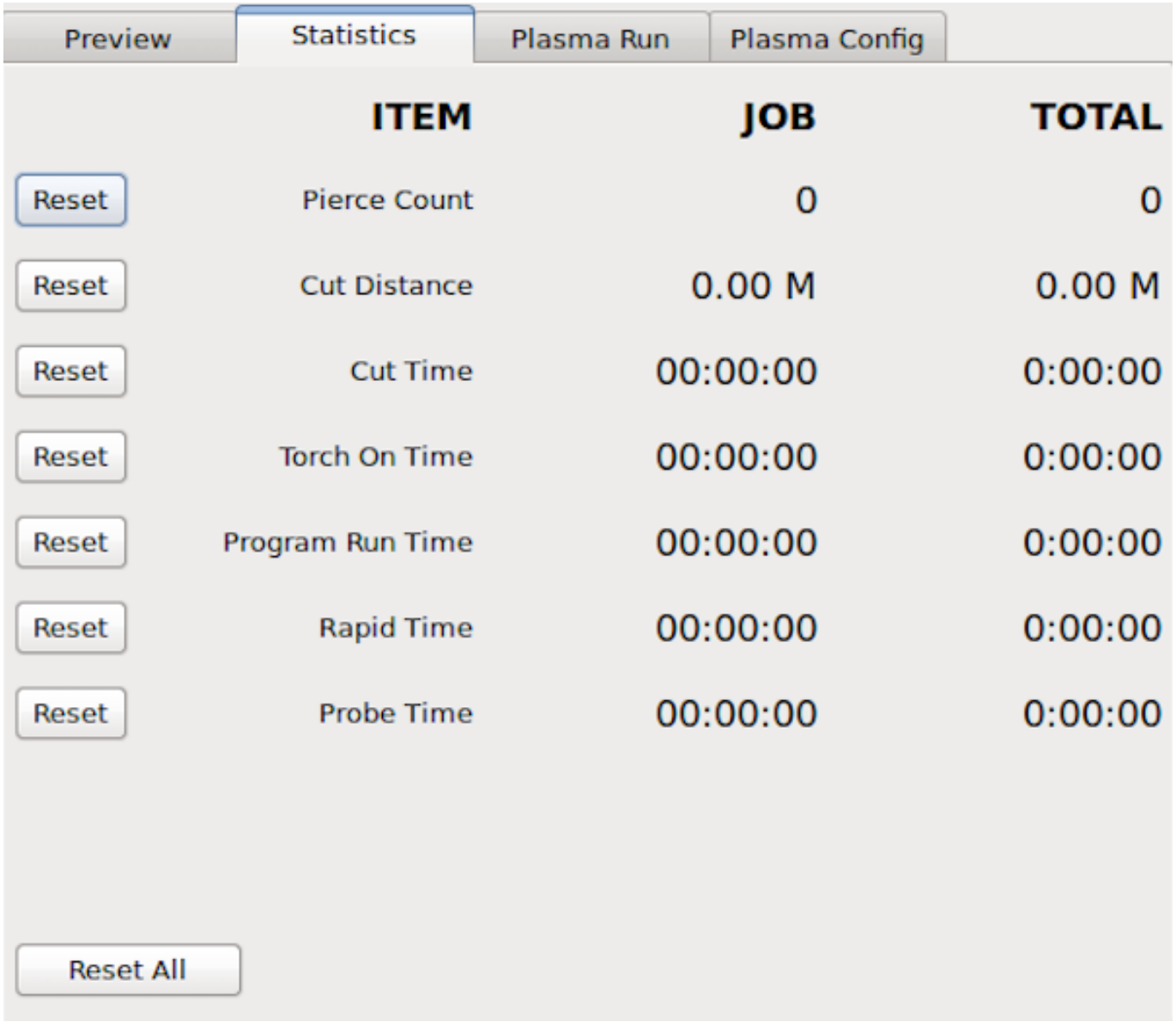
4.7.10.6. Panel de estadísticas

Esto proporciona estadísticas para permitir el seguimiento del desgaste de los consumibles y los tiempos de trabajo.

Se muestran para el trabajo actual y también se muestran como un total acumulado.

Las estadísticas de trabajo se restablecen cuando se ejecuta el programa actual.

Los valores totales se pueden restablecer individualmente o todos juntos.



4.7.10.7. Panel de extras

Esto proporciona una biblioteca de formas conversacionales con algunas formas simples que se pueden seleccionar y cargar en LinuxCNC.

También proporciona diez botones de usuario personalizados que funcionan igual que los otros botones de usuario personalizados, además de que tienen la capacidad de cargar un programa de código G y mostrar una imagen en el botón.

Vea la biblioteca de formas conversacional para el uso de formas.

Vea botones de usuario personalizados para el uso del botón.

El panel Extras se puede deshabilitar comentando las entradas apropiadas en la sección [DISPLAY] del archivo .ini. Esto no tendrá ningún efecto en el funcionamiento de la configuración de PlasmaC.

4.7.11. Usando PlasmaC

El único requisito obligatorio del Código G para una configuración de PlasmaC es **M3 $0 S1** para comenzar un corte y **M5 $0** para finalizar un corte, además, por supuesto, de códigos de movimiento estándar para moverse de principio a fin.

Si está utilizando PlasmaC sin múltiples herramientas habilitadas, entonces está permitido usar **M3 S1** en lugar de **M3 $0 S1** para comenzar un trabajo de corte y **M5** en lugar de **M5 $0**.

4.7.11.1. Coordenadas

Consulte la configuración del eje Z recomendada.

Cada vez que se inicia LinuxCNC, se requiere homing. Esto permite que LinuxCNC sepa en qué lugar del sistema universal está la antorcha y establece los límites soft a los valores especificados en el archivo .ini, evitando así que la máquina se bloquee por completo durante el uso normal.

Si su máquina no tiene interruptores home, entonces debe asegurarse de que todos los ejes estén en las coordenadas home especificadas en el archivo .ini antes del homing.

Si su máquina tiene interruptores home, se moverá a las coordenadas home especificadas cuando se le indique.

Dependiendo de su configuración, es posible que tenga un botón **Home Todo** o puede que necesite iniciar cada eje individualmente. Use el botón/botones apropiados para iniciar la máquina.

Después del homing de la antorcha, necesita saber dónde está el material, esto se logra con **Touch Off**.

Se recomienda que, la primera vez que use PlasmaC, mueva el eje Z hacia abajo hasta cerca de las tablillas y **Touch Off** el eje Z con un desplazamiento cero. Cuando termine, mueva el eje Z hasta donde estaba originalmente. Esto no debería necesitar hacerse nuevamente. De ahora en adelante, deje el eje Z en esta posición, ya que PlasmaC controlará todo el movimiento del eje Z.

Si tiene la intención de colocar su material en el mismo lugar en la mesa cada vez, entonces podría mover los ejes X e Y a la posición X0, Y0 del material y luego **Touch Off** ambos ejes con un desplazamiento cero.

Si tiene la intención de colocar el material al azar en la mesa, entonces antes de cada trabajo deberá **Touch Off** como se indica arriba o colocar el material en la ubicación anterior.

4.7.11.2. Tolerancia de ruta

Los archivos << rs274,RS274NGC_STARTUP_CODE>> proporcionados, metric_startup.ngc e imperial_startup.ngc establecen la tolerancia de ruta a 0.1 mm (0.004 ") con un comando G64. Si LinuxCNC ve un Estop en cualquier etapa, entonces la tolerancia de ruta si se configura por defecto que redondeará las esquinas, por lo que se recomienda que la tolerancia de ruta se establezca en el encabezado de cada archivo G-Code.

4.7.11.3. Pausa al final del corte

Esto hace que todo el movimiento se detenga al final del corte con la antorcha todavía encendida. Después de que haya pasado el tiempo de espera especificado, la antorcha se apagará y el eje Z se elevará.

El valor de tiempo para esta función es **Pausa al final** en los parámetros de corte del panel Ejecutar. Este valor también está en el archivo de material y puede establecerse para cada material.

Esta característica se usa para permitir que el arco alcance la posición de la antorcha para terminar completamente el corte. Por lo general, se requiere para materiales más gruesos, especialmente inoxidables.

4.7.11.4. Herramientas múltiples

Varias herramientas permiten el uso de más de una herramienta, las herramientas válidas son:

- Antorcha de plasma: para corte normal
- grabador - para grabar
- Antorcha de plasma - para puntear

Si se habilitan varias herramientas, se requiere un número de herramienta LinuxCNC en el comando **M3** para seleccionar la herramienta requerida.

- **M3 $0 S1** seleccionará la herramienta de corte por plasma.
- **M3 $1 S1** seleccionará el grabador.
- **M3 $2 S1** seleccionará la herramienta de detección de plasma.

Para habilitar la función de herramientas múltiples, debe editar la siguiente línea en <machine_name>_connections.hal.

de:

```
setp plasmac.multi-tool 0
```

a:

```
setp plasmac.multi-tool 1
```

PlasmaC leerá los parámetros de corte del Panel de ejecución y los usará cuando sea necesario.

4.7.11.5. Manejo de materiales

El manejo de materiales no tiene nada que ver con la tabla de herramientas LinuxCNC, sino que utiliza un archivo de material que se creó al mismo tiempo que la configuración.

La tabla de herramientas de LinuxCNC y los comandos de herramientas solo entran en juego si está utilizando un grabador además de la antorcha de plasma.

Hay un archivo de material con su nombre derivado de **[EMC] MACHINE** en el archivo de entrada, por lo que una máquina llamada **METRIC_PLASMAC** tendría un archivo de material llamado **metric_plasmac_material.cfg**.

No es un requisito que utilice un archivo de material; si es necesario, puede cambiar los parámetros de corte manualmente en el Panel de ejecución. Tampoco es un requisito utilizar los cambios automáticos de material, simplemente omítalos del archivo G-Code.

Los siguientes códigos pueden usarse para una configuración de PlasmaC:

- **M190 Pn** - cambia el material al número n.
- **M66 P3 L3 Q1** - espera a que se confirme el cambio de material.
- **F#<_hal[plasmac.cut-feed-rate]>** - establece la velocidad de alimentación a la velocidad de alimentación que se muestra en los parámetros de corte del panel Ejecutar.
- **M3 $0 S1** - inicia el componente PlasmaC.

Para el manejo manual de materiales, seleccione manualmente el material de la lista de materiales en el Panel de ejecución antes de ejecutar su Código G, que debe tener el siguiente código mínimo:

```
F#<_hal[plasmac.cut-feed-rate]>
M3 $0 S1
.
.
M5 $0
```

NOTA: El manejo manual de materiales lo restringe a un solo material para todo el trabajo.

Para el manejo automático de materiales, los códigos DEBEN estar en el orden mostrado. Podrías tener otros códigos entre ellos. En su código G necesita:

```
M190 Pn
M66 P3 L3 Q1
F#<_hal[plasmac.cut-feed-rate]>
M3 $0 S1
.
.
M5 $0
```

Los números de material en el archivo de materiales no necesitan ser consecutivos ni deben estar en orden numérico.

Cuando se cambia un material, solo cambia los parámetros de corte en el Panel de ejecución, LinuxCNC no sabe nada del material ni PlasmaC sabe nada sobre las herramientas de LinuxCNC. (es decir, NO hace un cambio de herramienta)

Los materiales se pueden seleccionar manualmente con el cuadro combinado Parámetros de corte o mediante MDI con M190 Pn.

Si se carga un programa de código G que contiene uno o más comandos de cambio de material, el primer material se mostrará en el Panel de ejecución mientras se carga el programa.

4.7.11.6. Materiales que se agregan y editan desde el código G

Es posible agregar nuevos materiales o editar materiales existentes mediante el uso de "comentarios mágicos" en un archivo G-Code.

Las opciones son:

Opción	Descripción
0	Crear un material predeterminado temporal Esto será descartado por una recarga manual o un reinicio de LinuxCNC
1	Agregar un nuevo material si no existe
2	Sobrescribir un material existente si existe Agregar un nuevo material si no existe

Los parámetros obligatorios son:

Nombre	Descripción
o	opción
nu	número de material (no requerido para la opción 0)
na	nombre del material (no requerido para la opción 0)
ph	altura de perforación
pd	retraso de perforación
ch	altura de corte
fr	velocidad de alimentación

Los parámetros opcionales son:

Nombre	Descripción
kw	ancho de corte
th	thc enable (0 = deshabilitar, 1 = habilitar)
ca	amplificadores de corte
cv	cortar voltaje
pe	pausa al final del retraso
gp	presión de gas (Powermax)
cm	modo de corte (Powermax)
jh	altura de salto de charco
jd	retraso de salto de charco

Las reglas de formato son:

- Todo el comentario debe estar entre paréntesis
- El comienzo del comentario mágico debe ser **(o=**
- El signo igual debe seguir inmediatamente cada parámetro sin espacio
- Los parámetros obligatorios anteriores deben estar en el comentario mágico (**nu** y **na** no son necesarios para la opción 0)
- Puede haber cualquier número y tipo de comentarios mágicos en un archivo de Código G con la excepción de la opción 0. Si hay más de una opción 0 en el archivo de Código G, entonces solo se usará el primero encontrado y los demás tener una advertencia informada.

Un ejemplo completo es:

```
(o=1, nu=2, na=5mm Mild Steel 40A, ph=3.1, pd=0.1, ch=0.75, fr=3000, kw=0.5, th=1, ca=45,  ↩
    cv=110, pe=0.1, gp=5, cm=1, jh=0, jd=0)
```

4.7.11.7. THC

El THC se puede controlar desde el marco de THC del Panel de ejecución, off = deshabilitado, on = habilitado y auto significa cumplir con la casilla de verificación THC Enable en los parámetros de corte.

El THC también se puede habilitar o deshabilitar directamente desde G-Code siempre que el THC no esté deshabilitado en el Panel de ejecución.

PlasmaC utiliza un voltaje de control que depende del estado de la casilla de verificación Usar voltios automáticos en el Panel de ejecución:

1. Si se selecciona Usar voltios automáticos, el voltaje de corte real se muestrea después de que comienza el corte y luego se usa como voltaje de control. Para permitir que el voltaje del arco se estabilice, PlasmaC espera la cantidad de tiempo que se muestra como Retraso en el marco de THC del Panel de análisis antes de tomar la muestra.
2. Si la opción Usar voltios automáticos no está marcada, el voltaje que se muestra como Voltios de corte en el cuadro Parámetros de corte del panel de ejecución se usa como voltaje de control.

El THC no se activa hasta que la velocidad alcanza el 99.9 % de **CutFeedRate**.

Código G THC

El THC puede deshabilitarse y habilitarse directamente desde el Código G, siempre que el THC no esté deshabilitado en el Panel de ejecución, configurando o restableciendo el pin **motion.digital-out-02** con los M-Codes M62-M65:

- **M62 P2** desactivará THC (sincronizado con movimiento)
- **M63 P2** habilitará THC (sincronizado con movimiento)
- **M64 P2** deshabilitará el THC (inmediatamente)
- **M65 P2** habilitará THC (inmediatamente)

Velocidad basada en THC

Si la velocidad de corte cae por debajo de un porcentaje de **CutFeedRate**, entonces el THC se bloqueará hasta que la velocidad de corte regrese al menos al 99 % de **CutFeedRate**.

Este porcentaje se muestra como % de umbral VAD en el marco de THC del Panel de configuración.

El THC basado en la velocidad evita que se cambie la altura de la antorcha cuando la velocidad se reduce para una esquina afilada o un pequeño agujero.

Hay un pin HAL disponible llamado **motion.analog-out-03** que se puede cambiar en G-Code con los comandos M67/M68. Este pin reducirá la velocidad al porcentaje especificado en el comando.

- **M67 E3 Q0** establecería la velocidad al 100 % de **CutFeedRate**.
- **M67 E3 Q40** establecería la velocidad al 40 % de **CutFeedRate**.
- **M67 E3 Q60** establecería la velocidad al 60 % de **CutFeedRate**.
- **M67 E3 Q100** establecería la velocidad al 100 % de **CutFeedRate**.

El porcentaje mínimo permitido es del 10 %, los valores por debajo de este se establecerán en el 100 %.

El porcentaje máximo permitido es 100 %, los valores superiores a este se establecerán en 100 %.

Si tiene la intención de utilizar esta función, sería prudente agregar **M68 E3 Q0** a su preámbulo y posámbulo del Código G para que comience y termine en un estado conocido.

CONSEJO: Otra forma de lograr el mismo resultado es usar **F#<_hal[plasmac.cut-feed-rate]** con un multiplicador.

```
F[#<_hal[plasmac.cut-feed-rate]*0.6]
```

IMPORTANTE: **El THC código G** y **el THC basado en la velocidad** NO PUEDEN SER USADOS si **la compensación de la cuchilla** ESTÁ EN EFECTO, SE VISUALIZARÁ UN MENSAJE DE ERROR.

ADVERTENCIA: Si la velocidad de alimentación de corte en el panel de ejecución se establece en cero, PlasmaC usará **motion.requested-speed** para los cálculos de THC, que no es una forma muy confiable de THC basada en la velocidad y no se recomienda.

NOTA: Todas las referencias a CutFeedRate significan Velocidad de avance de corte como se muestra en el Panel de ejecución.

4.7.11.8. Compensación del cortador

Para usar la compensación del cortador, deberá usar G41.1, G42.1 y G40 con el pin HAL de ancho de corte:

- **G41.1 D#<_hal[plasmac_run.kerf-width-f]>**; para la izquierda de la ruta programada
- **G42.1 D#<_hal[plasmac_run.kerf-width-f]>** a la derecha de la ruta programada
- **G40** para desactivar la compensación

IMPORTANTE: SI **La compensación del cortador** ESTÁ EN EFECTO **El código G THC**, **El THC basado en la velocidad** Y **el sobrecorte** NO PUEDEN SER USADOS, SE VISUALIZARÁ UN MENSAJE DE ERROR.

4.7.11.9. Salto de detección de altura inicial

IHS se puede omitir de dos maneras diferentes:

Si el THC está deshabilitado, omita IHS si el inicio del corte es menor a **Omitir IHS** distancia desde la última sonda exitosa.

Si el THC está habilitado, omita IHS si el inicio del corte es menor que *Omitir IHS *distancia desde el final del último corte.

Un valor de cero para **Saltar IHS** deshabilitará el salto de IHS.

Cualquier error encontrado en un corte deshabilitará la omisión de IHS para el próximo corte si estaba habilitado.

4.7.11.10. Sondeo

El sondeo puede ser con detección óhmica o con un interruptor de flotador, también es posible combinar ambos con el interruptor de flotador que proporciona una alternativa al sondeo óhmico.

Si su antorcha no admite sondeo óhmico, podría tener una sonda separada al lado de la antorcha. En este caso, extendería la sonda debajo de la antorcha. La sonda NO debe extenderse más allá de su Altura mínima de corte debajo de la antorcha y esta distancia de compensación se ingresa como **ohmic-probe-offset** en el Panel de configuración.

La configuración del sondeo se realiza en el cuadro Movimiento del Panel de configuración.

PlasmaC puede sondear a la velocidad máxima del eje Z siempre que su máquina tenga suficiente movimiento en el interruptor de flotación para absorber cualquier desbordamiento. Si su máquina es adecuada, puede establecer la Altura de la sonda cerca del mínimo del eje Z y hacer todas las pruebas a toda velocidad.

NOTA: La altura de la sonda es la altura por encima del límite mínimo del eje Z.

4.7.11.11. Tipos de corte

PlasmaC permite dos tipos de corte diferentes:

- Perforar y cortar - ejecuta el programa G-Code cargado para perforar y luego cortar.
- Solo perforación - convierte y luego ejecuta el programa G-Code cargado para hacer solo perforación.

El modo de solo perforación es útil para materiales gruesos que pueden producir suficiente escoria en la superficie del material al perforar para interferir con la antorcha durante el corte. Esto permite perforar y limpiar toda la hoja antes de cortarla.

Esto también permite que los consumibles del final de la vida útil se usen para perforar y se pueden cambiar por buenos consumibles para cortar.

Hay dos formas de habilitar esta función:

- Programe un botón de usuario personalizado para alternar entre los tipos de corte.
- Agregar una línea **#<pierce-only> = 1** en cualquier parte de un archivo G-Code para habilitar el modo Pierce Only para el archivo actual.

Si usa un botón de usuario personalizado, la GUI volverá a cargar automáticamente el archivo cuando se alterna el tipo de corte.

4.7.11.12. Corte de agujeros - Introducción

Se recomienda que los orificios que se corten tengan un diámetro no inferior a una vez y media el grosor del material a cortar.

También se recomienda que los agujeros con un diámetro de menos de 32 mm (1.25") se corten al 60 % de la velocidad de alimentación utilizada para los cortes de perfil. Esto también debería bloquear el THC debido a restricciones de velocidad.

PlasmaC puede utilizar comandos de código G generalmente establecidos por un postprocesador CAM (PP) para ayudar en el corte de agujeros o si no tiene un PP o su PP no admite estos métodos, entonces PlasmaC puede adaptar automáticamente el código G para adaptarse. Este modo automático está deshabilitado de forma predeterminada.

Hay tres métodos disponibles para mejorar la calidad de los agujeros pequeños.

1. Reducción de velocidad - reduzca la velocidad a aproximadamente el 60 % de **CutFeedRate**.
2. Permanencia en el arco - mantenga la antorcha encendida durante un corto tiempo al final del hoyo mientras se detiene el movimiento para permitir que el arco se recupere.
3. Sobrecorte - apague la antorcha al final del agujero y luego continúe por el camino.

NOTA: Si tanto **Arc Dwell** como **Overcut** están activos al mismo tiempo, entonces **Overcut** tendrá prioridad.

IMPORTANTE: **Sobrecorte** NO SE PUEDE UTILIZAR SI LA COMPENSACIÓN DE CORTE ESTÁ EN EFECTO, SE VISUALIZARÁ UN MENSAJE DE ERROR.

4.7.11.13. Corte de agujeros - Comandos de código G

Sus comandos de código G se configuran mediante un postprocesador CAM (PP) o mediante codificación manual.

Reducción de velocidad

Si un agujero requiere una velocidad reducida, establezca la velocidad con **M67 E3 Q60**, que establecería la velocidad al 60 % de **CutFeedRate**.

Vea la sección TCH Basado en Velocidad.

Código de muestra:

```
G21 (métrico)
G64 P0.005
M52 P1 (permitir movimiento en pausa)
F#<_hal[plasmac.cut-feed-rate]> (velocidad de alimentación de los parámetros de corte)
G0 X10 Y10
M3 $0 S1 (corte inicial)
G1 X0
M67 E3 Q60 (reducir la velocidad de alimentación al 60%)
G3 I10 (el agujero)
M67 E3 Q0 (restaurar la velocidad de alimentación al 100%)
M5 $0 (corte final)
G0 X0 Y0
M2 (trabajo final)
```

Arc Dwell

Mantenga el movimiento durante un tiempo configurando **Pausa al final** en el cuadro Parámetros de corte del Panel de ejecución

Código de muestra:

```
G21 (métrico)
G64 P0.005
M52 P1 (permitir movimiento en pausa)
F#<_hal[plasmac.cut-feed-rate]> (velocidad de alimentación de los parámetros de corte)
G0 X10 Y10
M3 $0 S1 (corte inicial)
```

```
G1 X0
M67 E3 Q60 (reducir la velocidad de alimentación al 60%)
G3 I10 (el agujero)
M67 E3 Q0 (restaurar la velocidad de alimentación al 100%)
M5 $0 (corte final)
G0 X0 Y0
M2 (trabajo final)
```

Sobrecorte

La antorcha se puede apagar al final del agujero configurando el pin **motion.digital-out-03** con los códigos M **M62** o **M64 *. Después de apagar la antorcha, es necesario permitir que la antorcha se encienda nuevamente antes de comenzar el siguiente corte restableciendo el pin *motion.digital-out-03** con los códigos M **M63** o **M65**, esto El analizador de código G de PlasmaC puede hacerlo automáticamente si alcanza un comando M5 sin ver un **M63 P3** o **M65 P3**.

Después de apagar la antorcha, se seguirá la trayectoria del orificio durante una longitud predeterminada de 4 mm (0.0157"). Esta distancia se puede especificar agregando #<oclength> = n al archivo G-Code.

- **M62 P3** apagará la antorcha (sincronizada con el movimiento)
- **M63 P3** permitirá encender la antorcha (sincronizada con el movimiento)
- **M64 P3** apagará la antorcha (inmediatamente)
- **M65 P3** permitirá encender la antorcha (inmediatamente)

Código de muestra:

```
G21 (métrico)
G64 P0.005
M52 P1 (permitir movimiento en pausa)
F#<_hal[plasmac.cut-feed-rate]> (velocidad de alimentación de los parámetros de corte)
G0 X10 Y10
M3 $0 S1 (corte inicial)
G1 X0
M67 E3 Q60 (reducir la velocidad de alimentación al 60%)
G3 I10 (el agujero)
M62 P3 (apague la antorcha)
G3 X0.8 Y6.081 I10 (movimiento continuo durante 4 mm)
M63 P3 (permite encender la antorcha)
M67 E3 Q0 (restaurar la velocidad de alimentación al 100%)
M5 $0 (corte final)
G0 X0 Y0
M2 (trabajo final)
```

4.7.11.14. Corte de agujeros - Automático

PlasmaC tiene la capacidad de modificar automáticamente el Código G para reducir la velocidad y/o aplicar *Sobrecorte *.

El tamaño de orificio predeterminado para la detección de orificios PlasmaC es 32 mm (1.26"). Es posible cambiar este valor con los siguientes comandos:

- #<m_diameter> = nn - Para establecer un diámetro en medidas métricas
- #<i_diameter> = nn - Para establecer un diámetro en medidas imperiales

La detección de agujeros PlasmaC se habilita / deshabilita configurando el siguiente parámetro del código G. Está deshabilitado por defecto.

- #<holes> = 1 - Hace que PlasmaC reduzca la velocidad de los agujeros de menos de 32 mm (1.25") al 60% de **CutFeedRate**.

- #<holes> = 2 - Además de la reducción de velocidad anterior, también apaga la antorcha al final y sigue la trayectoria del agujero.

Arc Dwell

Mantenga el movimiento durante un tiempo configurando **Pausa al final** en el cuadro Parámetros de corte del Panel de ejecución

- #<holes> = 1 para permitir la reducción de velocidad.

Código de muestra:

```
G21 (métrico)
G64 P0.005
M52 P1 (permitir movimiento en pausa)
F#<_hal[plasmac.cut-feed-rate]> (velocidad de alimentación de los parámetros de corte)
#<holes> = 1 (reducción de velocidad para agujeros)
G0 X10 Y10
M3 $0 S1 (corte inicial)
G1 X0
G3 I10 (el agujero)
M5 $0 (corte final)
G0 X0 Y0
M2 (trabajo final)
```

Sobrecorte

- #<holes> = 2 para permitir la reducción de velocidad y el sobrecorte.
- #<oclength> = n para especificar una longitud de sobrecorte diferente, si no se especifica este parámetro, el valor predeterminado es 4 mm (0.0157").

Código de muestra:

```
G21 (métrico)
G64 P0.005
M52 P1 (permitir movimiento en pausa)
F#<_hal[plasmac.cut-feed-rate]> (velocidad de alimentación de los parámetros de corte)
#<holes> = 2 (sobrecorte para agujeros)
#<oclength> = 6.5 (opcional, longitud de sobrecorte de 6.5 mm)
G0 X10 Y10
M3 $0 S1 (corte inicial)
G1 X0
G3 I10 (el agujero)
M5 $0 (corte final)
G0 X0 Y0
M2 (trabajo final)
```

NOTA: Está bien tener comandos múltiples y mixtos #<*_diámetro> en un archivo G-Code.
Está bien tener múltiples y mixtos comandos #<holes> en un archivo G-Code.

4.7.11.15. Corte individual

Un corte único es un movimiento de corte unidireccional que se usa a menudo para cortar una hoja en pedazos más pequeños antes de ejecutar un programa de código G.

La máquina debe estar orientada antes de comenzar un solo corte.

Un solo corte comenzará desde la posición X / Y actual de la máquina.

Corte simple automático

Este es el método preferido y los parámetros para este método se ingresan en el Panel de ejecución.

1. Avance a la posición de inicio X / Y requerida.
2. Establezca la velocidad de avance de corte requerida en los parámetros de corte del panel Ejecutar.
3. Ingrese la longitud del corte a lo largo de los ejes X y / o Y.
4. Presione el botón **Iniciar corte** y el corte comenzará.

Corte simple con colgante

Si tiene un colgante que puede iniciar y detener el husillo, además de mover los ejes X e Y, puede realizar manualmente un solo corte.

1. Avance a la posición de inicio X / Y requerida.
2. Establezca la velocidad de alimentación requerida con el control deslizante Jog Speed.
3. Comience el procedimiento iniciando el husillo.
4. Después de sondear la antorcha se disparará.
5. Cuando se recibe Arc OK, la máquina se puede mover a lo largo de la línea de corte con los botones de desplazamiento.
6. Cuando se complete el corte, detenga el husillo.
7. La antorcha se apagará y el eje Z volverá a la posición inicial.

[underline]*Corte simple manual en Axis *

1. Avance a la posición de inicio X / Y requerida.
2. Establezca la velocidad de alimentación requerida con el control deslizante Jog Speed.
3. Inicie el procedimiento presionando *F9 *.
4. Después de sondear la antorcha se disparará.
5. Cuando se recibe Arc OK, la máquina se puede trotar a lo largo de la línea de corte con las teclas de desplazamiento.
6. Cuando el corte esté completo, presione **F9** o *Esc *.
7. La antorcha se apagará y el eje Z volverá a la posición inicial.

[underline]*Corte simple manual en Gmoccapy *

Si desea utilizar las teclas de desplazamiento del teclado, habilite los métodos abreviados de teclado en la pestaña Hardware de la página Configuración; de lo contrario, utilice los botones de desplazamiento de la GUI.

1. Avance a la posición de inicio X / Y requerida.
2. Establezca la velocidad de alimentación requerida con el control deslizante Jog Speed.
3. Ingrese al modo MDI.
4. Inicie el procedimiento ingresando *M3 S1 *.
5. Ingrese al modo manual.
6. Después de sondear la antorcha se disparará.
7. Cuando se recibe Arc OK, la máquina se puede trotar a lo largo de la línea de corte utilizando los botones de avance (o las teclas de avance si están habilitadas).
8. Cuando el corte esté completo, presione **Esc**.
9. La antorcha se apagará y el eje Z volverá a la posición inicial.

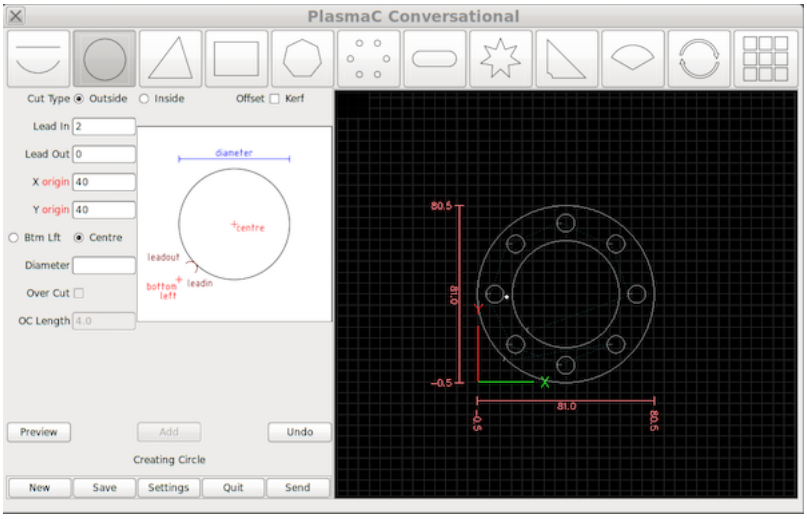
4.7.11.16. Modo de malla (metal expandido)

El metal expandido se puede cortar siempre que su máquina tenga una antorcha de arco piloto y sea capaz de un modo de arco piloto constante.

El modo de malla deshabilita el control de altura de la antorcha y también ignora una señal perdida de Arc OK. Para habilitar este modo, marque el botón de verificación Modo de malla en el Panel de ejecución.

También es posible iniciar un corte sin recibir la señal Arc OK marcando el botón Ignorar OK en el panel Ejecutar. Esto no tiene ningún efecto si el Modo de malla no está seleccionado.

4.7.12. Biblioteca de formas conversacionales



La biblioteca de formas conversacionales consta de varias formas y funciones básicas y está disponible en la pestaña **Extras** en la interfaz gráfica de usuario. No está destinado a ser un reemplazo CAD / CAM y existen limitaciones para lo que se puede lograr.

Las entradas en blanco en los cuadros de diálogo de forma utilizarán la configuración actual en ese momento. p.ej. Si **X start** estaba en blanco, se usaría la posición actual del eje X.

Todas las entradas y salidas son arcos, excepto:

- Pequeños agujeros
- Estrellas

El orden de corte es el mismo en que se creó la forma.

Si presiona **Retorno** en el teclado mientras edita los parámetros, se mostrará automáticamente la vista previa de la forma si hay suficientes parámetros ingresados para crear la forma. Hacer clic en las casillas de verificación hará lo mismo.

El botón **Nuevo** eliminará el archivo G-Code actual y cargará un archivo G-Code en blanco.

El botón **Guardar** abre un cuadro de diálogo que permite guardar la forma actual como un archivo G-Code.

El botón **Configuración** permite cambiar la configuración global.

El botón **Finalizar** cerrará el Panel de conversación y cargará la forma actual en LinuxCNC. Si no se agregó la última edición, se perderá.

El botón **Vista previa** mostrará una vista previa de la forma actual siempre que la información requerida esté presente.

El botón **Continuar** es solo para líneas y arcos y permite agregar otro segmento al segmento / segmentos actuales.

El botón **Agregar** almacenará la forma actual tal como está

El botón **Deshacer** volverá al estado almacenado anterior. Si no ha habido modificaciones desde la última tienda, se mantendrá en el estado actual.

Si hay un archivo de código G cargado en LinuxCNC cuando se abre el Panel de conversación, ese código se importará a la conversación como la primera forma del trabajo. Si no se requiere este código, se puede eliminar con el botón Nuevo.

NOTA: Todas las distancias están en unidades de máquina con respecto al Sistema de coordenadas de usuario actual y todos los ángulos están en grados.

4.7.12.1. Configuración

La configuración global de la biblioteca de formas se puede configurar presionando **Configuración** en el Panel de conversación. Esto mostrará todos los parámetros de configuración disponibles:

- Preámbulo
- Postambulo
- Origen (uno de centro o abajo a la izquierda)
- Longitud de entrada
- Longitud de salida
- Diámetro de agujero pequeño
- Velocidad de agujero pequeño
- Tamaño de ventana

Cualquier círculo interno que tenga un diámetro menor que **Diámetro de agujero pequeño** se clasifica como un agujero pequeño. Tendrá una entrada recta con una longitud menor del radio del orificio y la longitud de entrada especificada. También tendrá su velocidad de avance establecida en **Velocidad de agujero pequeño**.

Al presionar el botón **Guardar** se guardarán todas las configuraciones como se muestran y cambiará el tamaño de la ventana al tamaño que se muestra.

4.7.12.2. Forma única

1. Seleccione la forma que desea crear que mostrará los parámetros disponibles.
2. Ingrese los valores apropiados y presione **Vista previa** para mostrar la forma.
3. Si la forma no es correcta, edite los valores y presione **Vista previa** y se mostrará la nueva forma. Repita hasta que esté satisfecho con la forma.

4.7.12.3. Líneas y arcos

Las líneas y los arcos tienen una opción adicional, ya que se pueden unir para crear una forma compleja.

Hay dos tipos de línea y tres tipos de arco disponibles.

1. Línea desde el punto inicial y el punto final
2. Línea desde el punto de inicio, longitud y ángulo.
3. Arco desde el punto inicial, punto intermedio y punto final
4. Arco desde el punto inicial, el punto final y el radio
5. Arco desde el punto de inicio, longitud, ángulo y radio

Para usar líneas y arcos:

1. Seleccionar líneas y arcos
2. Seleccione el tipo de línea para crear
3. Ingrese los parámetros deseados
4. Vista previa de la forma
5. Si está satisfecho con la forma, presione Continuar
6. Cambie el tipo de línea si es necesario y continúe este procedimiento hasta que se complete la forma.

Si desea crear una forma cerrada, deberá crear cualquier leadin requerido como primer segmento de la forma. Si se requiere una salida, deberá ser el último segmento de la forma.

NOTA: En esta etapa, no hay una opción automática para una entrada / salida si la forma está cerrada.
Se pretende que esto esté disponible en el futuro, así como la compensación de ancho de corte.

4.7.12.4. Grupo de formas

Se pueden agregar varias formas para crear un grupo complejo.

El orden de corte del grupo está determinado por el orden en que las formas individuales se agregan al grupo.

Una vez que se agrega una forma al grupo, no se puede editar ni eliminar.

Los grupos no pueden editarse, solo agregarse a.

1. Cree la primera forma como en *Forma única *.
2. Presione **Agregar** y la forma se agregará al grupo.
3. Si desea agregar otra versión de la misma forma, edite los parámetros requeridos y presione **Agregar** cuando esté satisfecho con la forma.
4. Si desea agregar una forma diferente, seleccione esa forma y cree como en *Forma única *.
5. Repita hasta que haya agregado todas las formas necesarias para completar el grupo.

4.7.12.5. Rotar

La función de rotación rotará el código actual que se muestra en la pestaña Vista previa, puede ser una sola forma, un grupo o un archivo G-Code importado.

Las matrices creadas con Conversational pueden rotarse, pero las matrices importadas no pueden rotarse.

La rotación no funcionará en el código G que se calcula matemáticamente, los valores deben especificarse.

1. Cree una forma, un grupo o use el archivo G-Code importado.
2. Presione **Rotar** para abrir el diálogo de rotación.
3. Ingrese los valores apropiados en el diálogo de rotación y presione **Vista previa** para mostrar el código girado.
4. Si la rotación no es correcta, edite los valores y presione **Vista previa** y se mostrará el nuevo código girado. Repita hasta que esté satisfecho con el código girado.
5. Presione **Agregar** para completar el procedimiento.

4.7.12.6. Matriz

Se puede crear una matriz a partir de una forma, un grupo o un archivo G-Code importado.

Las matrices creadas con Conversacional pueden estar agrupadas, pero las matrices importadas no pueden estar agrupadas.

El orden de corte de una matriz es de izquierda a derecha, comenzando en la fila inferior y terminando en la fila superior.

1. Cree una forma, un grupo o use el archivo G-Code importado.
2. Presione **Array** para abrir el cuadro de diálogo de matriz.
3. Ingrese los valores apropiados en el cuadro de diálogo de la matriz y presione **Vista previa** para mostrar la matriz.
4. Si la matriz no es correcta, edite los valores y presione **Vista previa** y se mostrará la nueva matriz. Repita hasta que esté satisfecho con la matriz.
5. Presione **Agregar** para completar el procedimiento.

4.7.12.7. Guardar un trabajo

El trabajo actual que se muestra en el Panel de vista previa se puede guardar en cualquier momento utilizando el botón inferior **Guardar**.

4.7.12.8. Ejecutar un trabajo

Para ejecutar un trabajo, presione el botón **Finalizar** que cerrará el Panel de conversación y volverá a LinuxCNC.

4.7.13. Grabador

PlasmaC puede operar un escriba además de la antorcha de plasma, esto está deshabilitado de manera predeterminada para no afectar las configuraciones existentes.

El uso de un escriba requiere el uso de la tabla de herramientas LinuxCNC, la Herramienta 0 se asigna a la antorcha de plasma y la Herramienta 1 se asigna al escriba. Los desplazamientos de los ejes X e Y de la antorcha de plasma deben ingresarse en la tabla de herramientas LinuxCNC. Las compensaciones de la antorcha de plasma siempre serán cero. Las herramientas se seleccionan mediante el comando **Tn M6** y luego se requiere un **G43 H0** para aplicar los desplazamientos. PlasmaC se inicia con un comando **M3 $n S1**. Para **n**, use 0 para corte por plasma o 1 para trazar.

Hay dos pines de salida HAL utilizados para operar el trazo, uno para armar el trazo que mueve el trazo a la superficie del material, luego después de una retraso y el otro para iniciar el trazo, luego después de un retraso comenzará el movimiento.

El uso de PlasmaC después de habilitar el escriba requiere seleccionar la antorcha o el escriba en cada archivo G-Code como una herramienta LinuxCNC.

Para habilitar la escritura, vea herramientas múltiples.

Si aún no ha asignado los pines HAL para el escriba en el Configurador, deberá hacerlo invocando el modo reconfigurar del Configurador.

El escriba ahora está activo y la acción de armado se puede probar emitiendo un comando **M3 $1 S1** en la entrada MDI. Esto hará que el escriba se arme pero no lo iniciará.

El siguiente paso es abrir el editor de tablas de herramientas en la GUI e ingresar los desplazamientos de los ejes X e Y. Estas son las distancias en X e Y desde el centro de la boquilla de la antorcha. En la mayoría de los casos, si está parado en la parte delantera de la máquina si el escriba está a la derecha de la antorcha, sería un valor positivo y si el escriba está en la parte trasera de la antorcha, también sería un valor positivo. Las compensaciones para la antorcha siempre deben ser cero.

El paso final es establecer la demoras de escritura requerida:

1. **Arm Delay** permite que el escriba descienda a la superficie del material.
2. **On Delay** permite que el escriba comience antes de que comience el movimiento.

Guarde los parámetros de configuración.

El material se puede alinear con la antorcha como anteriormente y se pueden aplicar los desplazamientos para que ambas herramientas funcionen en las coordenadas correctas.

Para usar el grabador con G-Code:

```
...
M52 P1 (movimiento en pausa activado)
F#<_hal[plasmac.cut-feed-rate]>
T1 M6 (seleccione escriba)
G43 H0 (aplicar compensaciones)
M3 $1 S1 (comience plasmac con escriba)
.
.
T0 M6 (seleccione antorcha)
G43 H0 (aplicar compensaciones)
G0 X0 Y0 (posición de estacionamiento)
M5 $1 (final)
```

Es una buena idea volver a la antorcha al final del programa antes de su último movimiento de estacionamiento rápido para que su máquina esté siempre en el mismo estado en reposo.

Puede cambiar entre la antorcha y el escriba cualquier número de veces durante un programa utilizando el código anterior y emitiendo **M3 $0 S1** para iniciar PlasmaC con la antorcha.

La emisión de M3 S1 (sin $n) hará que la máquina se comporte como si se hubiera emitido un **M3 $0 S1** y la emisión de M5 (sin $n) hará que la máquina se comporte como si se hubiera emitido un *M5 $0 *, entonces la máquina intentará cortar la antorcha. Esto es para proporcionar compatibilidad con archivos G-Code anteriores.

ADVERTENCIA: Si tiene un cambio de herramienta manual existente en el archivo .hal de su máquina, entonces PlasmaC lo convertirá en un cambio de herramienta automático.

4.7.14. Punteo

Para lograr el punteo antes de perforar, etc., PlasmaC puede pulsar la antorcha durante un corto período de tiempo para marcar el punto de perforación.

Esto se hace configurando primero el umbral de voltaje de arco en el Panel de configuración. Si el voltaje de umbral se establece en cero, el temporizador de retraso comenzará desde que se envió la señal de antorcha encendida. Si el voltaje umbral se establece por encima de cero, el temporizador de retardo comenzará cuando el voltaje de arco alcance el voltaje umbral.

Luego configure el tiempo de retraso en el Panel de configuración. Cuando se completa el retraso, la antorcha se apagará. Los tiempos son ajustables de 0 a 9999 milisegundos.

La antorcha se enciende en G-Code con el comando **M3 $2 S1** que selecciona la antorcha de plasma como herramienta de detección.

Para habilitar la detección, vea herramientas múltiples.

LinuxCNC requiere algo de movimiento entre un comando **M3** y un comando *M5 *, por lo que se programa un movimiento mínimo a alta velocidad.

Un ejemplo de código G es:

```
G21 (métrico)
F99999 (alta velocidad de alimentación)
.
.
G0 X10 Y10
M3 $2 S1 (punteo)
G91 (modo de distancia relativa)
G1 X0.000001
G90 (modo de distancia absoluta)
M5 $2 (punteo)
.
.
G0 X0 Y0
G90
M2
```

NOTA: La **velocidad de alimentación alta** es para garantizar que el movimiento esté a la velocidad de alimentación más alta de la máquina.

IMPORTANTE: ALGUNOS CORTADORES DE PLASMA NO SERÁN ADECUADOS PARA ESTA CARACTERÍSTICA. SE RECOMIENDA QUE REALICE ALGUNOS PUNTEOS DE PRUEBA PARA ASEGURARSE DE QUE SU CORTADORA DE PLASMA PUEDA UTILIZAR ESTA CARACTERÍSTICA.

4.7.15. Mensajes de error

Hay una serie de mensajes de error impresos por PlasmaC. Probablemente podrían dividirse en dos grupos, **Crítico** y *Advertencia *.

4.7.15.1. Errores críticos

Los errores críticos harán que el programa haga una pausa, el operador debe borrar la causa del error.

Si el error fue durante el corte, entonces se permite el movimiento hacia adelante o hacia atrás para permitir que la máquina se posicione para reiniciar el corte.

Cuando se borra el error, se puede reanudar el programa.

Estos errores indican que el sensor correspondiente se activó durante el corte.

- **programa activado interruptor de ruptura está en pausa**
- **el programa activado del interruptor de flotación está en pausa**
- **el programa activado de la sonda óhmica está en pausa**
- **el programa válido de pérdida de arco está en pausa**

Estos errores indican que el sensor correspondiente se activó antes de comenzar el sondeo.

- **la sonda óhmica detectada antes de pasar al programa de altura de la sonda está en pausa**
- **interruptor de flotador detectado antes de pasar al programa de altura de la sonda está en pausa**
- **interruptor de ruptura detectado antes de pasar al programa de altura de la sonda está en pausa**

El eje Z alcanzó el límite inferior antes de que se detectara la pieza de trabajo.

- **límite inferior alcanzado mientras el programa de sondeo está en pausa**

La pieza de trabajo es demasiado alta para cualquier extracción rápida segura.

- **el material demasiado alto para un programa de recorrido seguro está en pausa**

Uno de estos valores en Parámetros de corte en el Panel de ejecución no es válido.

- **altura de perforación no válida o altura de corte no válida o voltios de corte no válidos**

No se detectó ningún arco después de intentar iniciar el número de veces indicado por Intentos de inicio de arco en el Panel de configuración.

- **no se detecta arco después de que <n> d el programa de intentos de inicio está en pausa**

El THC ha provocado que se alcance el límite inferior durante el corte.

- **límite inferior alcanzado mientras el programa THC que se mueve hacia abajo está en pausa**

El THC ha provocado que se alcance el límite superior durante el corte.

- Se alcanzó el límite superior mientras el programa THC que se mueve hacia arriba está en pausa*

4.7.15.2. Errores de advertencia

Los errores de advertencia no tienen efecto en un programa en ejecución y son solo informativos.

Estos errores indican que el sensor correspondiente se activó antes de que comenzara una prueba de sonda.

- **sonda óhmica detectada antes de pasar a la altura de la sonda**
- **interruptor flotante detectado antes de pasar a la altura de la sonda**
- **interruptor de ruptura detectado antes de pasar a la altura de la sonda**

Esto indica que se perdió el contacto de la sonda antes de sondear hasta el punto cero.

- **error de disparo de la sonda al sondear**

Esto indica que se alcanzó el límite inferior durante una prueba de sonda.

- **límite inferior alcanzado durante la prueba de la sonda**

Esto indica que la altura segura se ha reducido debido a que el THC eleva el eje Z durante el corte.

- **la altura transversal segura se ha reducido**

4.7.16. Archivo INI

PlasmaC requiere algunas variables de archivo .ini específicas de la siguiente manera:

4.7.16.1. Común

Sección **[PLASMAC]**

```
MODE                = 0 (use voltaje de arco externo para Arc Voltage)
                        (use voltaje de arco externo para Arc OK)
                    = 1 (use voltaje de arco externo para Arc Voltage)
                        (use un arc ok externo para Arc OK)
                    = 2 (Use arco externo ok para Arc OK)
                        (use arriba/abajo externo para THC)

CONFIG_DISABLE      = 0 (0 = activar o 1 = desactivar el panel de configuración de PlasmaC)
PAUSED-MOTION-SPEED = n (multiplica cut-feed-rate por este valor para la velocidad de  ↩
    movimiento en pausa)
TORCH-PULSE-TIME    = n (tiempo antorcha on cuando se solicita pulso manual)
BUTTON_n_NAME       = <NAME> (el nombre de un botón de usuario personalizado)
BUTTON_n_CODE       = <CODE> (el código ejecutado por un botón de usuario personalizado)
BUTTON_n_IMAGE      = <IMAGE> (la imagen mostrada por los botones 10 ~ 19)
```

Sección **[FILTER]**

```
PROGRAM_EXTENSION  = .ngc (filtrar archivos gcode)
ngc                = ./plasmac/plasmac_gcode.py
nc                 = ./plasmac/plasmac_gcode.py
tap                = ./plasmac/plasmac_gcode.py
```

Sección **[RS274NGC]**

```
RS274NGC_STARTUP_CODE   = o<metric_startup> call (código G de inicio de máquina)
SUBROUTINE_PATH         = ./:./plasmac:../../nc_files/subroutines (./ debe estar en esta  ↩
    ruta)
FEATURES                = 12 (para leer variables .ini y HAL)
USER_M_PATH             = ./:./plasmac (para cambio de material M190)
```

IMPORTANTE: VER TOLERANCIA DE RUTA

Sección **[HAL]**

```
TWOPASS = activado (necesario para múltiples archivos .hal)
HALFILE = <machine_name>.hal (el archivo .hal de su máquina base)
HALFILE = plasmac.tcl (el archivo .hal estándar de PlasmaC)
HALFILE = <machine_name>_connections.hal (conexiones PlasmaC a la máquina)
HALFILE = HALUI = halui (requerido)
```

El archivo <machine_name>.hal tiene **num_spindles=[TRAJ]SPINDLES** adjuntos al final de la línea **loadrt motmod** para permitir la adición de un grabador si es necesario.

NOTA: Puede colocar comandos HAL personalizados en <machine_name>_connections.hal ya que este archivo no se sobrescribe con una actualización.

Sección **[TRAJ]**

```
SPINDLES = 3
```

Sección **[AXIS_X]**

```
MAX_VELOCITY     = duplicar el valor en la articulación correspondiente
MAX_ACCELERATION = duplicar el valor en la articulación correspondiente
OFFSET_AV_RATIO  = 0.5
```

Sección **[AXIS_Y]**

```
MAX_VELOCITY     = duplicar el valor en la articulación correspondiente
MAX_ACCELERATION = duplicar el valor en la articulación correspondiente
OFFSET_AV_RATIO  = 0.5
```

Sección **[AXIS_Z]**

```
MIN_LIMIT        = la parte superior de los listones o justo debajo
MAX_VELOCITY     = duplicar el valor en la articulación correspondiente
MAX_ACCELERATION = duplicar el valor en la articulación correspondiente
OFFSET_AV_RATIO  = 0.5
```

NOTA: PlasmaC utiliza la función de Offsets externos para todo el movimiento del eje Z, y para mover los ejes X y/o Y para un cambio de consumible mientras está en pausa; para obtener más información sobre esta función, lea Offsets Externos en la documentación de Linuxcnc.

4.7.16.2. Especificos de GUI Axis

Sección **[PLASMAC]**

```
FONT = sans 10 (los tamaños de fuente válidos son de 9 a 15 inclusive)
THEME = Clearlooks (cualquier tema instalado, solo para las pestañas de plasmac)
WINDOW_SIZE = 0 (0 = tamaño mínimo para adaptarse a la fuente, 1 = maximizado, ancho x alto ↩
     = tamaño personalizado)
AXIS_ORIENT = portrait
```

Los valores predeterminados para lo anterior si no se especifican son:

```
FONT = sin 10
THEME = tema actual del sistema
WINDOW_SIZE = tamaño mínimo para adaptarse a la fuente
AXIS_ORIENT = landscape
```

Cambiar el tamaño FONT cambia el tamaño mínimo de la ventana. PlasmaC requiere un tamaño de ventana más grande que la ventana estándar de Axis. El tamaño de la ventana varía dependiendo de si tiene el Panel de ejecución como una pestaña o un panel en el lado derecho. Si usa el modo vertical, la altura de la ventana variará dependiendo de si tiene o no un eje giratorio. Si la ventana no cabe completamente en la pantalla, deberá reducir el tamaño de la fuente.

El tamaño de la ventana personalizada se especifica como ancho x alto, p. Ej. 1600 x 900. Los espacios se ignoran pero se pueden usar para facilitar la lectura y **x** puede ser minúscula o mayúscula.

NOTA: WINDOW_SIZE ha reemplazado a MAXIMIZED. Por compatibilidad con versiones anteriores, estos nombres son intercambiables.

Sección **[DISPLAY]**

```
TOOL_EDITOR = tooledit x y
USER_COMMAND_FILE = plasmac_axis.py.py

EMBED_TAB_NAME = Estadísticas
EMBED_TAB_COMMAND = gladevcp -c plasmac_stats -x <XID> -u ./plasmac/plasmac_stats.py -H ./ ↩
    plasmac/plasmac_stats.hal ./plasmac/plasmac_stats.glade

#use uno de los siguientes dos

# Ejecutar marco en la pestaña detrás de la vista previa
EMBED_TAB_NAME = Ejecución de plasma
EMBED_TAB_COMMAND = gladevcp -c plasmac_run -x <XID> -u ./plasmac/plasmac_run.py -H ./ ↩
    plasmac/plasmac_run.hal ./plasmac/plasmac_run_tab.glade

# marco de ejecución en el panel en el lado izquierdo
```

```
#GLADEVCP = -c plasmac_run -u ./plasmac/plasmac_run.py -H ./plasmac/plasmac_run.hal ./ ↩
    plasmac/plasmac_run_panel.glade

EMBED_TAB_NAME = Configuración de plasma
EMBED_TAB_COMMAND = gladevcp -c plasmac_config -x <XID> -u ./plasmac/plasmac_config.py -H  ↩
    ./plasmac/plasmac_config.hal ./plasmac/plasmac_config.glade

EMBED_TAB_NAME = Extras
EMBED_TAB_COMMAND = gladevcp -c plasmac_wizards -x {XID} -u ./plasmac/plasmac_wizards.py ./ ↩
    plasmac/plasmac_wizards.glade
```

La ventana Ejecutar se puede mostrar como una de:

- Un panel en el lado derecho de la GUI del eje que es adecuado para pantallas panorámicas.
- Una pestaña detrás de la pestaña Vista previa que es adecuada para pantallas de relación 4:3.

4.7.16.3. Específicos de GUI Gmoccapy

Sección **[DISPLAY]**

```
EMBED_TAB_NAME     = plasmac_buttons
EMBED_TAB_LOCATION = box_cooling
EMBED_TAB_COMMAND  = gladevcp -c plasmac_buttons -x <XID> -u ./plasmac/plasmac_buttons.py - ↩
    H ./plasmac/plasmac_buttons.hal ./plasmac/plasmac_buttons.glade
EMBED_TAB_NAME     = plasmac_control
EMBED_TAB_LOCATION = box_spindle
EMBED_TAB_COMMAND  = gladevcp -c plasmac_control -x <XID> -u ./plasmac/plasmac_control.py - ↩
    H ./plasmac/plasmac_control.hal ./plasmac/plasmac_control.glade
EMBED_TAB_NAME     = Statistics
EMBED_TAB_LOCATION = ntb_preview
EMBED_TAB_COMMAND  = gladevcp -c plasmac_stats -x <XID> -u ./plasmac/plasmac_stats.py -H ./ ↩
    plasmac/plasmac_stats.hal ./plasmac/plasmac_stats.glade
EMBED_TAB_NAME     = Plasma Run

#use uno de los siguientes dos

# panel de ejecución en la pestaña detrás de la vista previa
EMBED_TAB_LOCATION  = ntb_preview
EMBED_TAB_COMMAND   = gladevcp -c plasmac_run -x <XID> -u ./plasmac/plasmac_run.py -H ./ ↩
    plasmac/plasmac_run.hal ./plasmac/plasmac_run_tab.glade

# panel de ejecución en el panel del lado izquierdo
#EMBED_TAB_LOCATION = box_left
#EMBED_TAB_COMMAND  = gladevcp -c plasmac_run -x <XID> -u ./plasmac/plasmac_run.py -H ./ ↩
    plasmac/plasmac_run.hal ./plasmac/plasmac_run_panel.glade

EMBED_TAB_NAME     = Plasma Config
EMBED_TAB_LOCATION = ntb_preview
EMBED_TAB_COMMAND  = gladevcp -c plasmac_config -x <XID> -u ./plasmac/plasmac_config.py -H  ↩
    ./plasmac/plasmac_config.hal ./plasmac/plasmac_config.glade
EMBED_TAB_NAME     = plasmac_monitor
EMBED_TAB_LOCATION = box_tool_and_code_info
EMBED_TAB_COMMAND  = gladevcp -c plasmac_monitor -x <XID> -u ./plasmac/plasmac_monitor.py - ↩
    H ./plasmac/plasmac_monitor.hal ./plasmac/plasmac_monitor.glade

EMBED_TAB_NAME     = Extras
EMBED_TAB_LOCATION = ntb_preview
EMBED_TAB_COMMAND  = gladevcp -c plasmac_wizards -x {XID} -u ./plasmac/plasmac_wizards.py  ↩
    ./plasmac/plasmac_wizards.glade
```

El Panel de ejecución se puede mostrar como uno de:

- Un panel en el lado izquierdo de la GUI de Gmoccapy que es adecuado para pantallas panorámicas.
- Una pestaña detrás de la pestaña Vista previa que es adecuada para pantallas de relación 4:3.

4.7.17. Botones de usuario personalizados

Hay botones disponibles que son programables en el archivo .ini.

Hay cinco botones disponibles en la ventana principal de la GUI del eje, numerados del 1 al 5 de izquierda a derecha.

Hay cuatro botones disponibles en la ventana principal de la GUI de Gmoccapy, numerados del 1 al 4 de arriba a abajo.

Hay diez botones adicionales disponibles en el panel Extras, numerados del 10 al 19 desde la parte superior izquierda a la inferior derecha.

Todas las configuraciones de archivo .ini para los botones están en la sección [PLASMAC].

4.7.17.1. Nombres de botones

El texto que aparece en el botón se configura de la siguiente manera:

```
BUTTON_n_NAME = HAL Show
```

Donde n es el número del botón y **HAL Show** es el texto.

Para el texto en varias líneas en Axis, divida el texto con \

```
BUTTON_n_NAME = HAL\Show
```

4.7.17.2. Código de botón

Los botones pueden ejecutar Comandos externos, G-Code o cualquiera de las cuatro funciones especiales, Test de sonda, Test Ohmico, Tipo de corte o Cambiar consumibles. Además de esto, los botones 10-19 en el panel Extras pueden cargar un programa de código G.

Comandos externos

Para ejecutar un comando externo, el comando está precedido por un carácter %.

```
BUTTON_n_CODE = %halshow
```

[underline] **Código G**

Para ejecutar G-Code, simplemente ingrese el código que se ejecutará.

```
BUTTON_n_CODE = G0 X100
```

Para ejecutar una subrutina existente.

```
BUTTON_n_CODE = o<my_subroutine> call
```

Las variables del archivo .ini se pueden ingresar usando {} (debe poner un espacio después del })

```
BUTTON_n_CODE = G0 X {JOINT_0 HOME} Y1
BUTTON_n_CODE = G53 G0 Z [{AXIS_Z MAX_LIMIT} - 1.001]
```

Se pueden ejecutar múltiples códigos separandolos con una \

```
BUTTON_n_CODE = G0 X0 Y0 \ G1 X5 \ G1 Y5
```

Los comandos externos y el código G se pueden mezclar en el mismo botón.

```
BUTTON_n_CODE = %halshow \ g0x.5y.5 \ %halmeter
```

[underline] **Prueba de sonda**

```
BUTTON_n_CODE = probe-test 30
```

PlasmaC comenzará un sondeo y cuando se detecte la sonda, el eje Z se elevará a la altura de perforación que se muestra actualmente en el panel de ejecución. Luego esperará en este estado el tiempo especificado (en este caso, 30 segundos) y luego devolverá el eje Z a la parte superior.

Prueba óhmica

```
BUTTON_n_CODE = ohmic-test
```

PlasmaC activará la sonda óhmica y, si se detecta, se iluminará el indicador LED en el panel del monitor. El objetivo principal de esto es permitir una prueba rápida para una punta de antorcha en corto.

[underline] **Tipo de corte**

```
BUTTON_n_CODE = cut-type
```

Este botón, si está seleccionado, alternará entre los dos tipos de corte.

Cambiar consumibles

```
BUTTON_n_CODE = change-consumables X10 Y10 F1000
```

Este botón mueve la antorcha a las coordenadas especificadas cuando la máquina está en pausa para permitir un fácil acceso para cambiar los consumibles.

Las entradas válidas son Xnnn Ynnn Fnnn. F es obligatorio y se requiere al menos uno de X o Y.

Las entradas X e Y están en coordenadas absolutas de la máquina. Si faltan X o Y, se utiliza la coordenada actual para ese eje.

Hay tres métodos para volver a las coordenadas anteriores:

- Presione el botón nuevamente y la antorcha regresará y la máquina esperará un comando de reanudación.
- Reanude el programa que hace que la antorcha regrese, luego el programa comenzará a ejecutarse.
- Pare el programa que hace que la antorcha regrese, entonces el programa abortará.

Carga

Los botones 10-19 en el Panel Extras pueden cargar un programa de Código G utilizando el siguiente formato.

```
BUTTON_n_CODE = load ruta/a/G-Code.ngc
```

La ruta es relativa al directorio especificado por **PROGRAM_PREFIX** en el archivo .ini. Por lo general, esto es ~/linuxcnc/nc_files, por lo que si su archivo G-Code estaba en un subdirectorio de su directorio **PROGRAM_PREFIX** llamado **plasma**, entonces la entrada sería **plasma/G-Code.ngc**.

4.7.17.3. Imagen de botón

Los botones 10-19 en el Panel Extras pueden mostrar una imagen usando el siguiente formato. Si se especifica una imagen, esto tendrá prioridad y la imagen se mostrará en lugar del nombre del botón.

```
BUTTON_n_IMAGE = ruta/a/image.png
```

La ruta es relativa al directorio de configuración, por lo que si su imagen estaba en un subdirectorio de su directorio de configuración llamado **images**, la entrada sería *images/image.png *.

Todas las imágenes se muestran en un formato de 60 x 60.

4.7.18. Archivo de material

PlasmaC utiliza un archivo que contiene parámetros de corte para materiales. Este archivo se utiliza para el manejo de materiales.

El nombre del archivo se deriva de **[EMC]MACHINE** en el archivo de entrada, por lo que una máquina llamada **METRIC_PLASMAC** tendría un archivo de materiales llamado **metric_plasmac_material.cfg**.

Las siguientes variables son obligatorias y aparecerá un mensaje de error si falta alguna cuando se carga el archivo de material.

- PIERCE_HEIGHT
- PIERCE_DELAY
- CUT_HEIGHT
- CUT_SPEED

Las siguientes variables son opcionales. Si alguno no se detecta o no tiene un valor asignado, se les asignará un valor de 0 y no aparecerá ningún mensaje de error.

- NAME
- KERF_WIDTH
- THC
- PUDDLE_JUMP_HEIGHT
- PUDDLE_JUMP_DELAY
- CUT_AMPS
- CUT_VOLTS
- PAUSE_AT_END
- GAS_PRESSURE
- CUT_MODE

ADVERTENCIA: Es responsabilidad del operador asegurarse de que se incluyan las variables si son un requisito para que se ejecute el Código G.

El archivo de material utiliza el siguiente formato:

```
[MATERIAL_NUMBER_1]
NAME                            = nombre
KERF_WIDTH                      = valor
THC                             = valor (0 = apagado, 1 = encendido)
PIERCE_HEIGHT           = valor
PIERCE_DELAY            = valor
PUDDLE_JUMP_HEIGHT      = valor
PUDDLE_JUMP_DELAY       = valor
CUT_HEIGHT                      = valor
CUT_SPEED                       = valor
CUT_AMPS                        = valor (solo para información a menos que las  ↩
    comunicaciones Powermax estén habilitadas)
CUT_VOLTS                       = valor (solo modos 0 y 1, si no usa muestreo automático de ↩
     voltaje)
PAUSE_AT_END            = valor
GAS_PRESSURE            = valor (solo usado para comunicaciones Powermax)
CUT_MODE                        = valor (solo usado para comunicaciones Powermax)
```

Es posible agregar un material nuevo, eliminar material o editar material existente desde el Panel de ejecución.

También es posible editar el archivo de material con un editor de texto mientras se está ejecutando LinuxCNC y luego, una vez que se hayan guardado los cambios, presione **Volver a cargar** en el Panel de ejecucion para volver a cargar el archivo de material.

4.7.19. Configuraciones de ejemplo

Hay archivos de configuración de ejemplo para las GUI Axis y Gmoccapy que usan el componente PlasmaC HAL para simular máquinas de corte por plasma.

- metric_plasmac.ini
- imperial_plasmac.ini

Si ha copiado alguna configuración de ejemplo en su directorio ~/linuxcnc/configs, podrá actualizarlas ejecutando el Configurador en modo Actualización. Esto reemplazará cualquier archivo de sistema de plasmac copiado con un enlace a la última versión a la que se haya actualizado.

4.7.20. Muestras de NGC

Hay algunos archivos de código G de muestra en nc_files/plasmac.

4.7.21. Panel de prueba

Hay un directorio ./test en las fuentes que tiene un Panel de prueba simple y un archivo python asociado que se puede usar para probar la configuración de ejemplo como se referencia en el archivo .ini.

Si desea utilizar esto en su propia configuración de sim, deberá crear un enlace a este directorio y editar su archivo .ini para adaptarlo.

4.7.22. Actualizar una configuración existente de LinuxCNC

El método de actualización depende del tipo de instalación de LinuxCNC:

4.7.22.1. Actualizar LinuxCNC

Desde una terminal:

```
$ sudo apt-get update
$ sudo apt-get dist-upgrade
```

4.7.23. Actualizar PlasmaC

4.7.23.1. Actualización automática

Después de haber actualizado LinuxCNC, PlasmaC probará si necesita una actualización cuando se inicie LinuxCNC. Si descubre que necesita actualizarse, procederá automáticamente con la actualización. Una vez completada la actualización, puede recibir una advertencia de que LinuxCNC debe reiniciarse. Esto depende de qué archivos necesitan ser cambiados.

4.7.23.2. Actualización manual

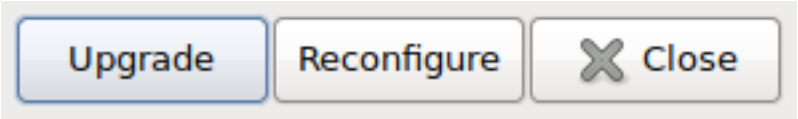
Si no desea permitir que PlasmaC realice actualizaciones automáticas, puede evitar esto agregando **MANUAL_UPGRADE = 1** a la sección [PLASMAC] del archivo .ini:

```
MANUAL_UPGRADE = 1
```

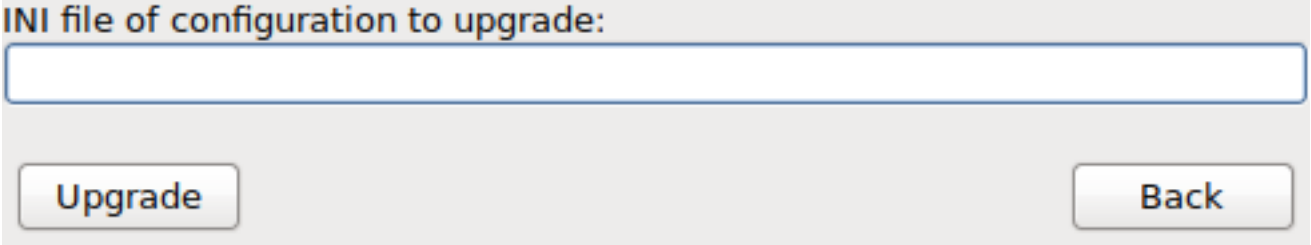
Entonces necesitará actualizar manualmente PlasmaC abriendo un terminal y ejecutando el configurador:

```
$ python linuxcnc/<your_configuration_directory>/configurator.py
```

La ventana de selección ahora es visible:



Seleccione **Actualizar** en la ventana de selección, esto muestra un cuadro de diálogo de información, seleccione **Continuar** y aparecerá la ventana Actualizar.



Seleccione el archivo .ini de la configuración de PlasmaC que desea actualizar, haga clic en **Actualizar** y se actualizará su configuración de PlasmaC.

4.7.24. Reconfigurar una configuración de PlasmaC existente

El Configurador solo puede modificar:

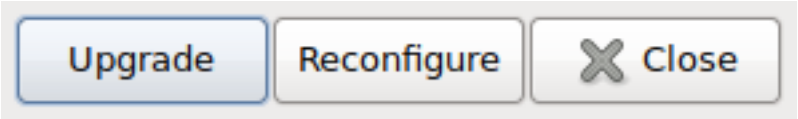
- Los pines HAL que conectan PlasmaC a la máquina.
- El modo utilizado por PlasmaC.
- La posición del Panel de ejecución en la GUI.

La reconfiguración de una configuración de Plasmac se realiza desde el Configurador que se encuentra en su directorio de configuración.

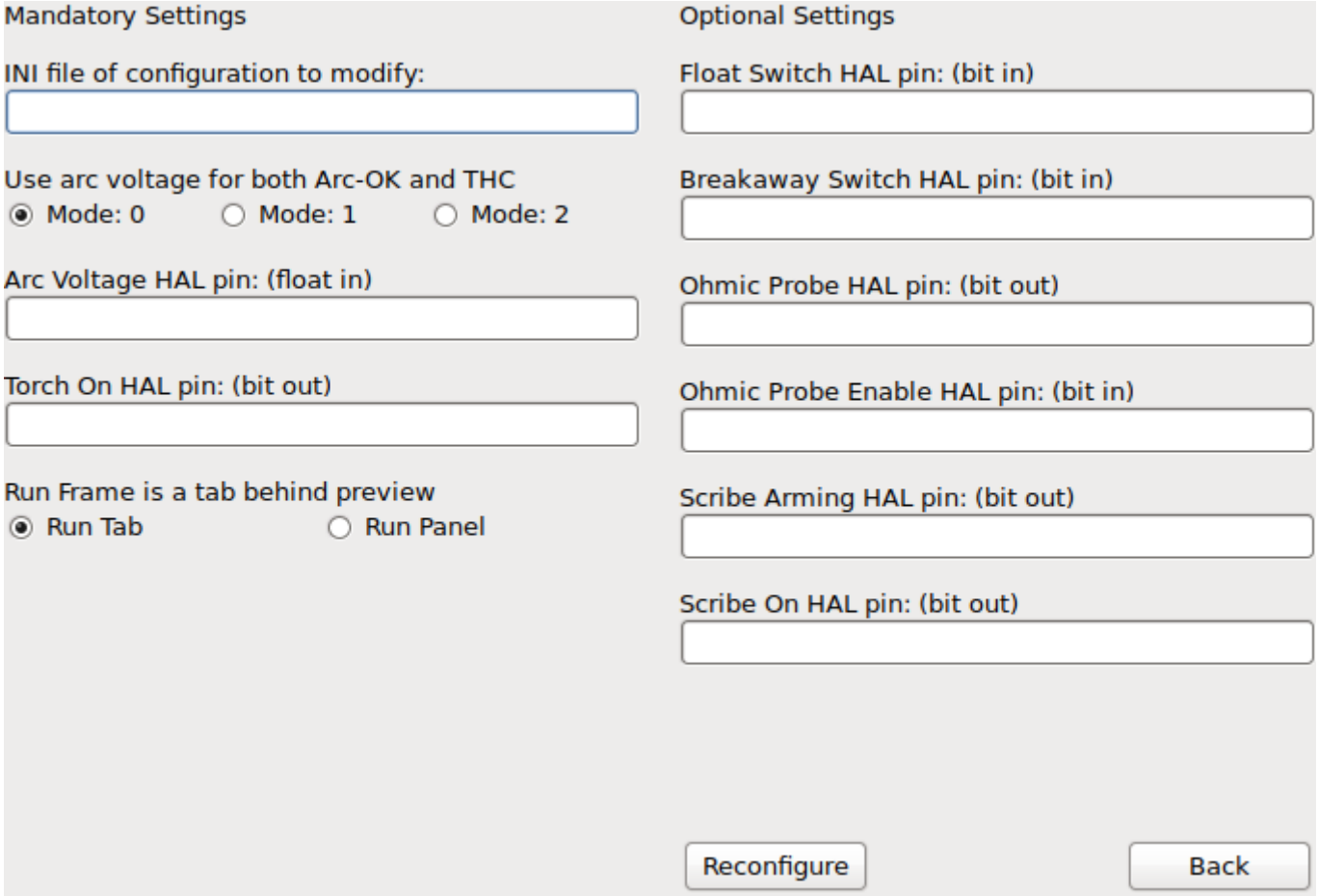
Para iniciar el configurador, abra una terminal e ingrese:

```
$ python linuxcnc/<your_configuration_directory>/configurator.py
```

La ventana de selección ahora es visible:



Seleccione **Reconfigurar** de la ventana de selección, esto muestra un diálogo de información, seleccione **Continuar** y se mostrará la ventana Reconfigurar.



Seleccione el archivo .ini de la configuración de PlasmaC que desea reconfigurar.

Si cambia de modo, notará que los cuadros de entrada cambian según el modo seleccionado actualmente.

Cuando todas las entradas sean correctas, haga clic en **Reconfigurar**.

Se pueden encontrar descripciones de entradas aquí

NOTA: El nombre de la máquina y el archivo .hal no se pueden modificar.

4.7.25. Cambiar tipo de instalación de LinuxCNC

Si ha cambiado el tipo de instalación de LinuxCNC de Buildbot a Run In Place o viceversa, deberá ejecutar el Configurador desde el nuevo directorio de instalación de LinuxCNC, NO desde el directorio ~/linuxcnc/configs. Esto es necesario para restablecer los enlaces de archivo para el tipo de instalación correcto.

Desde una terminal, realice una de las siguientes acciones:

Restablecer a una nueva instalación de Buildbot:

```
$ python /usr/share/doc/linuxcnc/examples/sample-configs/by_machine/plasmac/configurator.py
```

Restablecer a una nueva instalación Run In Place:

```
$ python ~/linuxcnc-dev/configs/by_machine/plasmac/configurator.py
```

4.7.26. Materialverter

Esta aplicación se utiliza para convertir tablas de herramientas existentes en archivos de material PlasmaC. También puede crear un archivo de material desde la entrada manual de usuario a los campos de entrada.

En esta etapa, las únicas conversiones disponibles son de SheetCam y Fusion360.

Las tablas de herramientas de Sheetcam están completas y la conversión es completamente automática. El archivo de herramienta SheetCam debe estar en el formato .tools de SheetCam.

Las tablas de herramientas de Fusion360 no tienen todos los campos obligatorios, por lo que se le solicita al usuario los parámetros faltantes. El archivo de herramienta Fusion360 debe estar en el formato Fusion360 .json.

Si tiene un formato que desea convertir, cree un **Nuevo tema** en la sección PlasmaC forum.

Materialverter se puede ejecutar desde un administrador de archivos GUI haciendo doble clic en **materialverter.py** en su directorio de configuración o se puede ejecutar desde un terminal con el siguiente comando:

```
$ python ~/linuxcnc/configs/<nombre_maquina>/materialverter.py
```

Aparecerá el cuadro de diálogo del convertidor de materiales:

Seleccione uno de:

- Manual: para crear manualmente un nuevo archivo de material.
- SheetCam: para convertir un archivo de herramienta SheetCam.
- Fusion360: para convertir un archivo de herramienta Fusion360.

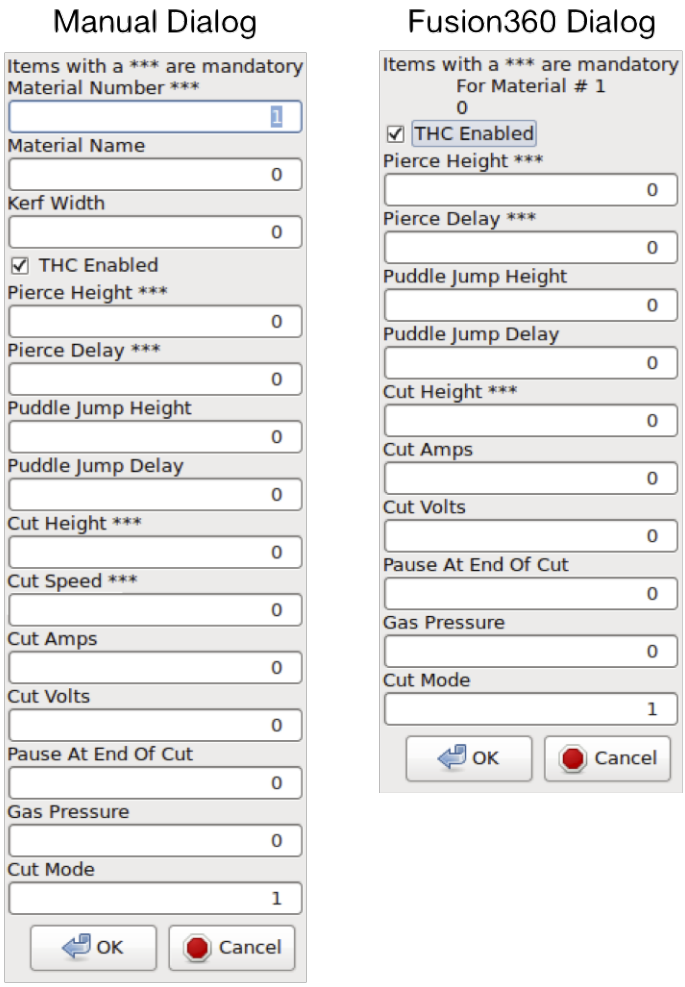
Solo para SheetCam, seleccione si desea un archivo de salida métrico o imperial.

Para una conversión, seleccione el archivo de entrada que desea convertir.

Seleccione el archivo de salida en el que desea escribir, normalmente sería ~/linuxcnc/configs/<machine_name>_material.cfg, aunque puede seleccionar un archivo diferente y editar manualmente su <machine_name>_material.cfg.

Haga clic en Crear/Convertir y se creará su nuevo archivo de material.

Para una creación manual o una conversión de Fusion360, se mostrará un cuadro de diálogo con todos los parámetros disponibles mostrados para la entrada. Cualquier entrada marcada con *** es obligatoria y todas las demás entradas son opcionales dependiendo de su configuración.



NOTA: Si selecciona ~/linuxcnc/configs/<machine_name>_material.cfg, si este archivo ya existe, se sobrescribirá.

4.7.27. Comunicaciones Hypertherm Powermax

Las comunicaciones se pueden establecer con una cortadora de plasma Hypertherm Powermax que tiene un puerto RS485. Esto permite configurar **Modo de corte**, **Amperaje de corte** y **Presión de gas** automáticamente a partir de los **Parámetros de corte** del archivo de materiales.

Si la presión de gas se establece en cero, la Powermax calculará automáticamente la presión requerida desde el modo de corte, la corriente de corte, el tipo de antorcha y la longitud de la antorcha.

Cambiar el modo de corte establecerá la presión de gas en cero, lo que hará que la máquina use su modo automático de presión de gas.

Los valores máximos y mínimos de estos parámetros se leen desde el cortador de plasma y los botones giratorios relacionados en los Parámetros de corte están limitados por estos valores. La presión de gas no se puede cambiar de cero hasta que se hayan establecido las comunicaciones.

Esta característica se habilita configurando el nombre de puerto correcto para la variable PM_PORT en la sección [PLASMAC] del archivo .ini. Si la variable PM_PORT no está configurada en el archivo .ini, los widgets asociados con esta función no serán visibles.

```
[PLASMAC]
PM_PORT = /dev/ttyusb0
```

Si no está seguro del nombre de su puerto, hay un script de Python en el directorio de configuración que mostrará todos los puertos disponibles y también se puede usar para probar las comunicaciones con la unidad de plasma antes de usar esta función en PlasmaC. Para usar el script de prueba, ingrese lo siguiente en una terminal.

```
$ python ~ /user/linuxcnc/configs/<your_config>/pmx_test.py
```

La visualización de las unidades de presión de gas, ya sea psi o bar, está determinada por los datos recibidos durante la configuración inicial del enlace de comunicación y luego se muestra junto al botón giratorio de Presión de gas en el Panel de control.

La máquina Powermax entrará en modo remoto después de que se hayan establecido las comunicaciones y se controlará de forma remota. La conexión se puede validar observando la pantalla de Powermax.

Para volver a cambiar la Powermax al modo local, puede:

1. Deshabilitar Powermax Comms desde el panel Ejecutar
2. Cerrar LinuxCNC, que pondrá el Powermax en modo local durante el apagado.
3. Apagar la Powermax durante 30 segundos y luego volver a encenderla.

NOTA: Para usar la función de comunicaciones Powermax es necesario tener instalado el módulo pyserial de python.
Si pyserial no está instalado, se mostrará un mensaje de error.

Para instalar pyserial, escriba lo siguiente en un terminal:

```
sudo apt-get install python-serial
```

En el apéndice de este documento se muestran los diagrama de conexión típicos, así como las interfaces de trabajo confirmadas.

4.7.28. Mesa THCAD

El Mesa THCAD es la forma más común de obtener el voltaje de arco de un cortador de plasma. El THCAD puede usarse para configuraciones de puerto paralelo o para configuraciones que usan otro hardware de Mesa Electronics.

El método preferido es configurar los parámetros necesarios para el THCAD en su archivo <machine_name> .hal.

4.7.28.1. Configuración de puerto paralelo

Un ejemplo de configuración de puerto paralelo usando un link:http://linuxcnc.org/docs/html/man/man9/encoder.9.html [codificador de software].

Configuración de puentes THCAD **UNIPOLAR** y **F/64**

NOTA: Si su computadora tiene buena latencia, entonces puede usar **F/32**

```
loadrt encoder num_chan=1
setp encoder.0.counter-mode 1
setp encoder.0.position-scale -1
addf encoder.update-counters base-thread
addf encoder.capture-position servo-thread
net arc-voltage-raw parport.1.pin-04-in encoder.0.phase-A
```

Luego debe tener en cuenta que la entrada **Pin HAL de voltaje de arco** en el Configurador PlasmaC será:

```
encoder.0.velocity
```

4.7.28.2. Configurar Mesa 7i96:

Ejemplo de configuración de Mesa 7i96 con el codificador hardware integrado. Otro hardware de Mesa será similar pero con un identificador de placa diferente.

Configuración de puentes THCAD **UNIPOLAR** y **F/32**

```
setp hm2_7i96.0.encoder.00.scale -1
setp hm2_7i96.0.encoder.00.filter 1
setp hm2_7i96.0.encoder.00.counter-mode 1
```

Luego debe tener en cuenta que la entrada **Pin HAL de voltaje de arco** en el Configurador PlasmaC será:

```
hm2_7i96.0.encoder.00.velocity
```

NOTA: Hay un filtro paso bajo disponible que puede ser útil si se usa un THCAD y hay mucho ruido en el voltaje de arco devuelto.

4.7.28.3. Obtención de la relación de división

THCAD-10

Si se conecta a un puerto CNC de plasma, la relación del divisor se selecciona de la máquina de plasma.

Si se conecta al voltaje de arco total de las máquinas de plasma, entonces una configuración común es una resistencia de 1 megOhm de arco negativo a THCAD negativo y una resistencia de 1 megOhm de arco positivo a THCAD positivo. Esto proporciona una lectura a escala completa de 210V. El THCAD puede manejar sobretensiones de hasta 500V, por lo que esto no es un problema.

Para obtener la relación de división, use (R1 + R2 + 100000) / 100000, por lo que para lo anterior sería (1000000 + 1000000 + 100000) / 100000 dando una relación de 21.

IMPORTANTE: SI UTILIZA UNA FUENTE DE ALIMENTACIÓN DE PLASMA HF START, CADA UNA DE ESTAS RESISTENCIAS DEBE ESTAR COMPUESTA POR VARIAS RESISTENCIAS DE ALTO VOLTAJE.

[underline] **THCAD-300**

Esto está conectado al voltaje de arco completo de las máquinas de plasma y la relación del divisor es 1

4.7.28.4. Valores de calibración

Veamos como obtener los valores iniciales para offsets de voltaje y escala de voltaje en el panel de configuración.

Si observa la parte posterior del THCAD, verá una etiqueta de calibración con:

```
    THCAD-nnn

 0V 121.1 kHz
 5V 925.3 kHz
```

o valores similares; estas son las frecuencias generadas por el THCAD para 0 voltios y voltios a escala completa, respectivamente.

Ahora puede calcular los valores como se muestra a continuación o usar este calulador en línea

Escala de voltaje

Usa la fórmula:

```
Divider_Ratio / ((THCAD_Full_Scale_Frequency - THCAD_0V_Frequency) /  ↩
    THCAD_Frequency_Divider / THCAD_Full_Scale_Voltage)
```

Este resultado se utilizará para configurar **Escala de voltaje** en el Panel de configuración.

El ejemplo de THCAD-10 con una configuración de puente de F/32 sería:

```
21 / ((925300 - 121100) / 32/10) = 0,00835613
```

Para nuestro cálculo anterior, utilizamos 21 de nuestro cálculo de divisor, 32 de la configuración de puente **F/32** y 10 como lectura a escala completa de la propia tarjeta THCAD.

El ejemplo de THCAD-300 sería:

```
1 / ((925300 - 121100) / 32/300) = 0.01193732
```

Para nuestro cálculo anterior, usamos 1 como nuestra relación de división, 32 de la configuración del puente **F/32** y 300 como lectura a escala completa de la tarjeta THCAD.

[underline] **Compensación de voltaje**

Usar la fórmula:

```
THCAD_0V_Frequency / THCAD_Frequency_Divider
```

Este resultado se utilizará para configurar **Compensación de voltaje** en el Panel de configuración.

Los ejemplos anteriores de THCAD-10 y THCAD-300 serían:

```
121100/32 = 3784.375
```

4.7.29. Soporte

La ayuda y el soporte en línea están disponibles en la sección PlasmaC del foro.

4.7.30. Apéndice

4.7.30.1. Códigos específicos de Plasmac

Descripción	* Código *
Comenzar corte	M3 $0 S1
Fin corte	M5 $0
Comenzar grabado	M3 $1 S1
Fin grabado	M5 $1
Comenzar punto central	M3 $2 S1
Fin punto central	M5 $0
Terminar todo lo anterior	M5 $-1
Seleccione un material	M190 Pn n denota número de material
Espere confirmación de cambio de material	M66 P3 L3 Q1 Q1 es el número de segundos a esperar, Es posible que deba aumentarse para archivos muy grandes de material
Establecer la velocidad de alimentación desde << material-handling, material>>	F#<_hal[plasmac.cut-feed-rate]>
Desactivar THC	M62 P2 (sincronizado con movimiento) M64 P2 (inmediato)
Habilitar THC	M63 P2 (sincronizado con movimiento) M65 P2 (inmediato)
Desactivar antorcha	M62 P3 (sincronizado con movimiento) M64 P3 (inmediato)
Habilitar antorcha	M63 P3 (sincronizado con movimiento) M65 P3 (inmediato)

Establezca velocidad en un porcentaje de la velocidad de avance	M67 E3 Qn (sincronizado con movimiento) M68 E3 Qn (inmediato) n es el porcentaje a establecer 10 es el mínimo, debajo de esto se establecerá en 100 % *Se recomienda tener M68 E3 Q0 en preámbulo y postámbulo *
Compensación del cortador - izquierda del camino	G41.1 D#<_hal[plasmac_run.kerf-width-f]>
Cortador << compensación de cortador, compensación>> - derecho de trayectoria	G42.1 D#<_hal[plasmac_run.kerf-width-f]>
Compensación Cortador off	G40 **Tenga en cuenta que M62 a M68 no son válidos mientras la compensación de corte está activada**
Corte de agujeros a una velocidad de alimentación del 60 %	#<holes> = 1 para agujeros de menos de 32mm (1.25") de diámetro
Corte de agujeros a una velocidad de alimentación del 60 %, apague la antorcha al final del agujero, continuar camino del agujero para sobrecorte	#<holes> = 2 para agujeros de menos de 32mm (1.25") de diámetro longitud de corte superior = 4mm (0.157")
Especifique diámetro de agujero para #<holes> = 1 y #<holes> = 2	#<m_diameter> = nn (métrica) #<i_diameter> = nn (imperial)
Especifique longitud de sobrecorte	#<oclength> = n
Crear o editar materiales. opciones: 1 - Crear temporal por defecto 2 - Agregar si no existe 3 - Sobrescribir si existe, agregue nuevo	parámetros obligatorios: (o=<opción>, nu=<nn>, na=<ll>, ph=<nn>, pd=<nn>, ch=<nn>, fr=<nn>) parámetros opcionales: kw=<nn>, th=<nn>, ca=<nn>, cv=<nn>, pe=<nn>, gp=<nn>, cm=<nn>, jh=<nn>, jd=<nn>

4.7.30.2. Ejemplos de Plasmac Gcode

Descripción	**Ejemplo**
Seleccionar material y hacer un corte normal	M190 P3 M66 P3 L3 Q1 F#<_hal[plasmac.cut-feed-rate]> M3 $0 S1 . . M5 $0
Establecer la velocidad al 100 % de CutFeedRate	M67 E3 Q0 or M67 E3 Q100
Establecer la velocidad al 40 % de CutFeedRate	M67 E3 Q60
Establecer la velocidad al 40 % de CutFeedRate	M67 E3 Q40

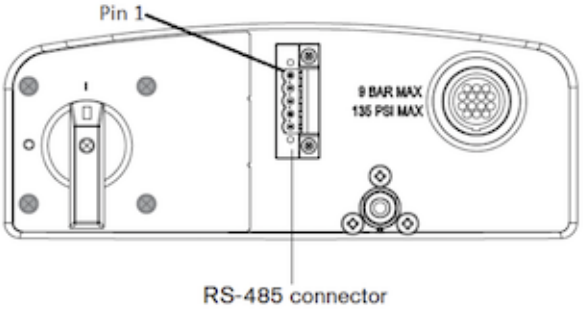
Corte un agujero con velocidad reducida al 60 % utilizando la configuración de velocidad	G21 (métrico) G64 P0.05 M52 P1 (permitir movimiento en pausa) F#<_hal[plasmac.cut-feed-rate]> G0 X10 Y10 M3 $0 S1 (corte inicial) G1 X0 M67 E3 Q60 (reducir la velocidad de alimentación al 60 %) G3 I10 (el agujero) M67 E3 Q0 (restablecer la velocidad de alimentación al 100 %) M5 $0 (corte final) G0 X0 Y0 M2 (final del trabajo)
Cortar un agujero con un 60 % de velocidad reducida utilizando el comando #<holes>	G21 (métrico) G64 P0.05 M52 P1 (permitir pausa en movimiento) #<holes> = 1 (reducción de velocidad para agujeros) F#<_hal[plasmac.cut-feed-rate]> G0 X10 Y10 M3 $0 S1 (corte inicial) G1 X0 G3 I10 (el agujero) M5 $0 (corte final) G0 X0 Y0 M2 (final del trabajo)
Corte un agujero con sobrecorte con la antorcha deshabilitada	G21 (métrico) G64 P0.05 M52 P1 (permitir pausa en movimiento) F#<_hal[plasmac.cut-feed-rate]> G0 X10 Y10 M3 $0 S1 (corte inicial) G1 X0 M67 E3 Q60 (reducir la velocidad de alimentación al 60 %) G3 I10 (el agujero) M62 P3 (apagar la antorcha) G3 X0.8 Y6.081 I10 (continua movimiento durante 4mm) M63 P3 (permitir encender la antorcha) M67 E3 Q0 (restablecer la velocidad de alimentación al 100 %) M5 $0 (corte final) G0 X0 Y0 M2 (end job)
Corte un agujero con sobrecorte con el comando #<holes>	G21 (métrico) G64 P0.05 M52 P1 (permitir pausa movimiento) #<holes> = 2 (sobrecorte para agujeros) F#<_hal[plasmac.cut-feed-rate]> G0 X10 Y10 M3 $0 S1 (empezar corte) G1 X0 G3 I10 (el agujero) M5 $0 (corte final) G0 X0 Y0 M2 (fin de trabajo)

Corte un agujero con sobrecorte de 6.5 mm con el comando #<holes>	G21 (métrico) G64 P0.05 M52 P1 (permitir pausa movimiento) #<holes> = 2 (sobrecorte para agujeros) <oclength> = 6.5 (6.5mm longitud de sobrecorte) F<_hal[plasmac.cut-feed-rate]> G0 X10 Y10 M3 $0 S1 (empezar corte) G1 X0 G3 I10 (el agujero) M5 $0 (corte final) G0 X0 Y0 M2 (fin de trabajo)
Seleccione grabador y seleccione la antorcha al final del grabado	. . M52 P1 (movimiento en pausa on) F#<_hal[plasmac.cut-feed-rate]> T1 M6 (seleccionar grabador) G43 H0 (aplicar offsets) M3 $1 S1 (empezar plasmac con grabador) . . T0 M6 (seleccionar antorcha) G43 H0 (aplicar offsets) G0 X0 Y0 (posicion parking) M5 $1 (fin)
punteo del centro de agujero.	(Requiere un pequeño comando de movimiento o no sucede nada) G21 (métrico) F99999 (alta velocidad de alimentación) G0 X10 Y10 M3 $2 S1 (punteo on) G91 (modo distancia relativa) G1 X0.000001 G90 (modo distancia absoluta) M5 $2 (punteo off) G0 X0 Y0 G90 M2
Crear material predeterminado de temporal	(o=1, nu=2, na=5mm Acero dulce 40A, ph=3.1, pd=0.1, ch=0.75, fr=3000)
Editar material, si no existe crear uno nuevo	(o=3, nu=2, na=5mm Acero dulce 40A, ph=3.1, pd=0.1, ch=0.75, fr=3000, kw=1.0)

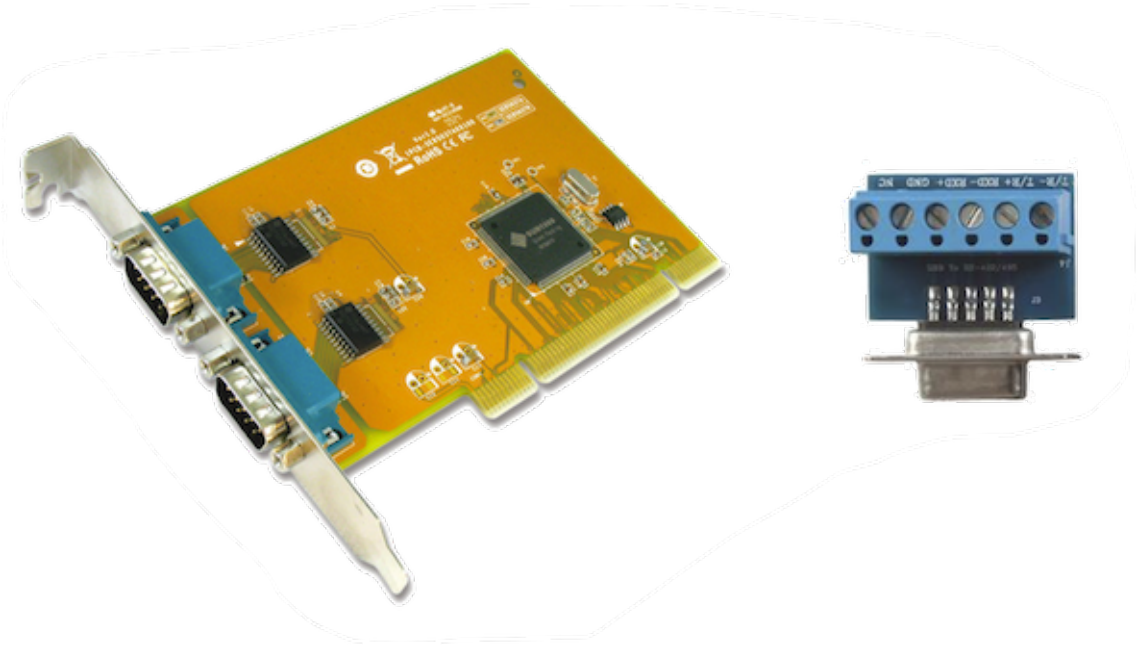
4.7.30.3. Conexiones RS485

Hypertherm Powermax 125:

Connection at Machine Pin No	Connection at USB RS485 Pin No (Left to Right)
1: Tx+ (Red)	1: T/R- (Black)
2: Tx-(Black)	2: T/R+ (Red)
3: Rx+ (Brown)	3: RXD- (White)
4: Rx- (White)	4: RXD+ (Brown)
5: GND (Green)	5: GND (Green)



Tarjeta PCI Sunnix SER5037A con placa de conexión:



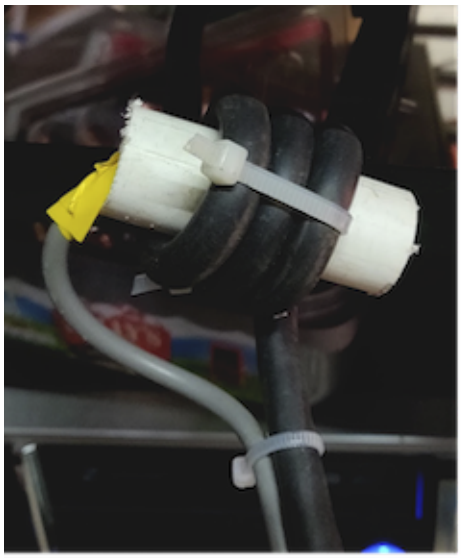
Adaptador convertidor Dtech DT-5019 USB a RS485:

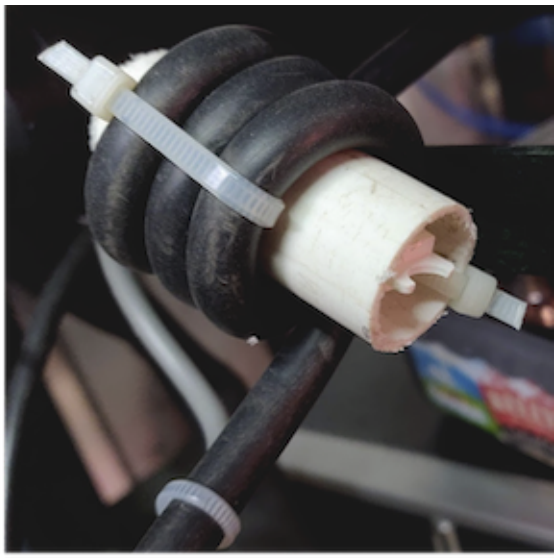
4.7.30.4. Arc OK con un relé reed

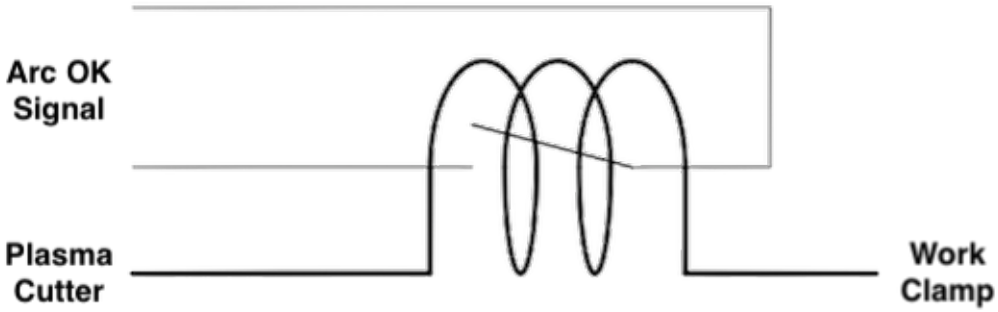
Un método efectivo y muy confiable para obtener una señal Arc OK de una fuente de alimentación de plasma sin un puerto CNC es montar un relé de láminas dentro de un tubo no conductor y envolver y asegurar tres vueltas del cable de trabajo alrededor del tubo.

Este conjunto actuará como un relé que se encenderá cuando la corriente fluya a través del cable de trabajo, lo que solo ocurre cuando se ha establecido un arco de corte.

Esto requerirá que PlasmaC se opere en Modo 1 en lugar del Modo 0. Consulte las secciones Archivo INI y Modos PlasmaC para obtener más información.






Arc OK
Signal
Plasma
Cutter
Work
Clamp

Capítulo 5

Programacion

5.1. Sistemas de coordenadas

5.1.1. Introducción

Este capítulo describe los offsets tal como los utiliza LinuxCNC. Éstos incluyen:

- Coordenadas de la máquina (G53)
- Nueve offsets del Sistema de Coordenadas (G54-G59.3)
- Offsets globales (G92) y locales (G52)

5.1.2. Sistema de Coordenadas de la Máquina

Cuando se inicia LinuxCNC, las posiciones de cada eje definen el origen de la máquina. Cuando se hace home de un eje, el origen de la máquina para ese eje se establece en esa posición home. El origen máquina es el sistema de coordenadas de la máquina en el que se basan todos los demás sistemas de coordenadas. El código G53 puede usarse para moverse en el Sistema de Coordenadas Máquina.

5.1.3. Sistemas de coordenadas

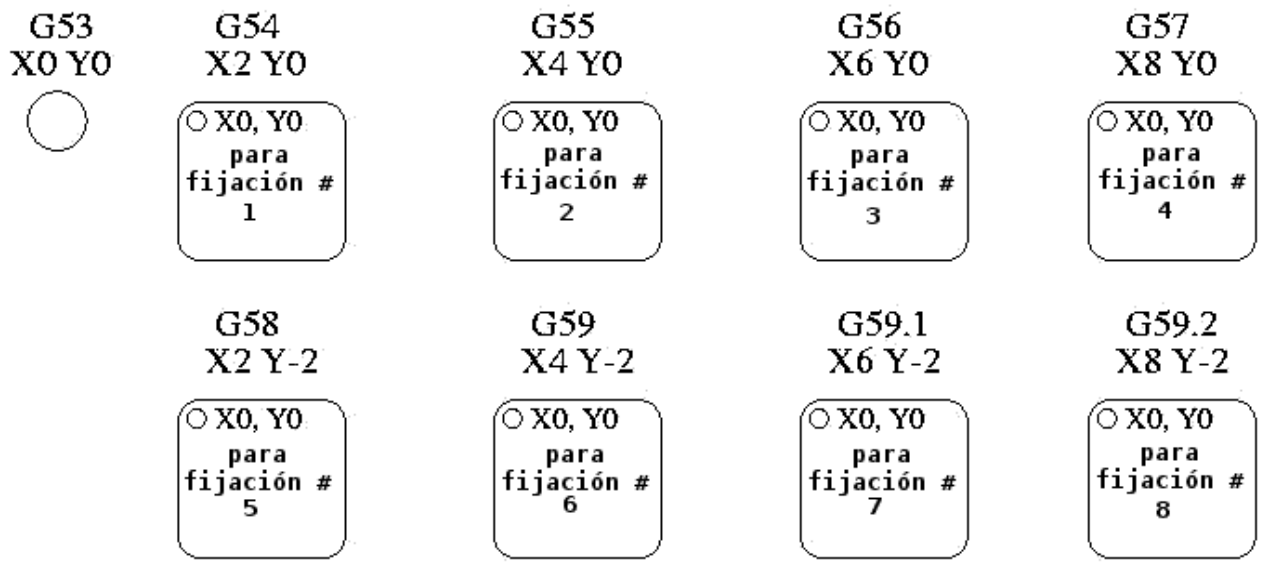


Figura 5.1: Ejemplo de sistemas de coordenadas

Desplazamientos del sistema de coordenadas

- G54 - usa el sistema de coordenadas 1
- G55 - usa el sistema de coordenadas 2
- G56 - usa el sistema de coordenadas 3
- G57 - usa el sistema de coordenadas 4
- G58 - usa el sistema de coordenadas 5
- G59 - usa el sistema de coordenadas 6
- G59.1 - usar el sistema de coordenadas 7
- G59.2 - usa el sistema de coordenadas 8
- G59.3 - usa el sistema de coordenadas 9

Los offsets del sistema de coordenadas se utilizan para desplazar cada sistema de coordenadas desde el Sistema de Coordenadas de la Máquina. Esto permite que el código G se programe para piezas iguales sin importar la ubicación de la pieza en la máquina. El uso de offsets del sistema de coordenadas le permitirán mecanizar piezas en múltiples ubicaciones con el mismo código G.

Los valores de los offsets se almacenan en el archivo VAR solicitado por el archivo INI durante el inicio de LinuxCNC.

En el esquema de archivo VAR, el primer número de variable almacena el desplazamiento X, el segundo, el desplazamiento Y, y así sucesivamente para los nueve ejes. Hay juegos numerados así para cada uno de los offsets de sistema de coordenadas.

Cada una de las interfaces gráficas tiene una forma de establecer valores para estos offsets. También puede establecer estos valores editando el propio archivo VAR y reinicializando luego LinuxCNC para que lea los nuevos valores. Sin embargo, esta no es la forma recomendada. El uso de G10, G52, G92, G28.1, etc. son mejores formas de establecer las variables.

Ejemplo de parámetros G55.

| ==== | Eje | Variable | Valor | X | 5241 | 2.000000 | Y | 5242 | 1.000000 | Z | 5243 | -2.000000 | A | 5244 | 0,000000 | B | 5245 | 0.000000 | C | 5246 | 0.000000 | U | 5247 | 0.000000 | V | 5248 | 0.000000 | W | 5249 | 0.000000 | ====

Debería leer esto como mover las posiciones cero de G55 a X = 2 unidades, Y = 1 unidad, y Z = -2 unidades desde la posición de cero absoluto.

Una vez que hay valores asignados, una llamada a G55 en un bloque de programa desplaza la referencia cero absoluta por los valores almacenados. La siguiente línea sería mover cada eje a la nueva posición cero. A diferencia de G53, G54 hasta G59.3 son comandos modales. Actuarán en todos los bloques de código después de que uno de ellos se ha establecido. El programa que podría ejecutarse usando offsets de fijaciones requeriría solo una coordenada de referencia para cada una de las ubicaciones y todo el trabajo se realizaria allí. El siguiente código se ofrece como un ejemplo de cómo hacer un cuadrado usando los offsets G55 que configuramos arriba.

```
G55; utilizar el sistema de coordenadas 2
G0 X0 Y0 Z0
G1 F2 Z-0.2000
X1
Y1
X0
Y0
G0 Z0
G54; utilizar el sistema de coordenadas 1
G0 X0 Y0 Z0
M2
```

En este ejemplo, el G54 cerca del final deja el sistema de coordenadas G54 con todos los offsets a cero para que haya un código modal basado en ejes absolutos de máquina. Este programa asume que lo hemos hecho y usamos el comando final como un comando para cero máquina. Hubiera sido posible usar G53 y llegar al mismo lugar pero ese comando no habría sido modal y cualquier comando emitido después hubiera vuelto a usar los offsets G55 porque ese sistema de coordenadas todavía estaría en vigor.

5.1.3.1. Sistema de Coordenadas Predeterminado

Otra variable en el archivo VAR se vuelve importante cuando pensamos sobre offsets. Esta variable es la 5220. En los archivos predeterminados, su valor se establece en 1.00000. Esto significa que cuando LinuxCNC se inicia, debería usar el primer sistema de coordenadas como el predeterminado. Si configura esto a 9.00000 usaría el noveno sistema de compensación como predeterminado al iniciar o reiniciar. Cualquier valor que no sea un entero (decimal realmente) entre 1 y 9, o si la variable 5220 falta, hará que LinuxCNC vuelva al valor predeterminado de 1.00000.

5.1.3.2. Configuración de Offsets del Sistema de Coordenadas

El comando G10 L2x se puede usar para establecer los offsets del sistema de coordenadas:

- *G10 L2 P (1-9)* - Establece los offsets a un valor. La posición actual es irrelevante. (vea G10 L2 para más detalles)
- *G10 L20 P (1-9)* - Establece los offsets de modo que la posición actual se convierte en un valor. (vea G10 L20 para más detalles)

5.1.4. Offsets Locales y Globales

5.1.4.1. El comando G52

G52 se usa en un programa de pieza como un "Offset temporal del sistema de coordenadas local" dentro del sistema de coordenadas de la pieza de trabajo. Un ejemplo de uso es el caso cuando se mecanizan varias características idénticas en diferentes ubicaciones del material. Para cada una, *G52* programa un punto de referencia local dentro de las coordenadas de pieza, y un subprograma es llamado para maquinar la característica relativa a ese punto.

Los offsets de ejes *G52* se programan relativos a la coordenada de offset de la pieza de trabajo *G54* a *G59.3*. Como compensación local, *G52* se aplica después del offset de la pieza de trabajo, incluida la rotación. Por lo tanto, una característica parcial será mecanizada de forma idéntica en cada parte, independientemente de la orientación de la parte en el palet.

atención
En otros intérpretes de código g *G52*, como offset temporal, al establecer y salir del alcance localizado de una parte del programa, no persiste después del reinicio de la máquina, *M02* o *M30*. En LinuxCNC, *G52* comparte parámetros con *G92* que, por razones históricas, hace **persistir** a estos parámetros. Ver G92 Precauciones con Persistencia a continuación.

La programación de *G52 X1 Y2* da offsets al sistema de coordenada actual de la pieza de trabajo, 1 para X y 2 para Y. Por consiguiente, en el DRO, las coordenadas X e Y de la posición actual de la herramienta se reducirán en 1 y 2, respectivamente. Los ejes sin establecer en el comando, como Z en el anterior ejemplo, no se verán afectados; cualquier offset Z *G52* anterior permanecerá en efecto o, si no lo habia, el offset Z será cero.

El desplazamiento local temporal puede cancelarse con *G52 X0 Y0*. Cualquier eje no puesto a cero explícitamente retendrá el offset anterior.

G52 comparte los mismos registros que *G92* y, por lo tanto, *G52* es visible en el DRO y vista previa etiquetado como *G92*.

5.1.4.2. Los Comandos G92

G92 se usa típicamente de dos maneras conceptualmente diferentes; como un "offset del sistema de coordenadas global" o como un "offset del sistema de coordenadas local". El conjunto de comandos *G92* incluye:

- *G92*: este comando, cuando se usa con nombres de eje, establece valores para las variables de offset
- *G92.1*: este comando establece valores cero para las variables G92.
- *G92.2*: este comando suspende G92, pero no pone a cero las variables
- *G92.3*: este comando aplica los valores de offset que se suspendieron.

Como offset global, *G92* se usa para cambiar todas los sistemas de coordenadas de la pieza de trabajo, *G54* a *G59.3*. Un ejemplo de uso es cuando se mecanizan varias piezas idénticas en fijaciones con ubicaciones conocidas en un palet, pero la ubicación del palet puede cambiar entre lotes o entre máquinas. Cada offset de ubicación de la fijacion, relativo a un punto de referencia en el palet, está preestablecido en uno de los sistemas de coordenadas de pieza, de *G54* hasta *G59.3*, y *G92* se usa para "touch off" del punto de referencia en el palet. Luego, para cada parte, se selecciona el sistema de coordenadas de la pieza de trabajo correspondiente y se ejecuta el programa de pieza.

nota
La rotación del sistema de coordenadas de la pieza *G10 R-* es específica del intérprete *rs274ngc*, y el desplazamiento *G92* se aplica *después* de la rotación. Cuando se usa *G92* como offset global, las rotaciones del sistema de coordenadas de pieza pueden tener resultados inesperados.

Como sistema de coordenadas local, *G92* se usa como offset temporal dentro del sistema de coordenadas de la pieza de trabajo. Un ejemplo de uso es al mecanizar una pieza con varias características idénticas en diferentes ubicaciones. Para cada función, *G92* se usa para establecer un punto de referencia local, y se llama a un subprograma para mecanizar la característica a partir de ese punto.

nota
Se desaconseja el uso de *G92* para programar con sistemas de coordenadas locales en un programa de pieza. En su lugar, vea G52, un offset local del sistema de coordenadas es más intuitivo cuando se conoce el offset deseado relativo a la pieza de trabajo, pero es posible que no se conozca la ubicación actual de la herramienta.

La programación *G92 X0 Y0 Z0* establece la ubicación actual de la herramienta en coordina X0, Y0 y Z0, sin movimiento. G92 **no** funciona desde coordenadas absolutas de la máquina. Funciona desde **ubicación actual**.

G92 también funciona desde la ubicación actual modificada por cualquier otro offset que esté vigente cuando se invoca *G92*. Mientras se testeaban las diferencias entre los offsets de trabajo y los actuales se encontró que un offset *G54* podría cancelar un *G92* y, por lo tanto, parecia que no habia offsets en vigor. Sin embargo, *G92* estaba todavía vigente para todas las coordenadas y produjo los offsets de trabajo esperados para los otros sistemas de coordenadas.

Por defecto, los offsets *G92* se restauran después de que se inicia la máquina. Los programadores que deseen un comportamiento tipo Fanuc, donde los offsets *G92* se borran al inicio de la máquina y después de un reinicio o finalización del programa, puede deshabilitar la persistencia *G92* configurando *DISABLE_G92_PERSISTENCE = 1* en el Sección *[RS274NGC]* del archivo *.ini*.

nota
Es una buena práctica eliminar los offsets *G92* al final de su uso. con *G92.1* o *G92.2*. Al iniciar LinuxCNC con persistencia *G92* habilitada (el valor predeterminado), se aplicará cualquier offset en las variables *G92* cuando un eje tenga home. Ver G92 Precauciones con Persistencia a continuación.

5.1.4.3. Configuración de valores G92

Los comandos G92 funcionan desde la ubicación actual del eje y suman y restan correctamente para dar a la posición actual del eje el valor asignado por el comando G92. Los efectos funcionan a pesar de que haya offsets anteriores.

Por tanto, si el eje X muestra actualmente 2.0000 como su posición, un *G92 X0* establecerá un offset de -2.0000 para que la ubicación actual de X se convierta cero. Un *G92 X2* establecerá un offset de 0.0000 y la posición mostrada no cambiará. Un *G92 X5.0000* establecerá un offset de 3.0000 para que la posición actual visualizada se convierte en 5.0000.

5.1.4.4. Precauciones de Persistencia G92

Por defecto, los valores de un desplazamiento *G92* se guardarán en el archivo VAR y se restaurará después de un inicio o reinicio de la máquina.

Los parámetros G92 son:

- 5210 - Activar/desactivar bandera (1.0 / 0.0)
- 5211 - Offset eje X
- 5212 - Offset eje Y
- 5213 - Offset eje Z
- 5214 - Offset eje A
- 5215 - Offset eje B
- 5216 - Offset eje C
- 5217 - Offset eje U
- 5218 - Offset eje V
- 5219 - Offset eje W

donde 5210 es la bandera de habilitación *G92* (1 para habilitado, 0 para deshabilitado) y 5211 a 5219 son los offsets de eje. Si se ven posiciones inesperadas como resultado de un movimiento ordenado, resultado de almacenar un offset en un programa anterior y no borrarlos al final, entonces emita un G92.1 en la ventana MDI para borrar los offsets almacenados.

Si existen valores G92 en el archivo VAR cuando se inicia LinuxCNC, los valores en el archivo var se aplicarán a los valores de la ubicación actual de cada eje. Si esta es la posición home y la posición home esta establecida como cero máquina, todo será correcto. Una vez que home ha sido establecido usando interruptores de máquina reales, o moviendo cada eje a una posición

inicial conocida y emitiendo un comando de home del eje, cualquier desplazamiento G92 será aplicado. Si tiene un G92 X1 en efecto cuando da home al eje X, el DRO leerá *X: 1.000* en lugar del esperado *X: 0.000* porque el G92 se aplicó al origen de máquina. Si emite un G92.1 y el DRO ahora lee todos los ceros, entonces tuvo un desplazamiento G92 vigente la última vez corrió LinuxCNC.

A menos que su intención sea usar los mismas offsets G92 en el próximo programa, la mejor práctica es emitir un G92.1 al final de cualquier archivos de código G donde utiliza offsets G92.

Cuando un programa se aborta durante el procesamiento y tiene offsets *G92* en efecto, el inicio hará que se activen nuevamente. Como salvaguarda, tenga siempre su preámbulo estableciendo el entorno como usted espera. Además, la persistencia *G92* puede deshabilitarse configurando *DISABLE_G92_PERSISTENCE = 1* en la sección *[RS274NGC]* del archivo *.ini*.

5.1.4.5. Precauciones de Interacción G92 y G52

G52 y *G92* comparten los mismos registros de desplazamiento. A menos que la persistencia *G92* está deshabilitada en el archivo *.ini* (vea Comandos G92), los offsets *G52* también persistirán después del reinicio de la máquina, *M02* o *M30*. Tenga en cuenta que un offset *G52* en efecto durante un programa abortado puede dar lugar a desplazamientos no deseados cuando se ejecuta el siguiente programa. Ver G92 Precauciones con Persistence más arriba.

5.1.5. Programas de Muestra usando Offsets

5.1.5.1. Programa de Muestra utilizando Offsets de Coordenadas de Pieza

Este proyecto de grabado de muestra moldea un conjunto de cuatro círculos de radio .1 en una forma aproximadamente de estrella alrededor de un círculo central. Podemos configurar el patrón de círculo individual como este.

---------- ---------- G10 L2 P1 X0 Y0 Z0 (asegúrese de que G54 esté configurado en la máquina cero) G0 X-0.1 Y0 Z0 G1 F1 Z-0.25 G3 X-0.1 Y0 I0.1 J0 G0 Z0 M2 ---------- ----------

Podemos emitir un conjunto de comandos para crear offsets para los otros cuatro círculos, como esto:

---------- --------- G10 L2 P2 X0.5 (compensa el valor de G55 X en 0,5 pulgadas) G10 L2 P3 X-0.5 (compensa el valor de G56 X en -0.5 pulgadas) G10 L2 P4 Y0.5 (compensa el valor G57 Y en 0.5 pulgadas) G10 L2 P5 Y-0.5 (compensa el valor G58 Y en -0.5 pulgadas) ---------- ---------

Los reunimos en el siguiente programa:

---------- ---------- (un programa para fresar cinco círculos pequeños en forma de diamante)

G10 L2 P1 X0 Y0 Z0 (asegúrese de que G54 sea la máquina cero) G10 L2 P2 X0.5 (compensa el valor de G55 X en 0,5 pulgadas) G10 L2 P3 X-0.5 (compensa el valor de G56 X en -0.5 pulgadas) G10 L2 P4 Y0.5 (compensa el valor G57 Y en 0.5 pulgadas) G10 L2 P5 Y-0.5 (compensa el valor G58 Y en -0.5 pulgadas)

G54 G0 X-0.1 Y0 Z0 (círculo central) G1 F1 Z-0.25 G3 X-0.1 Y0 I0.1 J0 G0 Z0

G55 G0 X-0.1 Y0 Z0 (offset primer círculo) G1 F1 Z-0.25 G3 X-0.1 Y0 I0.1 J0 G0 Z0

G56 G0 X-0.1 Y0 Z0 (offset segundo círculo) G1 F1 Z-0.25 G3 X-0.1 Y0 I0.1 J0 G0 Z0

G57 G0 X-0.1 Y0 Z0 (offset tercer círculo) G1 F1 Z-0.25 G3 X-0.1 Y0 I0.1 J0 G0 Z0

G58 G0 X-0.1 Y0 Z0 (offset cuarto círculo) G1 F1 Z-0.25 G3 X-0.1 Y0 I0.1 J0 G54 G0 X0 Y0 Z0

M2 ---------- ----------

Ahora llega el momento en que podríamos aplicar un conjunto de offsets G92 a este programa. Verá que se está ejecutando en cada caso en Z0. Si la fresa estaban en la posición cero, un G92 Z1.0000 emitido al inicio del programa cambiaría todo una pulgada. También puede cambiar todo el patrón en el plano XY agregando algunos desplazamientos X e Y con G92. Si hace esto, debe agregar un comando G92.1 justo antes de M2 que finaliza el programa. Si no lo hace, otros programas que podría ejecutar después de este también usará ese desplazamiento G92. Además, lo harían en un nuevo inicio ya que se guardan los valores de G92 cuando se cierra LinuxCNC y serán recargados cuando se inicia de nuevo.

5.1.5.2. Programa de muestra usando offsets G52

(Para ser escrito)

:lang:es

5.2. Códigos G

:ini:' *:hal:*' :ngc:"

5.2.1. Convenciones

Convenciones utilizadas en esta sección

En los prototipos de código G, un guión (-) representa un valor real y (<>) denota un elemento opcional.

Si *L-* está escrito en un prototipo, a menudo se hará referencia a - como el *número L*, y así sucesivamente para cualquier otra letra.

En los prototipos de código G, la palabra *ejes* representa cualquier eje como se definió en su configuración

Un valor opcional se escribirá como esto <*L-*>.

Un valor real puede ser:

- Un número explícito, *4*
- Una expresión, *[2 + 2]*
- Un valor de parámetro, *# 88*
- Un valor de función unaria, *acos [0]*

En la mayoría de los casos, si se dan palabras de *eje* (cualquiera o todos de *X Y Z A B C U V W*), especificarán un punto de destino.

Los números de eje están en el sistema de coordenadas actualmente activo, a menos que se describa explícitamente como en el sistema de coordenadas absoluto.

Cuando las palabras de eje son opcionales, cualquier eje omitido conservará su valor original.

Cualquier elemento en los prototipos de código G no descrito explícitamente como opcional es obligatorio.

Los valores que siguen a las letras a menudo se dan como números explícitos. A menos que se indique lo contrario, los números explícitos pueden ser valores reales. Por ejemplo, *G10 L2* podría igualmente escribirse *G[2*5] L[1+1]*. Si el el valor del parámetro 100 era 2, *G10 L#100* también significaría lo mismo.

5.2.2. Tabla de referencia rápida de código G

Código	Descripción
G0	Movimiento coordinado a velocidad rápida
G1	Movimiento coordinado a velocidad de avance
G2 G3	Movimiento helicoidal coordinado a velocidad de avance
G4	Dwell
G5	Spline cúbico
G5.1	B-Spline cuadrático
G5.2	NURBS, agregar punto de control
G7	Modo de diámetro (torno)
G8	Modo de radio (torno)

Código	Descripción
G10 L1	Establecer entrada de tabla de herramientas
G10 L10	Establecer tabla de herramientas, calculada, pieza de trabajo
G10 L11	Establecer tabla de herramientas, calculada, fijación
G10 L2	Configuración de origen del sistema de coordenadas
G10 L20	Configuración del origen del sistema de coordenadas calculada
G17 - G19.1	Seleccionar Plano
G20 G21	Establecer unidades de medida
G28 - G28.1	Ir a la posición predefinida
G30 - G30.1	Ir a la posición predefinida
G33	Movimiento sincronizado del husillo
G33.1	Roscado rigido
G38.2 - G38.5	Sondeo
G40	Cancelar la compensación del cortador
G41 G42	Compensación del cortador
G41.1 G42.1	Compensación dinámica del cortador
G43	Usar el offset de longitud de herramienta de la tabla de herramientas
G43.1	Offset dinámico de longitudde herramienta
G43.2	Aplicar offset de longitud de herramienta adicional
G49	Cancelar offset de longitud de herramienta
G52	Offset del sistema de coordenadas local
G53	Mover en coordenadas de máquina
G54-G59.3	Seleccionar sistema de coordenadas (1 - 9)
G61	Modo de ruta exacta
G61.1	Modo de parada exacta
G64	Modo de control de ruta con tolerancia opcional
G70	Ciclo de acabado del torno (2.9)
G71-G72	Ciclo de desbaste del torno (2.9)
G73	Ciclo de perforación con rotura de viruta
G74	Ciclo de roscado izquierdo con Dwell
G76	Ciclo de roscado de múltiples pasadas (Torno)
G80	Cancelar modos de movimiento
G81	Ciclo de perforación
G82	Ciclo de perforación con Dwell
G83	Ciclo de perforación con picado
G84	Ciclo de roscado derecho con Dwell
G85	Ciclo de mandrinado, sin Dwell, salida a alimentacion
G86	Ciclo de mandrinado, con Dwell, salida rápida
G89	Ciclo de perforación, Dwell, salida a alimentación
G90 G91	Modo distancia
G90.1 G91.1	Modo de distancia de arco
G92	Compensación del sistema de coordenadas
G92.1 G92.2	Cancelar compensaciones G92
G92.3	Restaurar compensaciones G92
G93 G94 G95	Modos de alimentación
G96	Modo de control del husillo
G98 G99	Modo de retracción del ciclo fijo Z

5.2.3. G0 Movimiento rápido

```
G0 ejes
```

Para movimiento rápido, programe *G0 ejes*, donde todas las palabras de eje son opcionales. *G0* es opcional si el modo de movimiento actual es *G0*. Esto producirá movimiento coordinado hacia el punto de destino a la velocidad máxima rápida (o más

lento). *G0* se usa típicamente como un movimiento de posicionamiento.

5.2.3.1. Velocidad rápida

La configuración MAX_VELOCITY en la sección del archivo ini [TRAJ] define la máxima velocidad de offset rápido. La velocidad máxima de offset rápido puede ser mayor que la configuración de MAX_VELOCITY de ejes individuales durante un movimiento coordinado. La maxima velocidad de offset rápido puede ser más lenta que la configuración MAX_VELOCITY en [TRAJ] si MAX_VELOCITY de un eje o restricciones de trayectoria la limitan.

Ejemplo G0

```
G90 (establecer modo de distancia absoluta)
G0 X1 Y-2.3 (movimiento lineal rápido desde la ubicación actual a X1 Y-2.3)
M2 (final del programa)
```

- Consulte las secciones G90 y M2 para obtener más información.

Si la compensación del cortador está activa, el movimiento será diferente de lo anterior; vea la sección Compensación del cortador.

Si *G53* está programado en la misma línea, el movimiento también será diferente; Consulte la sección G53 para obtener más información.

La trayectoria de un movimiento rápido G0 se puede redondear en los cambios de dirección y depende del control de trayectoria y de la máxima aceleración de los ejes.

Es un error si:

- Una letra de eje no tiene un valor real.
- Se utiliza una letra de eje que no está configurada

5.2.4. G1 Movimiento lineal

```
G1 ejes
```

Para movimiento lineal (línea recta) a velocidad de avance programada (para cortar o no), programe *G1 'ejes'*, donde todas las palabras de eje son opcionales. *G1* es opcional si el modo de movimiento actual es *G1*. Esto produce movimiento coordinado al punto de destino a la tasa de alimentación actual (o más lenta).

Ejemplo G1

```
G90 (establecer modo de distancia absoluta)
G1 X1.2 Y-3 F10 (movimiento lineal a velocidad de alimentación de 10 desde la posición ↩
    actual a X1.2 Y-3)
Z-2.3 (movimiento lineal a la misma velocidad de avance desde la posición actual a Z-2.3)
Z1 F25 (movimiento lineal a una velocidad de avance de 25 desde la posición actual a Z1)
M2 (final del programa)
```

- Consulte las secciones G90 , F y M2 para más información.

Si la compensación del cortador está activa, el movimiento será diferente de lo anterior; vea la sección Compensación del cortador.

Si *G53* está programado en la misma línea, el movimiento también será diferente; Consulte la sección G53 para obtener más información.

Es un error si:

- No se ha establecido la velocidad de alimentación.
- Una letra de eje no tiene un valor real.
- Se utiliza una letra de eje que no está configurada

5.2.5. G2, G3 Movimiento de arco

```
G2 o G3 offsets de ejes (formato centro)
G2 o G3 ejes R- (formato radio)
G2 o G3 offsets|R- <P-> (círculos completos)
```

Un arco circular o helicoidal se especifica utilizando *G2* (arco en sentido horario) o *G3* (arco en sentido antihorario) a la velocidad de avance actual. La dirección (CW, CCW) se ve desde extremo positivo del eje sobre el cual ocurre el movimiento circular.

El eje del círculo o hélice debe ser paralelo al eje X, Y o Z del sistema de coordenadas de máquina. El eje (o, equivalentemente, el plano perpendicular al eje) se selecciona con *G17* (eje Z, plano XY), *G18* (eje Y, plano XZ) o *G19* (eje X, plano YZ). Los planos *17.1 ', '18.1* y '19.1 'no son compatibles actualmente. Si el arco es circular, se encuentra en un plano paralelo al plano seleccionado.

Para programar una hélice, incluya la palabra del eje perpendicular al arco plano, por ejemplo, si está en el plano *G17*, incluya una palabra *Z*. Esta hará que el eje *Z* se mueva al valor programado durante el movimiento circular *XY*.

Para programar un arco que dé más de una vuelta completa, use la palabra *P* especificando el número de vueltas completas más el arco programado. La palabra *P* debe ser un entero. Si *P* no está especificado, el comportamiento es como si fuera dado *P1* que solo dará una vuelta completa o parcial. Por ejemplo, si un arco de 180 grados se programa con P2, el movimiento resultante será 1 1/2 rotaciones. Para cada incremento de P por encima de 1, se agrega un círculo completo adicional al arco programado. Se admiten arcos helicoidales de múltiples vueltas, que dan movimiento útil para fresar agujeros o roscas.

Si una línea de código forma un arco e incluye un movimiento de eje giratorio, los ejes rotativos giran a una velocidad constante para que el movimiento rotativo comienze y termine cuando el movimiento XYZ comienza y termina. Las líneas de este tipo casi nunca se programan.

Si la compensación del cortador está activa, el movimiento será diferente de lo anterior; vea la sección Compensación del cortador.

El centro del arco es absoluto o relativo según lo establecido por G90.1 o G91.1 respectivamente.

Se permiten dos formatos para especificar un arco: formato centro y formato radio.

Es un error si:

- No se ha establecido la velocidad de alimentación.
- La palabra P no es un número entero.

5.2.5.1. Arcos de formato centro

Los arcos de formato centro son más precisos que los arcos de formato radio y es el formato preferido.

El punto final del arco, junto con el offset al centro del arco desde la ubicación actual se usa para programar arcos que son menos que un círculo completo. Está bien si el punto final del arco es el mismo que la ubicación actual

Para programar círculos completos se usa el offset al centro del arco desde la ubicación actual y, opcionalmente, el número de vueltas .

Al programar arcos, se puede producir un error debido al redondeo al usar una precisión de menos de 4 decimales (0.0000) para pulgadas y menos de 3 decimales (0.000) para milímetros.

Modo de distancia de arco incremental Los offsets del centro del arco son una distancia relativa de la ubicación de inicio del arco. El modo de distancia de arco incremental es el predeterminado.

Se deben programar una o más palabras de eje y uno o más offsets para un arco de menos de 360 grados.

No se deben programar palabras de eje y uno o más offsets para círculos completos. La palabra *P* está por defecto en 1 y es opcional.

Para obtener más información sobre el 'Modo de distancia de arco incremental, consulte la sección G91.1 .

Modo de distancia de arco absoluta. Los offsets del centro del arco son la distancia absoluta desde la posición 0 actual del eje.

Una o más palabras de eje y *ambos* offsets deben programarse para arcos de menos de 360 grados

No se deben programar palabras de eje y ambos offsets para círculos completos. La palabra *P* está por defecto en 1 y es opcional.

Para obtener más información sobre el 'Modo de distancia absoluta del arco, consulte la sección G90.1.

Plano XY (G17)

```
G2 o G3 <X- Y- Z- I- J- P->
```

- *Z* - hélice
- *I* - offset X
- *J* - offset Y
- *P* - número de vueltas

Plano XZ (G18)

```
G2 o G3 <X- Z- Y- I- K- P->
```

- *Y* - hélice
- *I* - offset X
- *K* - offset Z
- *P* - número de vueltas

Plano YZ (G19)

```
G2 o G3 <Y- Z- X- J- K- P->
```

- *X* - hélice
- *J* - offset Y
- *K* - offset Z
- *P* - número de vueltas

Es un error si:

- No se establece la velocidad de avance con la palabra F.
- No hay offsets programados.
- Cuando el arco se proyecta en el plano seleccionado, la distancia desde el punto actual al centro difiere de la distancia desde el punto final al centro en más de (.05 pulgadas / .5 mm) o ((.0005 pulgadas / .005 mm) y .1% del radio).

Descifrando el mensaje de error *El radio final del arco difiere del radio inicial:*

- *inicio* - la posición actual
- *centro* - la posición central calculada con las palabras i, j o k
- *fin* - el punto final programado
- *r1* - radio desde la posición inicial hasta el centro
- *r2* - radio desde la posición final hasta el centro

5.2.5.2. Ejemplos de formato centro

Calcular arcos a mano puede ser difícil a veces. Una opción es dibujar el arco con un programa cad para obtener las coordenadas y los offsets. Tenga en cuenta la tolerancia mencionada anteriormente; puede que tenga que cambiar la precisión de su programa cad para obtener los resultados deseados. Otra opción es calcular las coordenadas y el offset utilizando fórmulas. Como puede ver en las siguientes figuras, se puede formar un triángulo a partir de la posición actual, la posición final y el centro del arco.

En la siguiente figura puede ver que la posición de inicio es X0 Y0 y la posición final es X1 Y1. La posición central del arco está en X1 Y0. Esto da un offset desde la posición inicial de 1 en el eje X y 0 en el eje Y. En este caso solo se necesita un offset I.

G2 Ejemplo de línea

```
G0 X0 Y0
G2 X1 Y1 I1 F10 (arco en sentido horario en el plano XY)
```

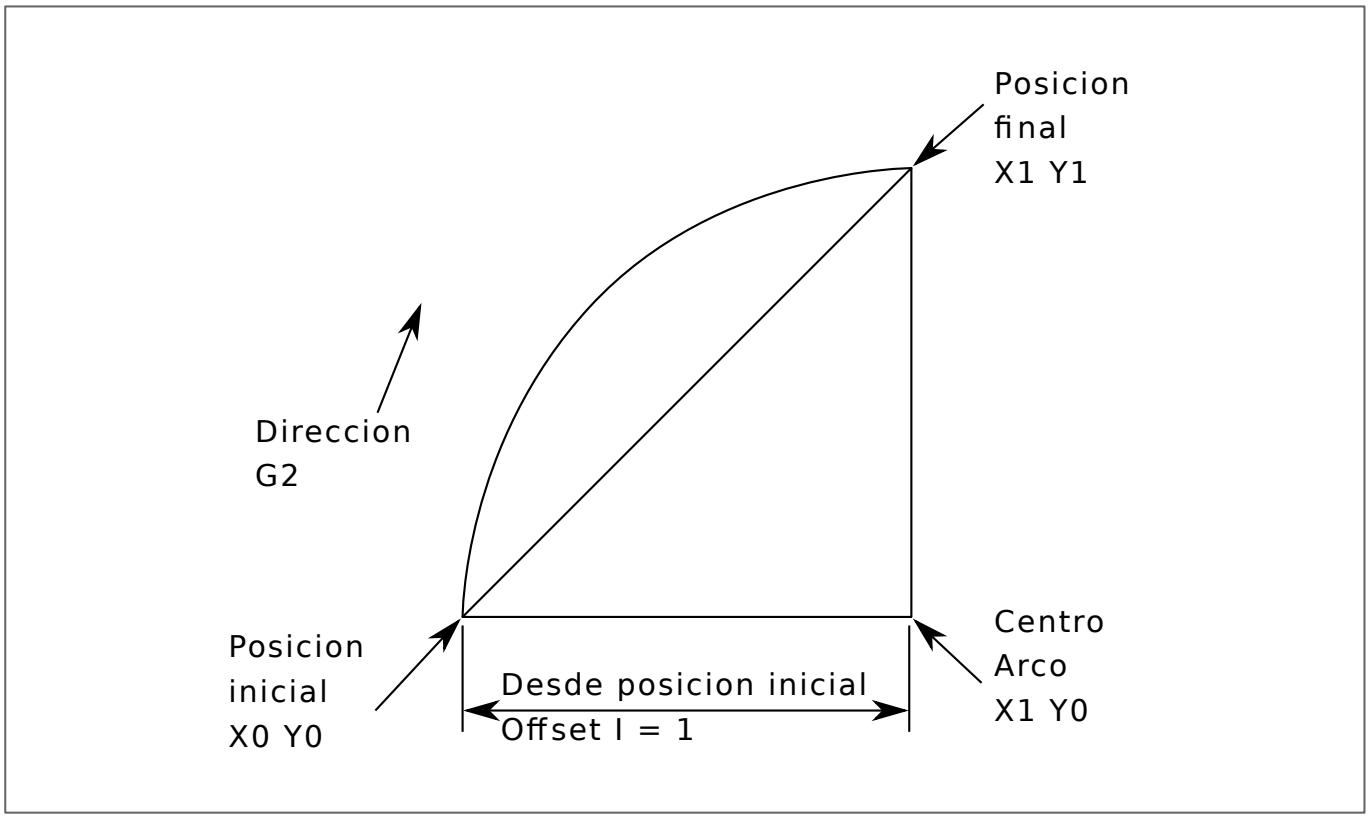


Figura 5.2: Ejemplo G2

En el siguiente ejemplo, vemos la diferencia entre los offsets para Y si estamos haciendo un movimiento G2 o G3. Para el movimiento G2, la posición inicial es X0 Y0, para el movimiento G3 es X0 Y1. El centro del arco está en X1 Y0.5 para ambos movimientos. Para G2, el offset J es 0.5 y para G3 el offset J es -0.5.

Lineas de ejemplo G2-G3

```
G0 X0 Y0
G2 X0 Y1 I1 J0.5 F25 (arco en sentido horario en el plano XY)
G3 X0 Y0 I1 J-0.5 F25 (arco en sentido antihorario en el plano XY)
```

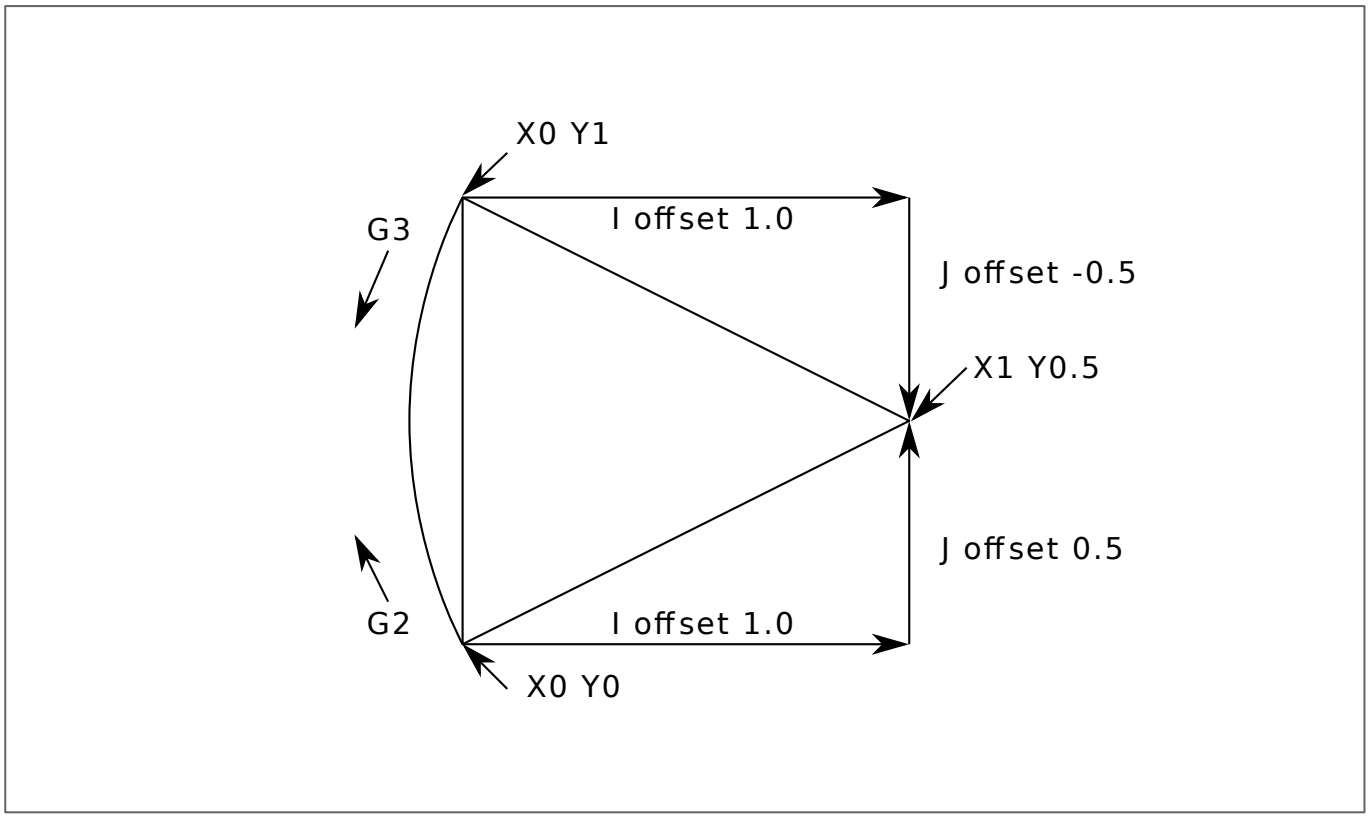


Figura 5.3: Ejemplo G2-G3

En el siguiente ejemplo mostramos cómo el arco puede hacer una hélice en el eje Z agregando la palabra Z.

Ejemplo Helice G2

```
G0 X0 Y0 Z0
G17 G2 X10 Y16 I3 J4 Z-1 (arco helicoidal con Z agregado)
```

En el siguiente ejemplo mostramos cómo hacer más de una vuelta usando la palabra P.

Ejemplo de palabra P

```
G0 X0 Y0 Z0
G2 X0 Y1 Z-1 I1 J0.5 P2 F25
```

En el formato centro, el radio del arco no está especificado, pero se puede encontrar fácilmente como la distancia desde el centro del círculo hasta el punto actual o el punto final del arco.

5.2.5.3. Arcos de formato radio

```
G2 o G3 ejes R- <P->
```

- *R* - radio desde la posición actual

No es una buena práctica programar arcos de formato de radio que sean círculos casi completos o casi semicírculos debido a que un pequeño cambio en la ubicación del punto final producirá un cambio mucho mayor en la ubicación del centro del círculo (y, por lo tanto, el centro del arco). El efecto de aumento es lo suficientemente grande como para que el error de redondeo en un número puede producir cortes fuera de tolerancia. Por ejemplo, un desplazamiento del 1 % del punto final de un arco de 180

grados produce un 7 % de desplazamiento de un punto a 90 grados a lo largo del arco. Los círculos casi completos son aún peores. Otros arcos de tamaño (en el rango de hasta 165 grados o entre 195 a 345 grados) están bien.

En el formato radio, las coordenadas del punto final del arco en el plano seleccionado se especifica junto con el radio del arco. Programe los ejes *G2 ejes '* R-*(o use 'G3* en lugar de *G2*). R es el radio. Las palabras de eje son todas opcionales, excepto que debe ser usada al menos una de las dos palabras para los ejes del plano seleccionado. El número R es el radio. Un radio positivo indica que el el arco gira menos de 180 grados, mientras que un radio negativo indica un giro de más de 180 grados. Si el arco es helicoidal, también se especifica el valor del punto final del arco en el eje de coordenadas paralelo al eje de la hélice.

Es un error si:

- se omiten las dos palabras de eje para los ejes del plano seleccionado
- el punto final del arco es el mismo que el punto actual.

G2 Ejemplo de línea

```
G17 G2 X10 Y15 R20 Z5 (formato de radio con arco)
```

El ejemplo anterior hace un arco circular o helicoidal en sentido horario (visto desde el eje Z positivo) cuyo eje es paralelo al eje Z, terminando en X = 10, Y = 15 y Z = 5, con un radio de 20. Si el valor inicial de Z es 5, este es un arco de un círculo paralelo al plano XY; de lo contrario es un arco helicoidal.

5.2.6. G4 Dwell

```
G4 P-
```

- *P* - segundos en parada (punto flotante)

El número P es el tiempo en segundos que todos los ejes permanecerán inmóviles. El número P es un número de coma flotante, por lo que se pueden usar fracciones de segundo. G4 no afecta al husillo, al refrigerante, ni a ninguna E/S.

Ejemplo de líneas G4

```
G4 P0.5 (espera de 0.5 segundos antes de continuar)
```

Es un error si:

- el número P es negativo o no está especificado.

5.2.7. G5 Spline cúbico

```
G5 X- Y- <I- J-> P- Q-
```

- *I* - offset incremental X desde el punto de inicio hasta el primer punto de control
- *J* - offset incremental Y desde el punto de inicio hasta el primer punto de control
- *P* - offset incremental X desde el punto final hasta el segundo punto de control
- *Q* - offset incremental Y desde el punto final hasta el segundo punto de control

G5 crea una B-spline cúbica en el plano XY con los ejes X e Y únicamente. P y Q deben especificarse para cada comando G5.

Para el primer comando G5 en una serie de comandos G5, I y J deben especificarse. Para los comandos G5 posteriores, tanto I como J deben especificarse ambos o ninguno. Si I y J no están especificados, la dirección inicial de este spline coincidirá automáticamente con la dirección final del anterior (como si I y J fueran la negación de P y Q del anterior)

Por ejemplo, para programar una forma de N con curvas:

G5 Muestra spline cúbica inicial

```
G90 G17
G0 X0 Y0
G5 I0 J3 P0 Q-3 X1 Y1
```

Ahora se puede hacer un segundo N curvilíneo que se adhiera suavemente a este sin especificar I y J:

G5 Muestra de spline cúbica posterior

```
G5 P0 Q-3 X2 Y2
```

Es un error si:

- P y Q no están especificados
- Solo se especifica uno de I o J
- I o J no están especificados en el primero de una serie de comandos G5
- Se especifica un eje distinto de X o Y
- El plano activo no es G17

5.2.8. G5.1 Spline cuadrático

```
G5.1 X- Y- I- J-
```

- *I* - offset incremental X desde el punto inicial al punto de control
- *J* - offset incremental Y desde el punto inicial al punto de control

G5.1 crea una B-spline cuadrática en el plano XY con los ejes X e Y solamente. No especificar I o J da un offset cero para el eje no especificado; por tanto, uno o ambos deben ser dados.

Por ejemplo, para programar una parábola, a través del origen, de X-2 Y4 a X2 Y4:

G5.1 Muestra de spline cuadrática

```
G90 G17
G0 X-2 Y4
G5.1 X2 I2 J-8
```

Es un error si:

- Tanto el offset I como J no están especificados o son cero
- Se especifica un eje distinto de X o Y
- El plano activo no es G17

5.2.9. G5.2 G5.3 Bloque NURBS

```
G5.2 <P-> <X- Y-> <L->
X- Y- <P->
...
G5.3
```

Advertencia: G5.2, G5.3 es experimental y no está completamente probado.

G5.2 abre el bloque de datos que define un NURBS y G5.3 lo cierra. En las líneas entre estos dos códigos, los puntos de control de la curva se definen con sus *pesos* (P) relacionados y el parámetro (L) que determina el orden de la curva.

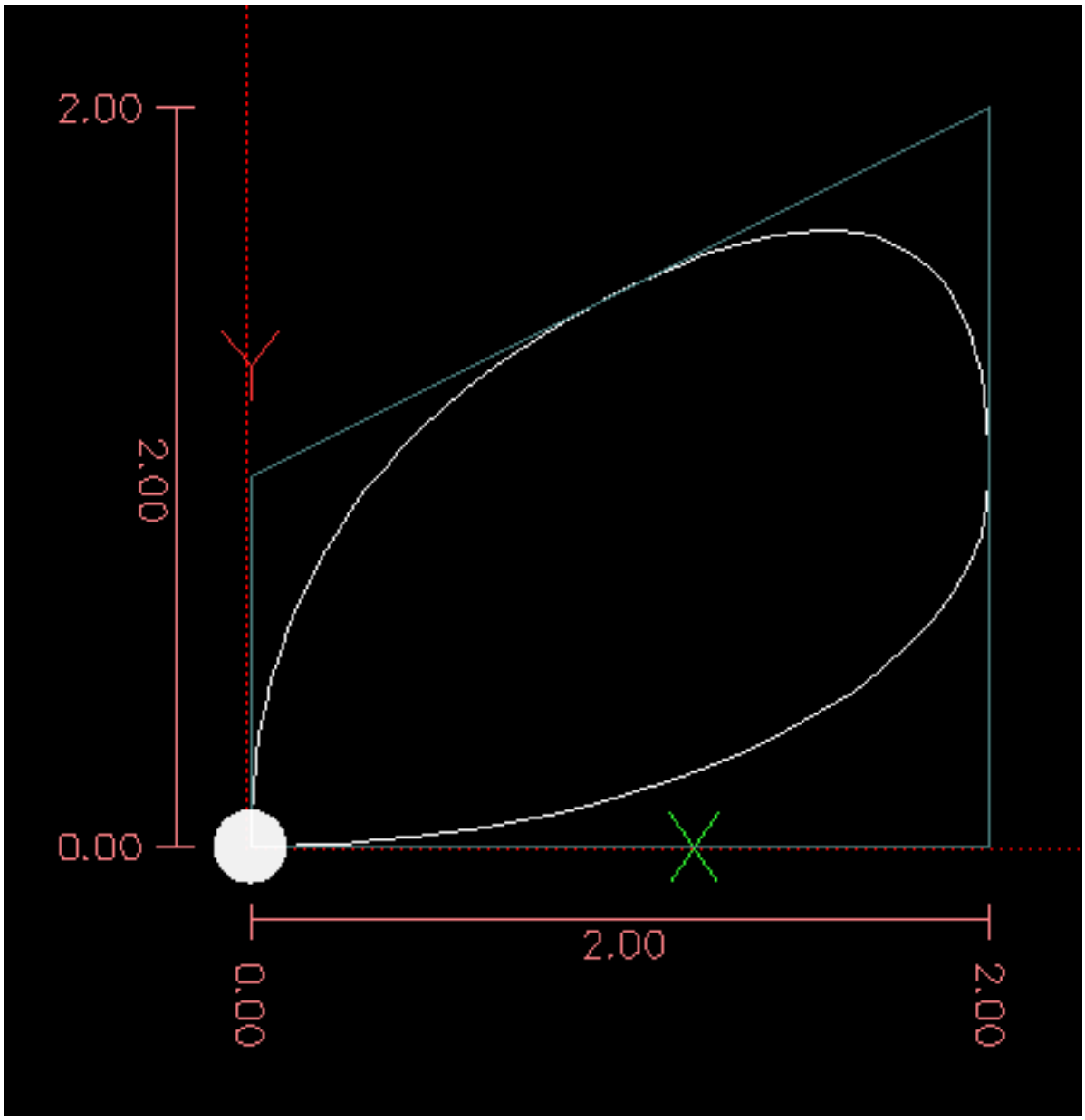
La coordenada actual, antes del primer comando G5.2, siempre se toma como el primer punto de control NURBS. Para establecer el peso para este primer punto de control, programe G5.2 P- sin dar ninguna X Y.

El peso predeterminado es 1. El orden predeterminado es 3.

G5.2 Ejemplo

```
G0 X0 Y0 (movimiento rápido)
F10 (velocidad de avance establecida)
G5.2 P1 L3
~~~~~X0 Y1 P1
~~~~~X2 Y2 P1
~~~~~X2 Y0 P1
~~~~~X0 Y0 P2
G5.3
; Los movimientos rápidos muestran el mismo camino sin el Bloque NURBS
G0 X0 Y1
~~~X2 Y2
~~~X2 Y0
~~~X0 Y0
M2
```

Salida de muestra NURBS



Puede encontrar más información sobre NURBS aquí:

http://wiki.linuxcnc.org/cgi-bin/wiki.pl?NURBS

5.2.10. Modo de diámetro de torno G7

```
G7
```

Programe G7 para ingresar el modo de diámetro para el eje X en un torno. En el modo de diámetro el eje X se mueve 1/2 de la distancia al centro del torno. Por ejemplo, X1 movería el cortador a 0.500" desde el centro del torno dando una parte de 1" de diámetro.

5.2.11. Modo de radio de torno G8

```
G8
```

Programe G8 para modo radio en el eje X de un torno. Cuando el eje X se mueva en modo radio en un torno será la distancia desde el centro. Por lo tanto, un corte en X1 daría como resultado una pieza de 2" de diámetro. G8 es el predeterminado al arranque.

5.2.12. G10 L1 Establecer tabla de herramientas

```
G10 L1 P- ejes <R- I- J- Q->
```

- *P* - número de herramienta
- *R* - radio de herramienta
- *I* - ángulo frontal (torno)
- *J* - ángulo trasero (torno)
- *Q* - orientación (torno)

G10 L1 establece la tabla de herramientas para el número de herramienta *P* a los valores de las palabras.

Un G10 L1 válido reescribe y vuelve a cargar la tabla de herramientas.

Ejemplo de línea G10 L1

```
G10 L1 P1 Z1.5 (ajustar la herramienta 1 con offset Z desde el origen de la máquina de 1.5)
G10 L1 P2 R0.015 Q3 (configuración de ejemplo de herramienta 2 de torno radio 0.015 y ↩
    orientación 3)
```

Es un error si:

- La compensación de cortador está activada
- El número P no está especificado
- El número P no es un número de herramienta válido de la tabla de herramientas
- El número P es 0

Para obtener más información sobre la orientación del cortador utilizada por la palabra *Q*, vea el diagrama Orientación Herramienta Torno.

5.2.13. G10 L2 Establecer sistema de coordenadas

```
G10 L2 P- <ejes R->
```

- *P* - sistema de coordenadas (0-9)
- *R* - rotación sobre el eje Z

G10 L2 desplaza el origen de los ejes en el sistema de coordenadas especificado el valor de cada palabra de eje. El offset es desde el origen de la máquina establecido durante el recorrido de homing. El valor de offset reemplazará cualquier offset actual vigente para el sistema de coordenadas especificado. Las palabras de eje no utilizadas no se cambiarán.

Programe P0 a P9 para especificar qué sistema de coordenadas cambiar.

Cuadro 5.1: Sistema coordinado

Valor P	Sistema de coordenadas	Código G
0	Activo	n/a
1	1	G54
2	2	G55
3	3	G56
4	4	G57
5	5	G58
6	6	G59
7	7	G59.1
8	8	G59.2
9	9	G59.3

Opcionalmente, programe R para indicar la rotación del eje XY alrededor del eje Z. La dirección de rotación es CCW vista desde el extremo positivo del eje Z.

Todas las palabras de eje son opcionales.

Estar en modo de distancia incremental (G91) no tiene efecto en *G10 L2*.

Conceptos importantes

- G10 L2 Pn no cambia del sistema de coordenadas actual al especificado por P; tiene que usar G54-59.3 para seleccionar un sistema de coordenadas.
- Cuando una rotación está activa, un eje solo permitirá jogging en una dirección positiva o negativa y no a lo largo del eje girado.
- Si un offset local *G52* o un offset de origen *G92* estaba vigente antes *G10 L2*, continuará vigente después.
- Al programar un sistema de coordenadas con R, cualquier *G52* o *G92* se aplicará **después** de la rotación.
- El sistema de coordenadas cuyo origen se establece mediante un comando *G10* puede estar activo o inactivo en el momento en que se ejecuta *G10*. Si está activo, las nuevas coordenadas surten efecto de inmediato.

Es un error si:

- El número P no se evalúa como un entero en el rango de 0 a 9.
- Se programa un eje que no está definido en la configuración.

Ejemplo de línea G10 L2

```
G10 L2 P1 X3.5 Y17.2
```

En el ejemplo anterior, el origen del primer sistema de coordenadas (el seleccionado por *G54*) se configura para ser X = 3.5 e Y = 17.2. Como solo se especifican X e Y, el punto de origen solo se mueve en X e Y; las otras coordenadas no cambian.

Ejemplo de línea G10 L2

```
G10 L2 P1 X0 Y0 Z0 (limpiar offsets para los ejes X, Y y Z en el sistema de coordenadas 1)
```

El ejemplo anterior establece las coordenadas XYZ del sistema de coordenadas 1 en el origen de máquina.

El sistema de coordenadas se describe en la sección Sistema de coordenadas.

5.2.14. G10 L10 Establecer tabla de herramientas

```
G10 L10 ejes P <R- I- J- Q->
```

- *P* - número de herramienta
- *R* - radio de herramienta
- *I* - ángulo frontal (torno)
- *J* - ángulo trasero (torno)
- *Q* - orientación (torno)

G10 L10 cambia la entrada de la tabla de herramientas para la herramienta P de modo que si la compensación de herramienta se vuelve a cargar, con la máquina en su posición actual y con las compensaciones actuales G5x y G52/G92 activas, las coordenadas actuales para los ejes dados se convertirán en los valores dados. Los ejes que no se han especificado en el comando G10 L10 no se cambiarán. Esto podría ser útil con un movimiento de sonda como se describe en la sección G38.

Ejemplo G10 L10

```
T1 M6 G43 (carga de herramienta 1 y compensaciones de longitud)
G10 L10 P1 Z1.5 (establecer la posición actual para Z en 1.5)
G43 (vuelve a cargar las compensaciones de longitud de herramienta de la tabla de ↩
    herramientas modificada)
M2 (final del programa)
```

- Consulte las secciones T , M6 y G43/G43.1 para más información.

Es un error si:

- La compensación de cortador está activada
- El número P no está especificado
- El número P no es un número de herramienta válido de la tabla de herramientas
- El número P es 0

5.2.15. G10 L11 Establecer tabla de herramientas

```
G10 L11 P- ejes <R- I- J- Q->
```

- *P* - número de herramienta
- *R* - radio de herramienta
- *I* - ángulo frontal (torno)
- *J* - ángulo trasero (torno)
- *Q* - orientación (torno)

G10 L11 es como G10 L10, excepto que en lugar de configurar la entrada de acuerdo con las compensaciones actuales, se establece de modo que las coordenadas actuales se convertirían en el valor dado si se vuelve a cargar el offset de la nueva herramienta y la máquina se coloca en el sistema de coordenadas G59.3 sin ningún offset G52/G92 activo.

Esto permite al usuario configurar el sistema de coordenadas G59.3 de acuerdo con un punto fijo en la máquina, y luego usar ese lugar para medir herramientas sin tener en cuenta otras compensaciones actualmente activas.

Es un error si:

- La compensación de cortador está activada
- El número P no está especificado
- El número P no es un número de herramienta válido de la tabla de herramientas
- El número P es 0

5.2.16. G10 L20 Establecer sistema de coordenadas

```
G10 L20 P- ejes
```

- *P* - sistema de coordenadas (0-9)

G10 L20 es similar a G10 L2, excepto que en lugar de configurar el offset/entrada al valor dado, se establece en un valor calculado que hace que las coordenadas actuales se conviertan en el valor dado.

Línea de ejemplo G10 L20

```
G10 L20 P1 X1.5 (establece la ubicación actual del eje X en el sistema de coordenadas 1 a ↩
    1.5)
```

Es un error si:

- El número P no se evalúa como un entero en el rango de 0 a 9.
- Se programa un eje que no está definido en la configuración.

5.2.17. G17 - G19.1 Seleccionar plano

Estos códigos establecen el plano actual de la siguiente manera:

- *G17* - XY (predeterminado)
- *G18* - ZX
- *G19* - YZ
- *G17.1* - UV
- *G18.1* - WU
- *G19.1* - VW

Los planos UV, WU y VW no admiten arcos.

Es una buena práctica incluir una selección de plano en el preámbulo de cada archivo de código G.

Los efectos de tener un plano seleccionado se discuten en las Secciones G2 G3 Arcos y G81 G89

5.2.18. Unidades G20, G21

- *G20* - usar pulgadas para unidades de longitud.
- *G21* - usar milímetros para unidades de longitud.

Es una buena práctica incluir selección de unidades en el preámbulo de cada archivo de código G.

5.2.19. G28, G28.1 Ir/Establecer posición predefinida

aviso
Solo use G28 cuando su máquina esté en una posición repetible y la posición G28 deseada se ha almacenado con G28.1.

G28 usa los valores almacenados en los parámetros 5161-5169 como el punto final X Y Z A B C U V W a donde moverse. Los valores de los parametros son coordenadas máquina *absolutas* en las *unidades* de máquina nativas especificadas en el archivo ini. Todos los ejes definidos en el archivo ini se moverán cuando se emite un G28. Si no se almacenan posiciones con G28.1, todos los ejes irán al origen de máquina.

- *G28* - hace un movimiento rápido desde ls actual posición a la posición *absoluta* de los valores en los parámetros 5161-5166.
- *G28 ejes* - hace un movimiento rápido a la posición especificada por *ejes* incluyendo cualquier offset, luego hará un movimiento rápido hacia la posición *absoluta* de los valores en los parámetros 5161-5166 para todos los *ejes* especificados. Ningun *eje* no especificado no se moverá.
- *G28.1* - almacena la posición *absoluta* actual en los parámetros 5161-5166.

Linea de ejemplo G28

```
G28 Z2.5 (rápido a Z2.5 y luego a la ubicación Z especificada en #5163)
```

Es un error si:

- La compensación del cortador está activada

5.2.20. G30, G30.1 Ir/Establecer posición predefinida

aviso
Use G30 solo cuando su máquina esté en una posición repetible y la posición G30 deseada se ha almacenado con G30.1.

G30 funciona igual que G28 pero usa los valores almacenados en los parámetros 5181-5189 como punto final X Y Z A B C U V W a donde moverse. Los valores de los parámetros son coordenadas máquina *absolutas* en las *unidades* de máquina nativas especificadas en el archivo ini. Todos los ejes definidos en el archivo ini se moverán cuando se emita un G30. Si no se almacenan las posiciones con G30.1, todos los ejes irán al origen máquina.

nota
Los parámetros G30 se usarán para mover la herramienta cuando se programe un M6 si TOOL_CHANGE_AT_G30 = 1 está en la sección [EMCIO] del archivo ini.

- *G30* - hace un movimiento rápido desde la posición actual a la posición "absoluta" de los valores en los parámetros 5181-5189.
- *G30 ejes*:realiza un movimiento rápido a la posición especificada por *ejes* incluyendo cualquier offset, luego hará un movimiento rápido a la posición *absoluta* de los valores en los parámetros 5181-5189 para todos los *ejes* especificados. Cualquier *eje* no especificado no se moverá.
- *G30.1* - almacena la posición absoluta actual en los parámetros 5181-5186.

G30 Ejemplo de línea

```
G30 Z2.5 (rápido a Z2.5 y luego a la ubicación Z especificada en #5183)
```

Es un error si:

- La compensación del cortador está activada

5.2.21. Movimiento sincronizado del husillo G33

```
G33 X- Y- Z- K- $ -
```

- *K* - distancia por revolución

Para movimiento sincronizado con husillo en una dirección, codifique *G33 X- Y- Z- K-* donde K es la distancia movida en XYZ para cada revolución del husillo. Por ejemplo, si comienza en *Z = 0*, *G33 Z-1 K.0625* produce un movimiento de 1 pulgada en Z cada 16 revoluciones del husillo. Este comando podría ser parte de un programa para producir una rosca 16TPI. Otro ejemplo en métrica, *G33 Z-15 K1.5* produce un movimiento de 15 mm mientras el husillo gira 10 veces para una rosca de 1,5 mm.

El argumento $ (opcional) establece qué husillo se sincroniza con el movimiento (el valor predeterminado es cero). Por ejemplo, G33 Z10 K1 $1 moverá el eje en sincronía con el valor del pin HAL spindle.N.revs.

El movimiento sincronizado con el husillo espera los pines index y husillo-a-velocidad, por lo que se alinean múltiples pases. *G33* mueve el extremo al punto final programado. G33 podría usarse para cortar roscados cónicos.

Todas las palabras del eje son opcionales, pero se debe utilizar al menos una.

---

nota
K sigue la línea descrita por *X- Y- Z-*. K no es paralelo a el eje Z si se utilizan puntos finales X o Y, por ejemplo, al cortar roscas cónicas.

Información técnica Al comienzo de cada pasada G33, LinuxCNC usa la velocidad del eje y los límites de aceleración de la máquina para calcular cuánto tiempo llevará acelerar Z después del pulso índice y determina cuántos grados rotará el husillo durante ese tiempo. Luego agrega ese ángulo a la la posición index y calcula la posición Z utilizando el angulo de husillo corregido. Eso significa que Z alcanzará la posición correcta justo al terminar de acelerar a la velocidad adecuada y puede comenzar de inmediato cortando un buen hilo.

Conexiones HAL El pin *spindle.N.at-speed* debe estar configurado o accionado como true para que el movimiento comience. Además spindle.N.revs debe aumentar en 1 por cada revolución del husillo y elpin spindle.N.index-enable debe estar conectado a un contador de codificador (o resolver) que restablece la habilitación de índice una vez por rev.

Consulte el Manual de integradores para obtener más información sobre el movimiento sincronizado del husillo.

Ejemplo G33

```
G90 (modo de distancia absoluta)
G0 X1 Z0.1 (rápido a la posición)
S100 M3 (comenzar a girar el husillo)
G33 Z-2 K0.125 (mover el eje Z a -2 a una velocidad igual a 0.125 por revolución)
G0 X1.25 (movimiento rápido herramienta fuera del trabajo)
Z0.1 (movimiento rápido a la posición Z inicial)
M2 (final del programa)
```

- Consulte las secciones G90 , G0 y M2 para obtener más información.

Es un error si:

- Todas las palabras del eje se omiten.
- El husillo no gira cuando se ejecuta este comando
- El movimiento lineal solicitado excede los límites de velocidad de la máquina debido a la velocidad del husillo

5.2.22. G33.1 Roscado rígido

```
G33.1 X- Y- Z- K- I- $ -
```

- *K* - distancia por revolución
- *I* - multiplicador de velocidad del husillo opcional para un movimiento de retorno más rápido
- *$* - selector de husillo opcional

aviso
Para roscado solo en Z preposicionar la ubicación XY antes de llamar a G33.1 y solo use una palabra Z en G33.1. Si las coordenadas especificadas no son las actuales, al llamar a G33.1 el movimiento no será a lo largo del eje Z sino un movimiento coordinado y sincronizado con el husillo desde la ubicación actual a la ubicación especificada y viceversa.

Para roscado rígido (movimiento sincronizado del husillo con retorno), codifique *G33.1 X- Y- Z- K-* donde *K-* es la distancia recorrida por cada revolución del husillo.

Un movimiento de roscado rígido consiste en la siguiente secuencia:

1. Un movimiento desde la coordenada actual a la coordenada especificada, sincronizada con el husillo seleccionado en la proporción dada y comenzando desde la coordenada actual tras un pulso de índice del husillo.
2. Al llegar al punto final, comanda la inversion del eje y acelera por un factor establecido por el multiplicador (p. ej., de derecha a izquierda).
3. Movimiento sincronizado continuo más allá de la coordenada final especificada hasta que el husillo realmente se detenga y se invierta.
4. Movimiento sincronizado continuo de vuelta a la coordenada original.
5. Al alcanzar la coordenada original, comanda para invertir el eje por segunda vez (por ejemplo, de izquierda a derecha).
6. Movimiento sincronizado continuo más allá de la coordenada original hasta que el husillo realmente se detenga y se invierta.
7. Un movimiento **no sincronizado** retrocediendo a la coordenada original.

Los movimientos sincronizados con el husillo esperan el índice del husillo, así que se alinean múltiples pases. Los movimientos *G33.1* terminan en la coordenada original.

Todas las palabras del eje son opcionales, pero debe utilizar al menos una.

Ejemplo G33.1

```
G90 (establecer modo absoluto)
G0 X1.000 Y1.000 Z0.100 (movimiento rápido a la posición inicial)
S100 M3 (encender el husillo, 100 RPM)
G33.1 Z-0.750 K0.05 (roscado rígido de 20 TPI de 0.750 de profundidad)
M2 (final del programa)
```

- Consulte las secciones G90 , G0 y M2 para obtener más información.

Es un error si:

- Se omiten todas las palabras de eje.
- El husillo no gira cuando se ejecuta este comando
- El movimiento lineal solicitado excede los límites de velocidad de la máquina. debido a la velocidad del husillo

5.2.23. G38.n Sonda recta

```
Ejes G38.n
```

- *G38.2* - sonda hacia la pieza de trabajo, parada en contacto, señal de error si falla
- *G38.3* - sonda hacia la pieza de trabajo, parada en contacto
- *G38.4* - alejar la sonda de la pieza de trabajo, parada en caso de pérdida de contacto, señal de error si falla
- *G38.5* - alejar la sonda de la pieza de trabajo, parada en caso de pérdida de contacto

importante
No podrá utilizar un movimiento de sonda hasta que su máquina se ha configurado para proporcionar una señal de entrada de sonda. La señal de entrada de la sonda debe estar conectada a *motion.probe-input* en un archivo .hal. G38.n usa motion.probe-input para determinar cuándo la sonda ha hecho (o perdido) el contacto. True para el contacto de la sonda cerrado (en contacto), false para el contacto de la sonda abierto.

Programe *G38.n ejes* para realizar una operación de sonda recta. Las palabras de eje son opcionales, pero se debe utilizar al menos una de ellas. Las palabras del eje juntas definen el punto de destino hacia el cual se moverá la sonda, a partir de la ubicación actual. Si la sonda no se dispara antes alcanzar el destino con G38.2 y G38.4, se indicará un error.

La herramienta en el eje debe ser una sonda o contactar un interruptor de sonda.

En respuesta a este comando, la máquina mueve el punto controlado (que debe estar en el centro de la bola de la sonda) en línea recta al avance actual hacia el punto programado. En el modo de alimentación de tiempo inverso, la velocidad de alimentación será tal que todo el movimiento desde el punto actual hasta el punto programado tomara el tiempo especificado. El movimiento se detiene (dentro de los límites de aceleración de la máquina) cuando se alcanza el punto programado, o cuando se produce el cambio solicitado en la entrada de la sonda, lo que ocurra primero.

Después de una prueba exitosa, los parámetros #5061 a #5069 se establecerán en las coordenadas X, Y, Z, A, B, C, U, V, W de la ubicación del punto controlado en el momento en que la sonda cambió de estado (en el sistema de coordenadas de trabajo actual). Después de un sondeo fallido, se establecen en las coordenadas del punto programado. El parámetro #5070 se establece en 1 si la sonda tuvo éxito y 0 si la sonda falló. Si la operación de sondeo falla, G38.2 y G38.4 señalarán un error mostrando un mensaje en pantalla si la GUI seleccionada lo admite, y al detener la ejecución del programa.

Un comentario de la forma *(PROBEOPEN filename.txt)* abrirá *filename.txt* y almacenará las coordenadas XYZABCUVW de cada sonda recta exitosa en ella. El archivo debe cerrarse con *(PROBECLOSE)*. Para más información vea la sección comentarios.

Se incluye un archivo de ejemplo *smartprobe.ngc* (en el directorio de ejemplos) para demostrar el uso de movimientos de sonda para registrar en un archivo las coordenadas de una pieza. El programa *smartprobe.ngc* podría usarse con *ngcgui* con cambios mínimos.

Es un error si:

- el punto actual es el mismo que el punto programado.
- no se usa palabra de eje
- la compensación del cortador está habilitada
- la velocidad de alimentación es cero
- la sonda ya está en el estado objetivo

5.2.24. Compensación G40 desactivada

- *G40* - apaga la compensación del cortador. Si la compensación de herramienta está ON, el siguiente movimiento debe ser lineal y más largo que el diámetro de la herramienta. Se puede desactivar la compensación cuando ya está desactivada.

Ejemplo G40

```
; La ubicación es X1 después de terminar el movimiento compensado del cortador
G40 (desactivar compensación)
G0 X1.6 (movimiento lineal más largo que el diámetro actual de la fresa)
M2 (final del programa)
```

Consulte las secciones G0 y M2 para obtener más información.

Es un error si:

- Un movimiento de arco G2/G3 se programa después de un G40.
- El movimiento lineal después de desactivar la compensación es menor que el diámetro de la herramienta.

5.2.25. G41, G42 Compensación del cortador

```
G41 <D-> (a la izquierda de la ruta programada)
G42 <D-> (a la derecha de la ruta programada)
```

- *D* - número de herramienta

La palabra D es opcional; si no hay una palabra D, se utilizará el radio de la herramienta cargada actual (si no se carga ninguna herramienta y no se proporciona una palabra D, se usará un radio de 0).

Si se proporciona, la palabra D es el número de herramienta a utilizar. Esto normalmente será el número de la herramienta en el husillo (en cuyo caso la palabra D es redundante y no necesita ser suministrada), pero puede ser cualquier número de herramienta válido.

nota
G41/G42 D0 es un poco especial. Su comportamiento es diferente en máquinas de cambio de herramientas aleatorio y no aleatorio (vea la sección Cambio de herramienta). En no aleatorio, *G41/G42 D0* aplica el TLO de la herramienta actualmente en el husillo, o un TLO de 0 si no hay herramienta en el husillo. En máquinas de cambio de herramienta aleatorio, *G41/G42 D0* aplica el TLO de la herramienta T0 definida en el archivo de tabla de herramientas (o causa un error si T0 no está definido en la tabla de herramientas).

Para iniciar la compensación del cortador a la izquierda del perfil de la pieza, use G41. G41 inicia la compensación del cortador a la izquierda de la línea programada visto desde el extremo positivo del eje perpendicular al plano.

Para iniciar la compensación del cortador a la derecha del perfil de la pieza, use G42. G42 inicia la compensación del cortador a la derecha de la línea programada vista desde el extremo positivo del eje perpendicular al plano.

El movimiento de entrada debe ser al menos tan largo como el radio de la herramienta y puede ser un movimiento rápido.

La compensación del cortador se puede realizar si el plano XY o el plano XZ está activo.

Los comandos de usuario M100-M199 están permitidos cuando la compensación de cortador está activada.

El comportamiento del centro de mecanizado cuando la compensación del cortador está activada se describe en la sección Compensación de cortador junto con ejemplos de código.

Es un error si:

- El número D no es un número de herramienta válido o 0.
- El plano YZ está activo.
- Se ordena que la compensación del cortador se active cuando ya está activada.

5.2.26. G41.1, G42.1 Compensación dinámica del cortador

```
G41.1 D- <L-> (a la izquierda de la ruta programada)
G42.1 D- <L-> (a la derecha de la ruta programada)
```

- *D* - diámetro del cortador
- *L* - orientación de herramienta (ver orientación de herramienta de torno)

G41.1 y G42.1 funcionan igual que G41 y G42 con la capacidad adicional de poder programar el diámetro de la herramienta. La palabra L por defecto es 0 si no se especifica.

Es un error si:

- El plano YZ está activo.
- El número L no está en el rango de 0 a 9 inclusive.
- El número L se usa cuando el plano XZ no está activo.
- Se ordena que la compensación del cortador se active cuando ya está activada.

5.2.27. Compensación de longitud de herramienta G43

```
G43 <H->
```

- *H* - número de herramienta (opcional)

G43 permite la compensación de la longitud de la herramienta. G43 cambia los movimientos posteriores compensando las coordenadas de eje con la longitud del offset. G43 no causa ningún movimiento. La próxima vez que se mueva un eje compensado, el punto final de ese eje será la ubicación compensada.

G43 sin una palabra H usa la herramienta cargada actualmente del último *Tn M6*.

G43 Hn usa el offset de la herramienta n.

nota
G43 H0 es un poco especial. Su comportamiento es diferente en máquinas de cambio de herramientas aleatorio y máquinas de cambio no aleatorio (ver la sección Cambiadores de herramientas). En máquinas de cambiador de herramientas no aleatorio, *G43 H0* aplica el TLO de la herramienta actualmente en el husillo, o un TLO de 0 si no hay herramienta. En máquinas con cambiador de herramientas aleatorio, *G43 H0* aplica el TLO de la herramienta T0 definida en la tabla de herramientas (o causa un error si T0 no está definido en la tabla).

Línea de ejemplo G43 H-

```
G43 H1 (establecer compensaciones de herramienta utilizando los valores de la herramienta 1 ↩
    en la tabla de herramientas)
```

Es un error si:

- el número H no es un entero, o
- el número H es negativo, o
- el número H no es un número de herramienta válido (aunque tenga en cuenta que 0 es un número válido de herramienta en máquinas no aleatorias y significa "la herramienta actualmente en el husillo")

5.2.28. G43.1 Offset de longitud dinámica de herramienta

```
G43.1 ejes
```

- *G43.1 ejes* - cambia los movimientos siguientes reemplazando los offsets actuales de ejes. G43.1 no causa ningún movimiento. La próxima vez un eje compensado se mueva, el punto final de ese eje será la ubicación compensada.

Ejemplo G43.1

```
G90 (establecer modo absoluto)
T1 M6 G43 (carga herramienta 1 y offset de longitud de herramienta, Z está en 0 máquina y ↩
    DRO muestra Z1.500)
G43.1 Z0.250 (offset actual de la herramienta en 0.250, DRO ahora muestra Z1.250)
M2 (final del programa)
```

- Consulte las secciones G90 , T y M6 para más información.

Es un error si:

- el movimiento se ordena en la misma línea que *G43.1*

NOTA:G43.1 no escribe en la tabla de herramientas.

5.2.29. G43.2 Aplicar offset de longitud de herramienta adicional

```
G43.2 H- ejes-
```

- *H* - número de herramienta

G43.2 aplica una compensación de herramienta adicional.

G43.2 Ejemplo

```
G90 (establecer modo absoluto)
T1 M6 (herramienta de carga 1)
G43 (o G43 H1:reemplaza todas las compensaciones de herramienta con el offset de T1)
G43.2 H10 (también agregue en el offset de herramienta de T10)
M2 (final del programa)
```

Puede sumar un número arbitrario de compensaciones llamando a G43.2 mas veces. No hay suposiciones integradas sobre qué números son compensaciones de geometría y cuales son compensaciones de desgaste, o que solo deba tener una de cada una.

Al igual que los otros comandos G43, G43.2 no causa ningún movimiento. La próxima vez que un el eje compensado se mueva, el punto final de ese eje será la ubicación compensada.

Es un error si:

- *H* no está especificado y no se especifican offsets de eje
- 'H está especificado y el número de herramienta dado no existe en la tabla de herramientas
- Se especifica *H* y también se especifican los ejes

NOTA:G43.2 no escribe en la tabla de herramientas.

5.2.30. G49 Cancelar compensación de longitud de herramienta

- *G49* - cancela la compensación de longitud de herramienta

Está bien programar usando el mismo offset que ya está en uso. Tambien es correcto programar sin offset de longitud de herramienta si no se está utilizando ninguno.

5.2.31. G52 Compensación del sistema de coordenadas local

```
G52 ejes
```

G52 se utiliza en un programa de pieza como un "offset del sistema de coordenadas local" temporal dentro del sistema de coordenadas de la pieza de trabajo. Más información sobre G52 en la sección offsets locales y globales .

5.2.32. G53 Mover en coordenadas de máquina

```
G53 ejes
```

Para moverse en el sistema de coordenadas de máquina, programe *G53* en la misma línea que un movimiento lineal. *G53* no es modal y debe ser programado en cada línea. *G0* o *G1* no tiene que ser programado en la misma línea si están actualmente activos.

Por ejemplo, *G53 G0 X0 Y0 Z0* moverá los ejes a la posición inicial incluso si el sistema de coordenadas actualmente seleccionado tiene compensaciones en efecto.

Ejemplo G53

```
G53 G0 X0 Y0 Z0 (movimiento lineal rápido al origen de máquina)
G53 X2 (movimiento lineal rápido a coordenada absoluta X2)
```

- Consulte la sección G0 para obtener más información.

Es un error si:

- G53 se usa sin que G0 o G1 estén activos,
- G53 se usa mientras la compensación del cortador está activada.

5.2.33. G54-G59.3 Seleccionar sistema de coordenadas

- *G54* - seleccione el sistema de coordenadas 1
- *G55* - selecciona el sistema de coordenadas 2
- *G56* - selecciona el sistema de coordenadas 3
- *G57* - selecciona el sistema de coordenadas 4
- *G58* - selecciona el sistema de coordenadas 5
- *G59* - selecciona el sistema de coordenadas 6
- *G59.1* - selecciona el sistema de coordenadas 7
- *G59.2* - selecciona el sistema de coordenadas 8
- *G59.3* - selecciona el sistema de coordenadas 9

Los sistemas de coordenadas almacenan los valores del eje y el ángulo de rotación XY alrededor del eje Z en los parámetros que se muestran en la siguiente tabla.

Cuadro 5.2: Parámetros del sistema de coordenadas

Selec.	SC	X	Y	Z	A	B	C	U	V	W	R
G54	1	5221	5222	5223	5224	5225	5226	5227	5228	5229	5230
G55	2	5241	5242	5243	5244	5245	5246	5247	5248	5249	5250
G56	3	5261	5262	5263	5264	5265	5266	5267	5268	5269	5270
G57	4	5281	5282	5283	5284	5285	5286	5287	5288	5289	5290
G58	5	5301	5302	5303	5304	5305	5306	5307	5308	5309	5310
G59	6	5321	5322	5323	5324	5325	5326	5327	5328	5329	5330
G59.1	7	5341	5342	5343	5344	5345	5346	5347	5348	5349	5350
G59.2	8	5361	5362	5363	5364	5365	5366	5367	5368	5369	5370
G59.3	9	5381	5382	5383	5384	5385	5386	5387	5388	5389	5390

Es un error si:

- La selección de un sistema de coordenadas se utiliza mientras la compensación del cortador está activada.

Consulte la sección Sistema de coordenadas para obtener una descripción general de los sistemas de coordenadas.

5.2.34. Modo de ruta exacta G61

- *G61* - Modo de ruta exacta, movimiento exactamente como se programó. Los movimientos se ralentizarán o detendran según sea necesario para llegar a cada punto programado. Si dos movimientos secuenciales son exactamente co-lineales, no se detendrá.

5.2.35. G61.1 Modo de parada exacta

- *G61.1* - Modo de parada exacta, el movimiento se detendrá al final de cada segmento programado.

5.2.36. Mezcla de ruta G64

```
G64 <P- <Q->>
```

- *P* - tolerancia de mezcla de movimiento
- *Q* - tolerancia naive cam
- *G64* - la mejor velocidad posible.
- *G64 P- <Q->* mezcla con tolerancia.
- *G64* - sin P significa mantener la mejor velocidad posible, sin importar cómo de lejos del punto programado se termina.
- *G64 P- Q-* - es una forma de ajustar su sistema para obtener el mejor compromiso entre velocidad y precisión. La tolerancia P significa que la ruta real no se apartará más de P- del punto final programado. La velocidad se reducirá si es necesario para mantener el camino. Además, cuando se active G64 P- Q- , se enciende *nive cam*; cuando hay una serie de alimentación lineal XYZ se mueve al mismo feed rate que están a menos de Q- lejos de ser colineales, se colapsan en un movimiento lineal único. En G2 / G3 se mueve en el plano G17 (XY) cuando el máximo La desviación de un arco de una línea recta es menor que el G64 P- tolerancia el arco se divide en dos líneas (desde el inicio del arco hasta punto medio, y desde el punto medio hasta el final). esas líneas están sujetas a El ingenuo algoritmo de leva para líneas. Por lo tanto, line-arc, arc-arc y los casos de línea de arco y la línea de línea se benefician de la *cámara ingenua detector*. Esto mejora el rendimiento de contorneado al simplificar el camino. Está bien programar para el modo que ya está activo. Ver también la sección control de trayectoria para más información sobre estos modos. Si Q no se especifica, tendrá el mismo comportamiento que antes y use el valor de P-.

G64 P- Línea de ejemplo

```
G64 P0.015 (configure la siguiente ruta para estar dentro de 0.015 de la ruta real)
```

Es una buena idea incluir una especificación de control de ruta en el preámbulo de cada archivo de código G.

5.2.37. Ciclo de acabado del torno G70

```
G70 Q- <X-> <Z-> <D-> <E-> <P->
```

- *Q*:el número de subrutina.
- *X* - La posición X inicial, por defecto es la posición inicial.
- *Z* - La posición Z inicial, por defecto es la posición inicial.
- *D* - La distancia inicial del perfil, por defecto es 0.
- *E*:la distancia final del perfil, por defecto es 0.
- *P* - El número de pases para usar, por defecto es 1.

El ciclo *G70* está diseñado para usarse después de la forma del perfil dado en la subrutina con el número Q se ha cortado con G71 o G72.

1. Movimiento preliminar
   - Si se usan Z o X, movimiento rápido a esa posición está hecho. Esta posición también se usa entre cada pasada de acabado.
   - Entonces un movimiento rápido al inicio del perfil es ejecutado.
   - La ruta dada en Q- se sigue usando G1 y Sección 5.2.5 comandos.
   - Si se requiere un próximo pase, hay otro rápido al intermedio ubicación, antes de que se realice un rápido al inicio del perfil.
   - Después de la pasada final, la herramienta se deja al final del perfil incluyendo E-.
2. Pases múltiples La distancia entre el pase y el perfil final es (pase-1) * (D-E) / P + E. Donde pasar el número de pase y D, E y P son los números D / E / P.
3. La distancia se calcula utilizando la posición inicial del ciclo, con una distancia positiva hacia este punto.
4. Filete y chaflanes en el perfil. Es posible agregar filetes o chaflanes en el perfil, vea Sección 5.2.38 para más detalles.

Es un error si:

- No hay una subrutina definida con el número dado en Q.
- La ruta dada en el perfil no es monotónica en Z o X.
- Sección 5.2.17 no se ha utilizado para seleccionar el plano ZX.

5.2.38. G71 G72 Ciclo de desbaste en Torno

```
G71 Q- <X-> <Z-> <D-> <I-> <R->
G71.1 Q- <X-> <Z-> <D-> <I-> <R->
G71.2 Q- <X-> <Z-> <D-> <I-> <R->
G72 Q- <X-> <Z-> <D-> <I-> <R->
G72.1 Q- <X-> <Z-> <D-> <I-> <R->
G72.2 Q- <X-> <Z-> <D-> <I-> <R->
```

- *Q*:el número de subrutina.
- *X* - La posición X inicial, por defecto es la posición inicial.
- *Z* - La posición Z inicial, por defecto es la posición inicial.
- *D* - La distancia restante al perfil, por defecto es 0.
- *I*:el incremento de corte, por defecto es 1.
- *R* - La distancia de retracción, por defecto es 0..

El ciclo G71 / G72 está diseñado para cortar un perfil en un torno. El G71 Los ciclos eliminan las capas del material mientras atraviesan en la dirección Z. Los ciclos G72 eliminan material mientras atraviesan el eje X, el llamado ciclo de enfrentamiento. La dirección de viaje es la misma que en el camino dado en La subrutina. Para el ciclo G71, la coordenada Z debe ser monotónicamente cambiando, para el G72 esto es necesario para el eje X.

El perfil se da en una subrutina con el número Q-. Esta subrutina puede contener comandos de movimiento G0, G1, G2 y G3. Todos los otros comandos son ignorado, incluidos los ajustes de alimentación y velocidad. Los comandos Sección 5.2.3 son interpretado como G1 comandos. Cada comando de movimiento también puede incluir un número A o C opcional. Si se agrega el número A- un filete con el radio dado por A se insertará en el punto final de ese movimiento, si este radio es demasiado grande, el algoritmo fallará con una ruta no monotónica error. También es posible usar el número C, que permite un chaflán para ser insertado Este chaflán tiene los mismos puntos finales que un filete del mismo tendría una dimensión pero se inserta una línea recta en lugar de un arco.

Cuando está en modo absoluto, U (para X) y W (para Z) pueden usarse como offsets incrementales.

Los ciclos G7x.1 no cortan los bolsillos. Los ciclos G7x.2 solo se cortan después de primer bolsillo y continuar donde se detuvo G7x.1. Es recomendable salir algo de material adicional para cortar antes del ciclo G7x.2, por lo que si se usa G7x.1 a D1.0, el G7x.2 puede usar D0.5 y se eliminarán 0.5 mm mientras se mueve de un bolsillo al siguiente.

Los ciclos G7x normales cortan todo el perfil en un ciclo.

1. Movimiento preliminar
   - Si se usan Z o X, movimiento rápido a esa posición está hecho.
   - Después de cortar el perfil, la herramienta se detiene al final del perfil, incluida la distancia especificada en D.
2. El número D se usa para mantener una distancia del perfil final, para permitir que quede material para el acabado.

Es un error si:

- No hay una subrutina definida con el número dado en Q.
- La ruta dada en el perfil no es monotónica en Z o X.
- Sección 5.2.17 no se ha utilizado para seleccionar el plano ZX.
- Sección 5.2.25 está activo.

5.2.39. G73 Ciclo de taladrado con rotura de viruta

```
G73 X- Y- Z- R- Q- <L->
```

- *R* - posición de retraccion a lo largo del eje Z.
- *Q* - incremento delta a lo largo del eje Z.
- *L* - repetir

El ciclo *G73* es taladrar o fresar con rotura de viruta. Este ciclo toma un número Q que representa un incremento *delta* a lo largo del eje Z.

1. Movimiento preliminar
   - Si la posición Z actual está por debajo de la posición R, el eje Z hace un movimiento rápido a la posición R.
   - Moverse a las coordenadas X Y
2. Mueve el eje Z solo a avance actual hacia abajo la cantidad delta o hacia la posición Z, lo que sea menos profundo.
3. Rápido elevandose un poco.
4. Repite los pasos 2 y 3 hasta alcanzar la posición Z en el paso 2.
5. rapido del eje Z a la posición R.

Es un error si:

- el número Q es negativo o cero.
- el número R no está especificado

5.2.40. G74 Ciclo de roscado izquierdo, con Dwell

```
G74 (X- Y- Z-) o (U- V- W-) R- L- P- $ -
```

El ciclo *G74* está diseñado para roscar con mandril flotante y parada en el fondo del agujero.

1. Movimiento preliminar, como se describe en la sección movimientos preliminares e intermedios
2. Deshabilita los ajustes de avance y velocidad.
3. Mueve el eje Z a la velocidad de avance actual a la posición Z.
4. Detiene el husillo seleccionado (elegido por el parámetro $)
5. Inicia la rotación del husillo en sentido horario.
6. Espera P segundos.
7. Mueve el eje Z a la velocidad de avance actual para despejar Z
8. Restaurar ajustes de alimentación y velocidad al estado anterior

La longitud de la pausa se especifica mediante una palabra *P-* en el bloque G74. El pitch del hilo es F dividido por S. En el ejemplo, el S100 F125 ofrece un paso de 1,25 mm por revolución.

5.2.41. Ciclo de roscado G76

```
G76 P- Z- I- J- R- K- Q- H- E- L- $-
```

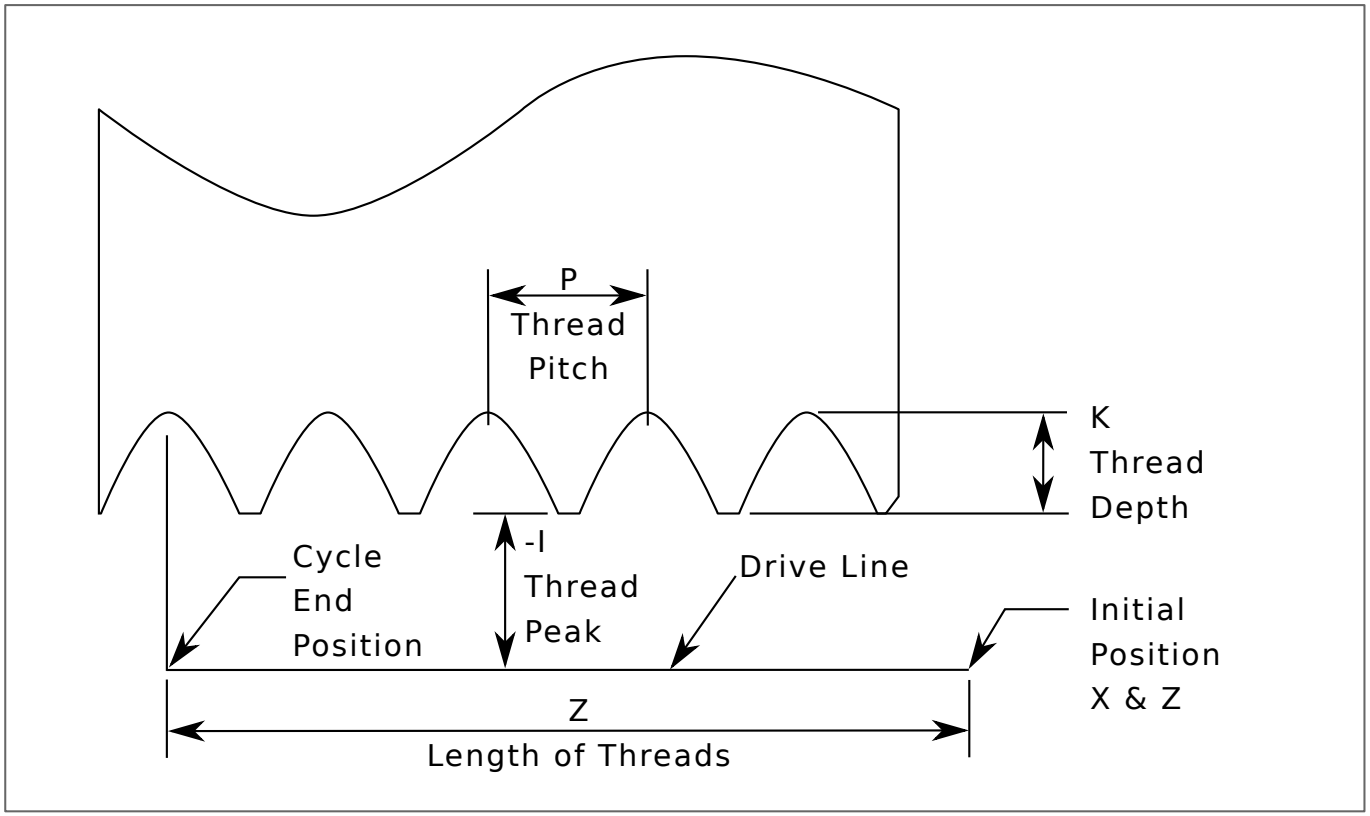


Figura 5.4: G76 Roscado

- *Línea guia* - una línea a través de la posición X inicial paralela a Z.
- *P- - paso de hilo* en distancia por revolución.
- *Z-* - posición final de los hilos. Al final del ciclo, la herramienta estará en esa posición Z

---

nota
Cuando G7 *Modo de diámetro de torno* está en vigor, los valores para *I*, *J* y *K* son mediciones de diámetro. Cuando G8 *Modo de radio de torno* está en vigor, los valores de *I*, *J* y *K* son medidas de radio.

- *I-* - offset de la *cresta de hilo* desde la *línea guia*. Valores negativos de *I* son hilos externos, y valores positivos de *I* son hilos internos. Generalmente, el material se ha convertido a este tamaño antes del ciclo *G76*.
- *J-* - un valor positivo que especifica la *profundidad de corte inicial*. El primer corte de roscado será *J* más allá de la posición de *cresta de hilo*.
- *K-* - un valor positivo que especifica la *profundidad del hilo completo*. El final del corte de roscado será *K* más allá de la posición de *cresta de hilo*.

Configuraciones opcionales

- *$* - - el número de husillo con el que se sincronizará el movimiento (predeterminado 0). Por ejemplo, si está programado $1, entonces el movimiento comenzará en el reinicio de spindle.1.index-enable y procede en sincronía con el valor de spindle.1.revs
- *R*- - La *degresión de profundidad*. (degresión = un descenso por etapas o pasos) *R1.0* selecciona profundidad constante en sucesivos pases de roscado. *R2.0* selecciona área constante. Valores entre 1.0 y 2.0 seleccionan profundidad decreciente pero área creciente. Los valores superiores a 2.0 seleccionan área decreciente. Tenga en cuenta que los valores de degresión innecesariamente altos causarán un gran número de pases.
- *Q*- - *ángulo de deslizamiento compuesto* es el ángulo (en grados) que describe en qué medida los pases sucesivos deben compensarse a lo largo de la línea guia. Esto se usa para hacer que un lado de la herramienta elimine más material que el otro. Un valor positivo *Q* hace que el borde de ataque de la herramienta corte más fuerte. Los valores típicos son 29, 29.5 o 30.
- *H*- - El número de *pases elasticos*. Los pases elasticos son pases adicionales a la profundidad total del hilo. Si no se desean pases adicionales, programe *H0*.
- *E*- - Especifica la distancia a lo largo de la línea guia utilizada para conicidad de la entrada. El ángulo del cono será de modo que el último pase se estrecha hacia la cresta del hilo sobre la distancia especificada con E. *E0.2* dará un cono para las primeras/últimas 0.2 unidades de longitud a lo largo del hilo. Para un programa de conicidad de 45 grados, E igual que K
- *L*- - Especifica qué extremos del hilo tendran conicidad. Programar *L0* para no cono (predeterminado), *L1* para el cono de entrada, *L2* para el cono de salida o *L3* tanto para entradas y salidas cónicas. Las entradas cónicas se detendrán en la línea guia para sincronizar con el pulso de índice y luego moverse a avance en el comienzo del cono. Sin entrada cónica, la herramienta hará un rapido a la profundidad de corte, luego sincronizara y comenzara el corte.

La herramienta se movera a las posiciones X y Z iniciales antes de emitir G76. La posición X es la *línea guia* y la posición Z es el inicio de los hilos.

La herramienta se detendrá brevemente para la sincronización antes de cada subproceso de pase, por lo que se requerirá una ranura de alivio en la entrada a menos que el comienzo del hilo pase el final del material o se usa una conicidad de entrada.

A menos que use conicidad de salida, el movimiento de salida no está sincronizado con la velocidad del husillo y será un movimiento rápido. Con un husillo lento, el movimiento de salida puede tomar solo una pequeña fracción de una revolución. Si la velocidad del husillo aumenta después de completar varios pases, los movimientos de salida posteriores requerirán una porción mayor de una revolución, lo que resultará en un corte muy pesado durante el movimiento de salida. Esto se puede evitar proporcionando un surco de alivio en la salida, o no cambiar la velocidad del husillo mientras se rosca.

La posición final de la herramienta estará al final de la *línea guia*. Se necesitará un movimiento Z seguro en un hilo interno para sacar la herramienta del agujero.

Es un error si:

- El plano activo no es el plano ZX
- Se especifican otras palabras de eje, como X- o Y-
- El valor de degresión *R*- es menor que 1.0.
- No se especifican todas las palabras requeridas
- *P*-, *J*-, *K*- o *H*- es negativo
- *E*- es mayor que la mitad de la longitud de la línea guia

Conexiones HAL Los pines *spindle.N.at-speed* y *encoder.n.phase-Z* para el husillo debe estar conectados en su archivo HAL antes de que G76 funcione. Vea los pines de husillo en la sección Motion para más información.

Información técnica El ciclo fijo G76 se basa en el movimiento sincronizado del husillo G33. Para más información ver G33 Información técnica.

El programa de ejemplo *g76.ngc* muestra el uso del ciclo fijo G76, y se puede previsualizar y ejecutar en cualquier máquina usando la configuración *sim/lathe.ini*.

G76 Ejemplo

```
G0 Z-0.5 X0.2
G76 P0.05 Z-1 I-.075 J0.008 K0.045 Q29.5 L2 E0.045
```

En la figura, la herramienta está en la posición final después del ciclo G76 que está completado. Puede ver la ruta de entrada a la derecha desde Q29.5 y la ruta de salida a la izquierda desde la L2 E0.045. Las lineas blancas son los movimientos de corte.

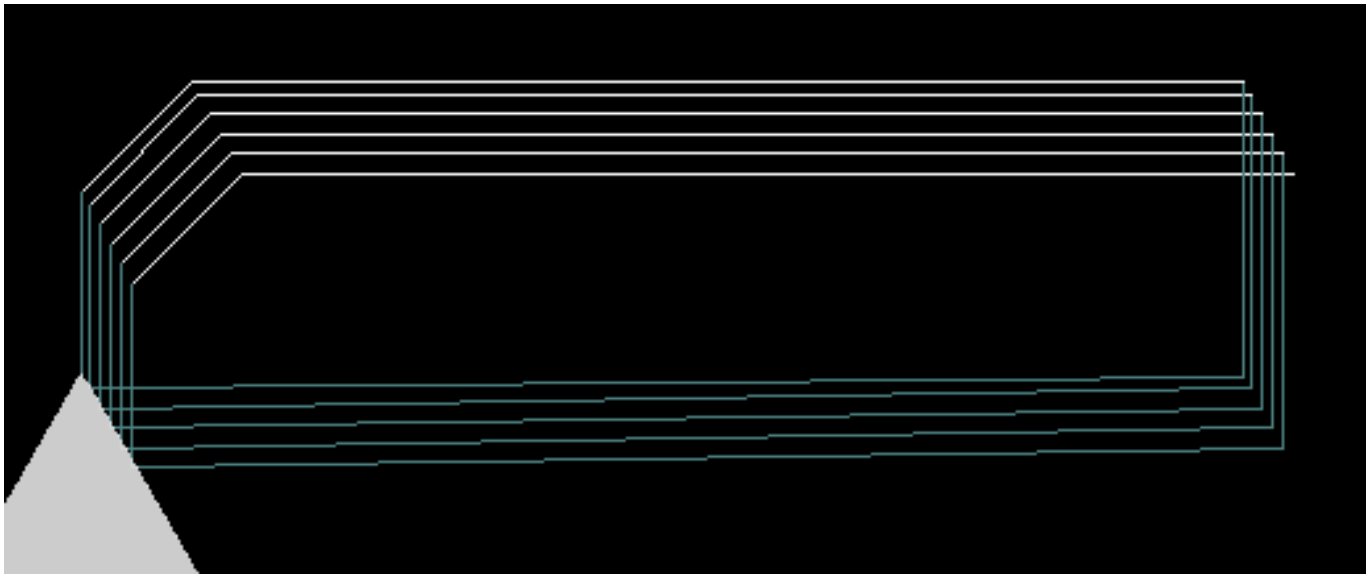


Figura 5.5: G76 Ejemplo

5.2.42. Ciclos Fijos

En esta sección se describen los ciclos fijos *G81* a *G89* y la detencion de ciclo fijo *G80*.

Todos los ciclos fijos se realizan con respecto al plano seleccionado actualmente. Se puede seleccionar cualquiera de los nueve planos. A lo largo de esta sección, la mayoría de las descripciones asumen que el plano XY ha sido seleccionado. El comportamiento es análogo si se selecciona otro plano y se deben usar las palabras correctas. Por ejemplo, en el plano *G17.1*, la acción del ciclo fijo es a lo largo de W, y las ubicaciones o incrementos se dan con U y V. En este caso, sustituya X, Y, Z por U, V, W en las instrucciones a continuación.

Las palabras de eje rotativo no están permitidas en ciclos fijos. Cuando el plano activo es uno de la familia XYZ, las palabras del eje UVW no estan permitidas. Del mismo modo, cuando el plano activo es uno de la familia UVW, las palabras XYZ no están permitidas.

5.2.42.1. Palabras comunes

Todos los ciclos fijos usan grupos X, Y, Z o U, V, W dependiendo del plano seleccionado y la palabra R. La posición R (generalmente significa retracción) es a lo largo del eje perpendicular al plano seleccionado actualmente (eje Z para el plano XY, etc.) Algunos ciclos fijos usan argumentos adicionales.

5.2.42.2. Palabras Sticky

Para los ciclos fijos, llamaremos a un número *sticky* si, cuando el mismo ciclo se utiliza en varias líneas de código en fila, el número debe ser usado la primera vez, pero es opcional en el resto de las líneas. Los números sticky mantienen su valor en el resto de las líneas si no son programados explícitamente con valor diferente. El número R siempre es sticky. En el modo de distancia incremental, los números X, Y y R se tratan como incrementos desde la posición actual y Z como un incremento desde la posición del eje Z antes de que tenga lugar el movimiento que involucra a Z. En modo de distancia absoluta, los números X, Y, R y Z son posiciones absolutas en el sistema de coordenadas actual

5.2.42.3. Repetir ciclo

El número L es opcional y representa el número de repeticiones. L = 0 no está permitido. Si se utiliza la función de repetición, normalmente se usa en modo de distancia incremental, de modo que la misma secuencia de movimientos se repite en varios lugares igualmente espaciados a lo largo de una línea recta. Cuando L- es mayor que 1 en modo incremental con el plano XY seleccionado, las posiciones X e Y se determinan sumando los números X e Y dados a las posiciones X e Y actuales (en el primera vuelta) o a las posiciones X e Y al final de la anterior vuelta (en las repeticiones). Por lo tanto, si programa *L10*, obtendra 10 ciclos El primer ciclo será la distancia X, Y desde la ubicación original. Las posiciones R y Z no cambian durante la repeticion. El número L no es sticky. En modo de distancia absoluta, L > 1 significa *hacer el mismo ciclo en el mismo lugar varias veces*. Omitir la palabra L es equivalente a especificar L = 1.

5.2.42.4. Modo de retracción

La altura del movimiento de retracción al final de cada repetición (llamada *despejar Z* en las descripciones) está determinado por la configuración del modo de retracción, ya sea a la posición Z original (si está por encima de la posición R y el modo de retracción es *G98*, OLD_Z) o, de lo contrario, a la posición R. Consulte la sección G98 G99.

5.2.42.5. Errores de ciclo fijo

Es un error si:

- faltan palabras de eje durante un ciclo fijo,
- se usan juntas palabras de eje de diferentes grupos (XYZ) (UVW),
- se requiere un número P pero se usa un número P negativo,
- se utiliza un número L que no evalúa a un entero positivo,
- se usa movimiento de eje giratorio durante un ciclo fijo,
- la velocidad de alimentación de tiempo inverso está activa durante un ciclo fijo,
- o la compensación del cortador está activa durante un ciclo fijo.

Si el plano XY está activo, el número Z es sticky y es un error si:

- falta el número Z y el mismo ciclo fijo no estaba activo,
- o el número R es menor que el número Z

Si otros planos están activos, las condiciones de error son análogas a las condiciones XY anteriores.

5.2.42.6. Movimientos preliminares e intermedios

El movimiento preliminar es un conjunto de movimientos que es común a todos los ciclos fijos de fresado. Si la posición Z actual está por debajo de la posición R, el eje Z hace un movimiento rápido a la posición R. Esto solo sucede una vez, independientemente del valor de L.

Además, al comienzo del primer ciclo y en cada repetición, se realizan uno o dos movimientos siguientes

1. Un movimiento rápido paralelo al plano XY para la posición XY dada,
2. El eje Z hace un movimiento rápido a la posición R, si no está ya en la posición R.

Si otro plano está activo, los movimientos preliminares y intermedios son análogo.

5.2.42.7. ¿Por qué usar un ciclo fijo?

Hay al menos dos razones para usar ciclos fijos. El primero es economía del código. Un solo orificio tomaría varias líneas de código para definirlo.

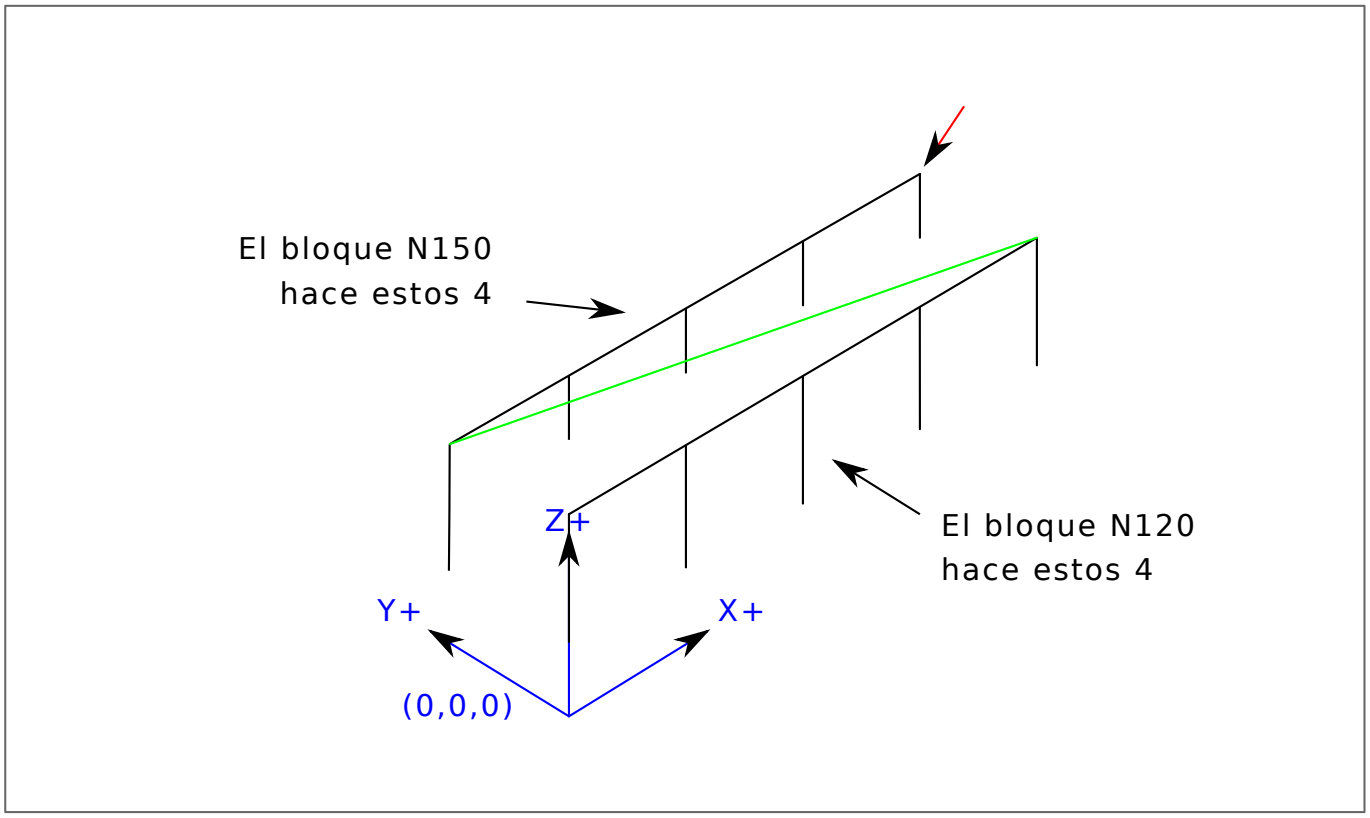
El Ejemplo 1 G81 demuestra cómo podría ser un ciclo fijo. Se utiliza para producir 8 agujeros con diez líneas de código G dentro del modo de ciclo fijo. El siguiente programa producirá el mismo conjunto de 8 agujeros usando cinco líneas. para el ciclo fijo. No sigue exactamente el mismo camino ni perfora en el mismo orden que el ejemplo anterior, pero la economía de la escritura de un buen ciclo fijo debería ser obvia.

NOTA:los números de línea no son necesarios, pero ayudan a aclarar estos ejemplos

Ocho agujeros

```
N100 G90 G0 X0 Y0 Z0 (home)
N110 G1 F10 X0 G4 P0.1
N120 G91 G81 X1 Y0 Z-1 R1 L4 (ciclo de taladro)
N130 G90 G0 X0 Y1
N140 Z0
N150 G91 G81 X1 Y0 Z-0.5 R1 L4 (ciclo de taladro)
N160 G80 (apagar ciclo fijo)
N170 M2 (final del programa)
```

El G98 a la segunda línea de arriba significa que el movimiento de retorno será al valor de Z en la primera línea ya que es más alto que el valor R especificado.

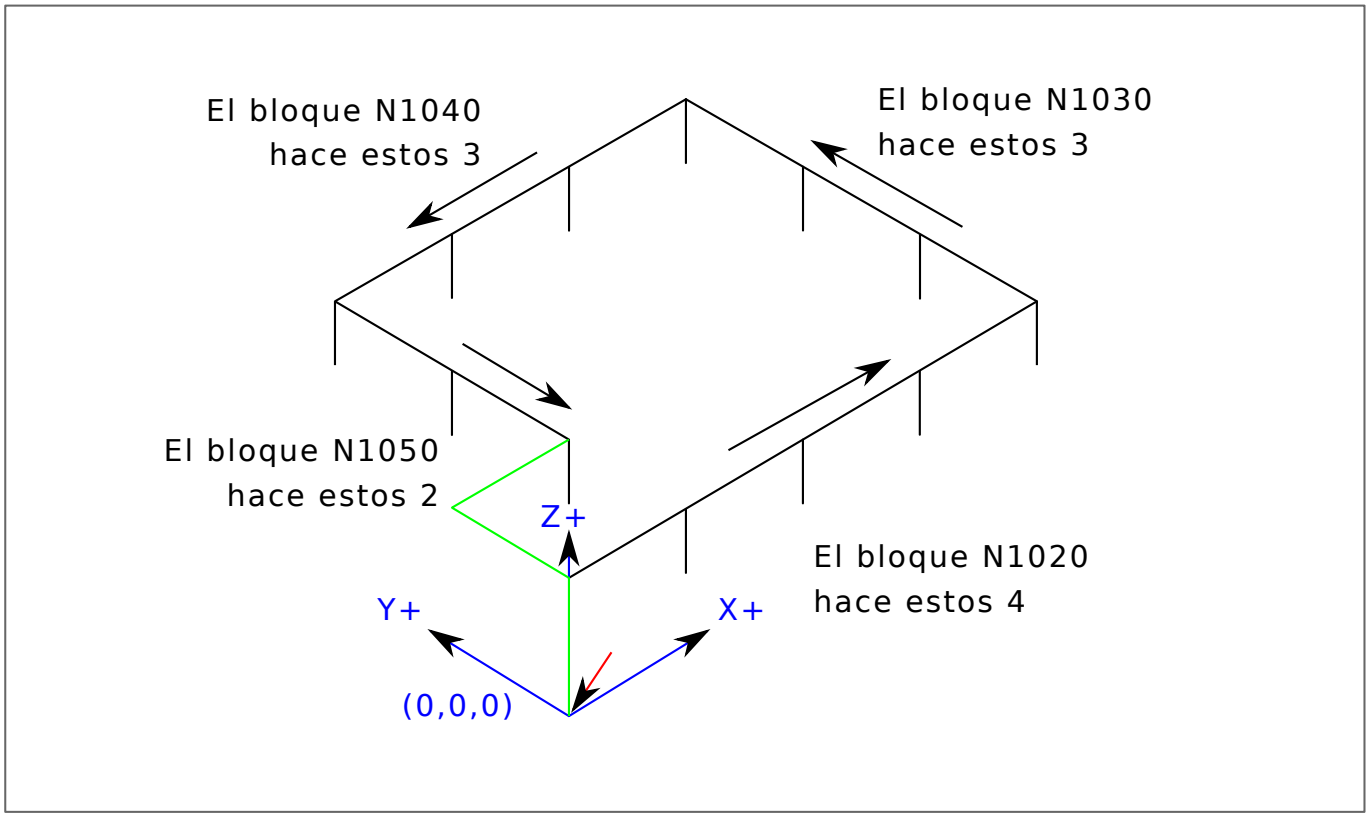


Doce agujeros en un cuadrado

Este ejemplo demuestra el uso de la palabra L para repetir un conjunto de ciclos de taladrado incrementales para bloques sucesivos de código dentro del mismo modo de movimiento G81. Aquí producimos 12 agujeros usando cinco líneas de código en el modo de movimiento de ciclo.

```
N1000 G90 G0 X0 Y0 Z0 (home)
```

```
N1010 G1 F50 X0 G4 P0.1
N1020 G91 G81 X1 Y0 Z-0.5 R1 L4 (ciclo de taladro)
N1030 X0 Y1 R0 L3 (repetir)
N1040 X-1 Y0 L3 (repetir)
N1050 X0 Y-1 L2 (repetir)
N1060 G80 (apagar ciclo fijo)
N1070 G90 G0 X0 (traslado rápido a home)
N1080 Y0
N1090 Z0
N1100 M2 (fin del programa)
```



La segunda razón para usar un ciclo fijo es que todos producen movimientos preliminares y retornos que se pueden anticipar y controlar independientemente del punto de inicio del ciclo fijo.

5.2.43. G80 Cancelar ciclo fijo

G80 Cancel Modal Motion)

- *G80* - cancela el movimiento modal de ciclo fijo. *G80* es parte del grupo modal 1, así que programar cualquier otro código G del grupo modal 1 también cancela el ciclo fijo.

Es un error si:

- Se programan palabras de eje cuando G80 está activo.

Ejemplo G80

```
G90 G81 X1 Y1 Z1.5 R2.8 (ciclo fijo de distancia absoluta)
G80 (desactivar el movimiento del ciclo fijo)
G0 X0 Y0 Z0 (movimiento rápido a home)
```

El siguiente código produce la misma posición final y el mismo estado de la máquina que el código anterior.

Ejemplo G0

```
G90 G81 X1 Y1 Z1.5 R2.8 (ciclo fijo de distancia absoluta)
G0 X0 Y0 Z0 (movimiento rápido para coordinar inicio)
```

La ventaja del primer conjunto es que, la línea G80 apaga claramente el ciclo fijo G81. Con el primer conjunto de bloques, el programador debe activar de nuevo el movimiento con G0, como se hace en la siguiente línea, o cualquier otra palabra G de modo de movimiento.

Si un ciclo fijo no se apaga con G80 u otra palabra de movimiento, el ciclo fijo intentará repetirse usando el siguiente bloque de código que contenga una palabra X, Y o Z. El siguiente archivo explora (G81) un conjunto de ocho agujeros como se muestra en lo siguiente.

G80 Ejemplo 1

```
N100 G90 G0 X0 Y0 Z0 (home)
N110 G1 X0 G4 P0.1
N120 G81 X1 Y0 Z0 R1 (ciclo de taladro fijo)
N130 X2
N140 X3
N150 X4
N160 Y1 Z0.5
N170 X3
N180 X2
N190 X1
N200 G80 (apagar ciclo fijo)
N210 G0 X0 (traslado rápido a home)
N220 Y0
N230 Z0
N240 M2 (final del programa)
```

nota
Observe el cambio de posición z después de los primeros cuatro agujeros. Además, este es uno de los pocos lugares donde los números de línea tienen algún valor; ser capaz de señalar a un lector una línea específica de código.

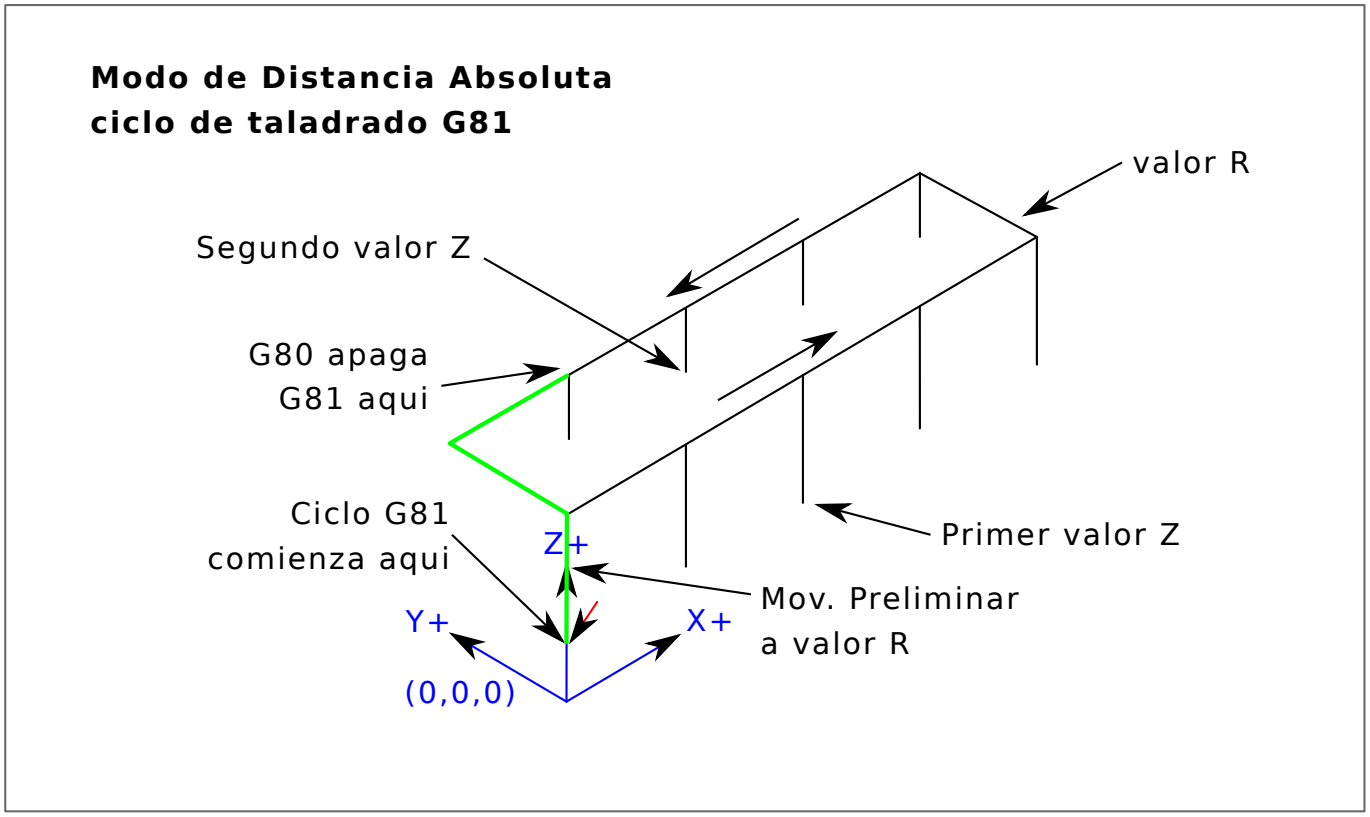


Figura 5.6: Ciclo G80

El uso de G80 en la línea N200 es opcional porque el G0 en la siguiente línea apagará el ciclo G81. Pero usando el G80 como se muestra en el ejemplo 1 proporcionará un ciclo fijo más fácil de leer. Sin el, no es tan obvio que todos los bloques entre N120 y N200 pertenecen al ciclo fijo.

==G81 Ciclo de taladrado

```
G81 (X- Y- Z-) o (U- V- W-) R- L-
```

El ciclo *G81* está destinado a taladrado.

El ciclo funciona de la siguiente manera:

1. Movimiento preliminar, como se describe en la sección movimientos preliminares e intermedios
2. Mover el eje Z al avance actual a la posición Z .
3. El eje Z hace un movimiento rápido para despejar Z.
4. Ejemplo 1 - Posición absoluta G81

Supongamos que la posición actual es (X1, Y2, Z3) y la siguiente línea de NC

```
G90 G98 G81 X4 Y5 Z1.5 R2.8
```

Esto requiere el modo de distancia absoluta (G90) y el modo de retracción OLD_Z (G98) y llama al ciclo de perforación G81 que se realiza una vez.

El valor X y la posición X es 4.

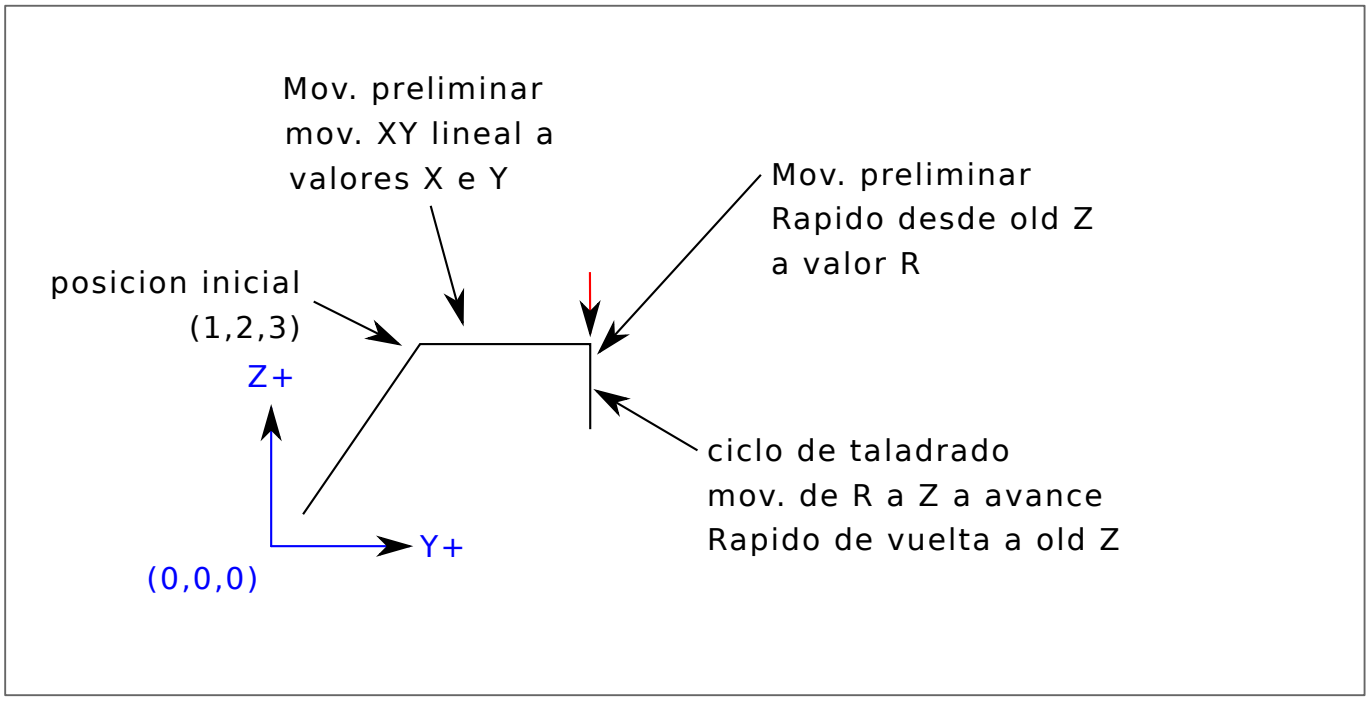
El valor Y y la posición Y es 5.

El valor Z y la posición Z es 1.5.

El valor R y la Z de despeje son 2.8. OLD_Z es 3.

Se realizan los siguientes movimientos:

1. un movimiento rápido paralelo al plano XY a (X4, Y5)
2. un movimiento rápido paralelo al eje Z hacia (Z2.8).
3. movimiento paralelo al eje Z al avance actual a (Z1.5)
4. Un movimiento rápido paralelo al eje Z a (Z3)



1. Ejemplo 2 - posición relativa G81

Supongamos que la posición actual es (X1, Y2, Z3) y la siguiente línea de NC.

```
G91 G98 G81 X4 Y5 Z-0.6 R1.8 L3
```

Esto requiere el modo de distancia incremental (G91) y el modo de retracción OLD_Z (G98). También exige que el ciclo de taladrado G81 se repita tres veces. El valor X es 4, el valor Y es 5, el valor Z es -0.6 y el valor R es 1.8. La posición X inicial es 5 (= 1 + 4), la Y inicial es 7 (= 2 + 5), la posición Z de despeje es 4.8 (= 1.8 + 3), y la Z es 4.2 (= 4.8-0.6). OLD_Z es 3.

El primer movimiento preliminar es un movimiento rápido a lo largo del eje Z para (X1, Y2, Z4.8), ya que OLD_Z < Z de despeje.

La primera repetición consiste en 3 movimientos.

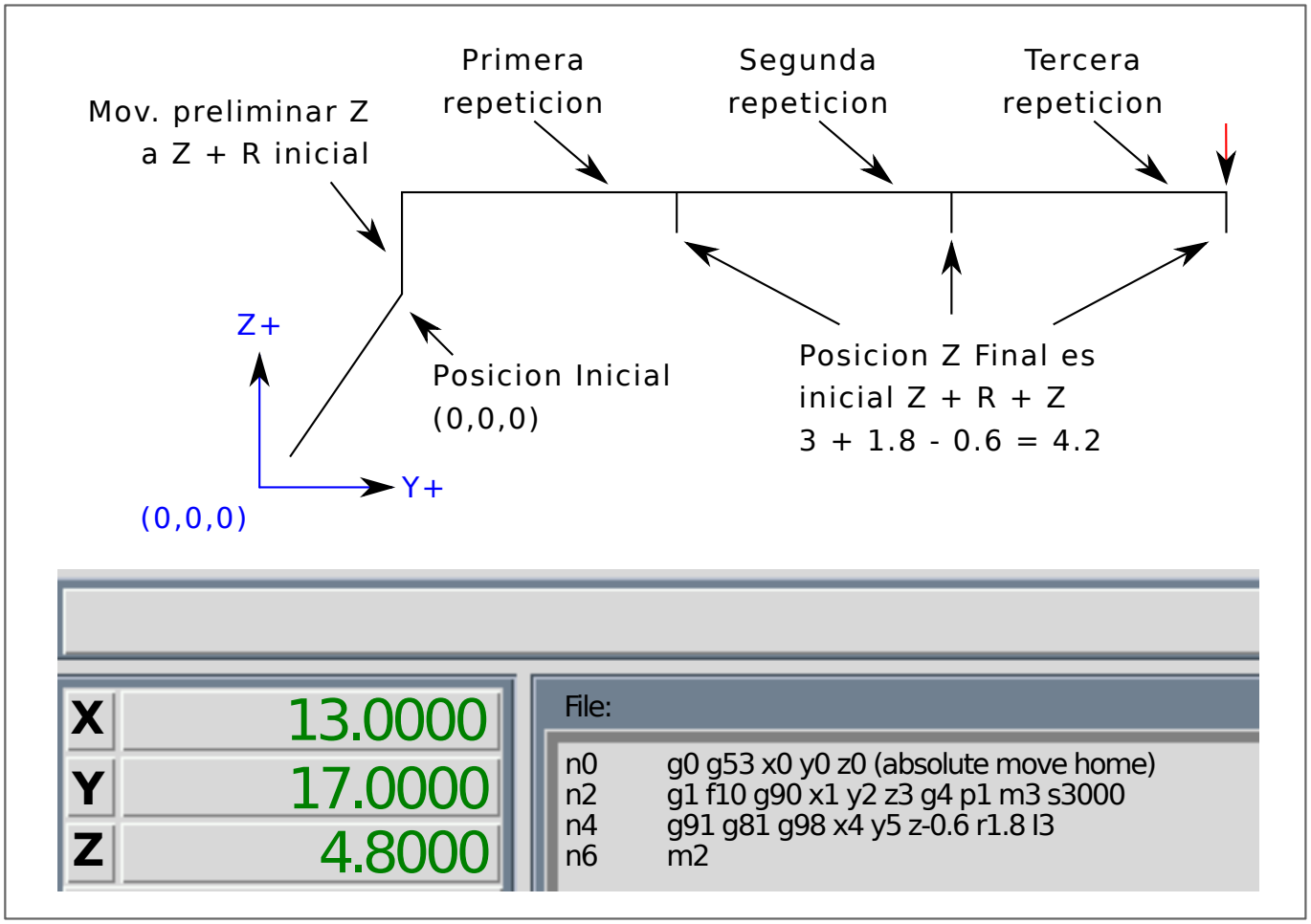
1. un movimiento rápido paralelo al plano XY a (X5, Y7)
2. movimiento paralelo al eje Z a avance hacia (Z4.2)
3. Un movimiento rápido paralelo al eje Z a (X5, Y7, Z4.8)

La segunda repetición consiste en 3 movimientos. La posición X se restablece a 9 (= 5 + 4) y la posición Y a 12 (= 7 + 5).

1. un movimiento rápido paralelo al plano XY a (X9, Y12, Z4.8)
2. movimiento paralelo al eje Z a la velocidad de avance a (X9, Y12, Z4.2)
3. Un movimiento rápido paralelo al eje Z a (X9, Y12, Z4.8)

La tercera repetición consiste en 3 movimientos. La posición X se restablece a 13 (= 9 + 4) y la posición Y a 17 (= 12 + 5).

1. un movimiento rápido paralelo al plano XY a (X13, Y17, Z4.8)
2. movimiento paralelo al eje Z a la velocidad de avance a (X13, Y17, Z4.2)
3. Un movimiento rápido paralelo al eje Z a (X13, Y17, Z4.8)

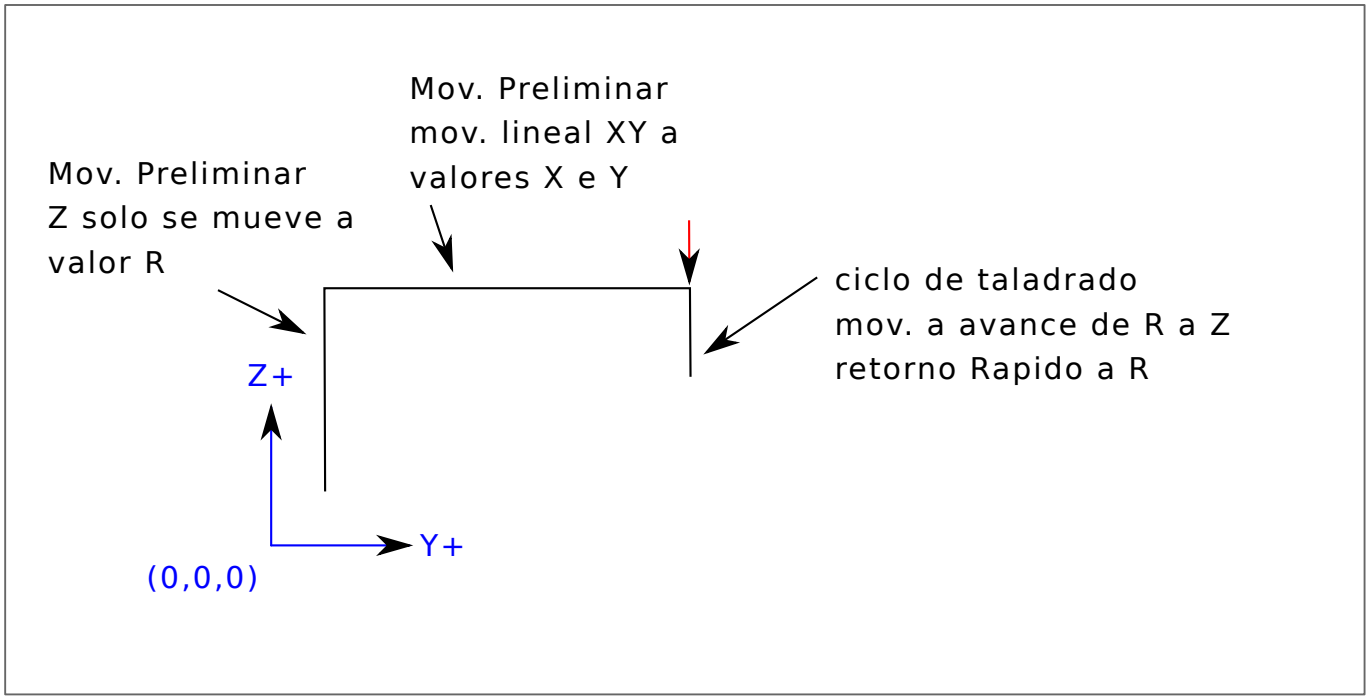


1. Ejemplo 3 - Posición relativa G81

Ahora suponga que ejecuta el primer bloque de código G81 pero desde (X0, Y0, Z0) en lugar de desde (X1, Y2, Z3).

```
G90 G98 G81 X4 Y5 Z1.5 R2.8
```

Como OLD_Z está por debajo del valor R, no agrega nada al movimiento, pero dado que el valor inicial de Z es menor que el valor especificado en R, habrá un movimiento inicial Z durante el movimiento preliminar

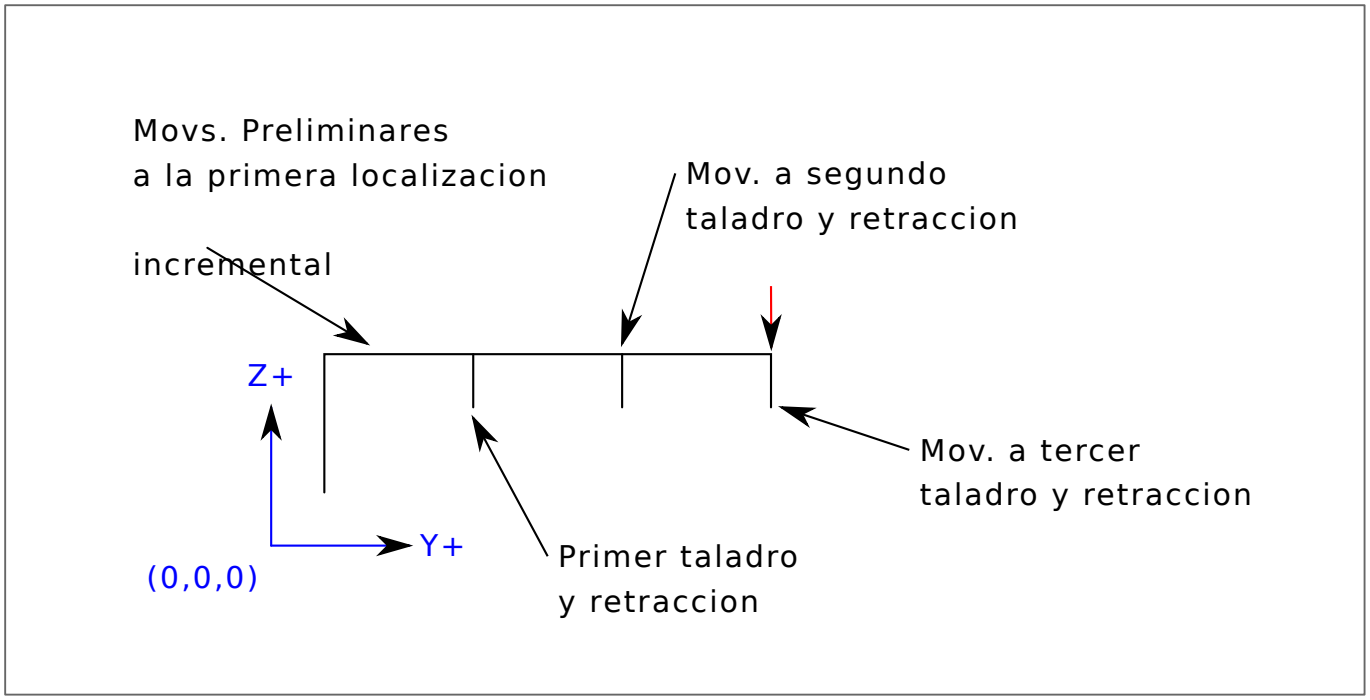


Ejemplo 4 - G81 Absoluto R> Z

Este es un diagrama de la ruta de movimiento para el segundo bloque de código g81.

```
G91 G98 G81 X4 Y5 Z-0.6 R1.8 L3
```

Como esta trama comienza con (X0, Y0, Z0), el intérprete agrega Z0 y R1.8 iniciales y movimientos rápidos a esa ubicación. Después de esa Z inicial, la función de repetición funciona igual que en el ejemplo 3 con la profundidad Z final de 0.6 por debajo del valor R.



Ejemplo 5 - Posición relativa R> Z

```
G90 G98 G81 X4 Y5 Z-0.6 R1.8
```

Como este gráfico comenzaria con (X0, Y0, Z0), el intérprete agrega el Z0 inicial y R1.8 y movimientos rápidos a esa ubicación como en el *Ejemplo 4*. Después de esa Z inicial, se realiza el movimiento rápido a X4 Y5. Entonces la profundidad Z final es 0.6 por debajo del valor R. La función de repetición haría que la Z se moviera al mismo lugar nuevamente.

==G82 Ciclo de perforación, con Dwell

```
G82 (X- Y- Z-) o (U- V- W-) R- L- P-
```

El ciclo *G82* está diseñado para perforar con una pausa en la parte inferior de el agujero.

1. Movimiento preliminar, como se describe en la sección movimientos preliminares e intermedios
2. Mueve el eje Z a velocidad de avance a la posición Z.
3. Espera P segundos.
4. El eje Z hace un movimiento rápido para despejar Z.

El movimiento de un ciclo fijo G82 se parece a G81 con una pausa en la parte inferior del movimiento Z. El tiempo de pausa se especifica mediante una palabra *P-* en el bloque G82.

==G83 Ciclo de perforación con picado

```
G83 (X- Y- Z-) o (U- V- W-) R- L- Q-
```

El ciclo *G83* está destinado a perforación o fresado con rotura de viruta. Las retracciones en este ciclo despejan el agujero de esquirlas y cortar las virutas largas (que son comunes cuando se perfora aluminio). Este ciclo toma un número Q que representa un incremento *delta* a lo largo del eje Z. La retracción antes de la profundidad final siempre esta en el plano de *retracción* incluso si G98 está en efecto. La retracción final obedecerá al G98/99 en efecto. G83 funciona igual que G81 con la adición de retracciones durante la operación de perforación.

1. Movimiento preliminar, como se describe en la sección movimientos preliminares e intermedios
2. Mueve el eje Z a velocidad de avance hacia abajo la cantidad delta o a la posición Z, lo que sea menos profundo.
3. Movimiento rápido de regreso al plano de retracción especificado por la palabra R.
4. Movimiento rápido hacia abajo hasta el fondo del agujero actual, retrocediendo un poco.
5. Repite los pasos 2, 3 y 4 hasta alcanzar la posición Z en el paso 2.
6. El eje Z hace un movimiento rápido para despejar Z.

Es un error si:

- el número Q es negativo o cero.

5.2.44. G84 Ciclo de roscado a derecha, con Dwell

```
G84 (X- Y- Z-) o (U- V- W-) R- L- P- $ -
```

El ciclo *G84* está diseñado para roscar con mandril flotante y permanecer en el fondo del agujero.

1. Movimiento preliminar, como se describe en la sección movimientos preliminares e intermedios
2. Deshabilita ajustes de avance y velocidad.
3. Mueve el eje Z a velocidad de avance actual a la posición Z.
4. Detiene el husillo seleccionado (elegido por el parámetro $)

5. Inicia la rotación del husillo en sentido antihorario.
6. Espere P segundos.
7. Mueve el eje Z a la velocidad de avance actual para despejar Z
8. Restaura y habilita ajustes de alimentación y velocidad al estado anterior

La longitud de la pausa se especifica mediante una palabra *P-* en el bloque G84. El pitch de la rosca es F dividido por S. En el ejemplo, S100 F125 ofrece un paso de 1,25 mm por revolución.

5.2.45. G85 Ciclo mandrinado, salida a avance

```
G85 (X- Y- Z-) o (U- V- W-) R- L-
```

El ciclo *G85* está diseñado para mandrinar o escariar, pero podría usarse para taladrar o fresar.

1. Movimiento preliminar, como se describe en la sección movimientos preliminares e intermedios.
2. Mueve el eje Z a velocidad de avance a la posición Z.
3. Retrae el eje Z a velocidad de avance actual al plano R si es más bajo que la Z inicial.
4. Retrae a la velocidad transversal para despejar Z.

5.2.46. G86 Ciclo de taladrado, parada del husillo, salida a rápido.

```
G86 (X- Y- Z-) o (U- V- W-) R- L- P- $-
```

El ciclo *G86* está destinado a mandrinado. Este ciclo usa P segundos de parada.

1. Movimiento preliminar, como se describe en la sección movimientos preliminares e intermedios.
2. Mueve el eje Z a velocidad de avance a la posición Z.
3. Espera P segundos.
4. Detiene el giro del husillo seleccionado (elegido por el parámetro $).
5. El eje Z hace un movimiento rápido para despejar Z.
6. Reinicia el husillo en la dirección en la que iba.

Es un error si:

- el husillo no gira antes de ejecutar este ciclo.

5.2.47. G87 Ciclo de mandrinado posterior

Este código no está implementado actualmente en LinuxCNC. Se acepta, pero el comportamiento es indefinido.

5.2.48. Ciclo de taladrado G88, parada del husillo, salida manual

Este código no está implementado actualmente en LinuxCNC. Se acepta, pero el El comportamiento es indefinido.

5.2.49. G89 Ciclo de mandrinado, con Dwell, salida a avance

```
G89 (X- Y- Z-) o (U- V- W-) R- L- P-
```

El ciclo *G89* está destinado a mandrinado. Este ciclo usa un número P que especifica el número de segundos de parada (Dwell).

1. Movimiento preliminar, como se describe en la sección movimientos preliminares e intermedios.
2. Mover el eje Z solo a avance actual a la posición Z.
3. Espera de P segundos.
4. Retraer el eje Z a velocidad de avance actual para despejar Z.

5.2.50. G90, G91 Modo de distancia

- *G90* - modo de distancia absoluta. En este modo, los números de eje (X, Y, Z, A, B, C, U, V, W) generalmente representan posiciones en términos del sistema coordinado activo. Las excepciones a esta regla se describen explícitamente en la sección G80 G89.
- *G91* - modo de distancia incremental. En este modo, los números de eje generalmente representan incrementos de la coordenada actual.

Ejemplo G90

```
G90 (establecer modo de distancia absoluta)
G0 X2.5 (movimiento rápido a coordenada X2.5 incluyendo cualquier compensación en efecto)
```

Ejemplo G91

```
G91 (establecer modo de distancia incremental)
G0 X2.5 (movimiento rápido de 2.5 udes. desde la posición actual a lo largo del eje X)
```

- Consulte la sección G0 para obtener más información.

5.2.51. G90.1, G91.1 Modo de distancia de arco

- *G90.1* - modo de distancia absoluta para offsets I, J y K. Cuando G90.1 está en vigor, I y J deben especificarse con G2/G3 para el plano XY o J y K para el plano XZ o es un error.
- *G91.1* - modo de distancia incremental para offsets I, J y K. G91.1 devuelve I, J y K a su comportamiento predeterminado.

5.2.52. G92 Offset de sistema de coordenadas G92

```
G92 ejes
```

aviso
Solo use *G92* después de que su máquina se haya posicionado en el punto deseado.

G92 hace que el punto actual tenga las coordenadas que desea (sin movimiento), donde las palabras de eje contienen los números de eje que desea. Todas las palabras de eje son opcionales, pero se debe utilizar al menos una. Si no se usa una palabra de eje para un eje dado, el offset para ese eje será cero.

Cuando se ejecuta *G92*, se mueven los origenes de todos los sistemas de coordenadas. Se mueven de tal manera que el valor del punto controlado actual, en el sistema de coordenadas actualmente activo, se convierte en el valor especificado. Todos los orígenes de sistema de coordenadas. (G53-G59.3) se compensan esta misma distancia.

G92 utiliza los valores almacenados en los parámetros 5211-5219 como los valores de offset X Y Z A B C U V W para cada eje. Los valores de los parámetros son coordenadas de máquina *absolutas* en las *unidades* nativas de máquina como se especificó en el archivo ini. Todos los ejes definidos en el archivo ini se compensarán cuando G92 esté activo. Si no se ingresó un eje después del G92, el offset de ese eje será cero

Por ejemplo, suponga que el punto actual está en X = 4 y no hay ningún offset *G92* activo. Entonces se programa *G92 X7*. Esto mueve todos los orígenes -3 en X, lo que hace que el punto actual se convierta en X = 7. Este -3 se guarda en el parámetro 5211.

Estar en modo de distancia incremental (G91 en lugar de G90) no tiene ningún efecto sobre la acción de *G92*.

Las compensaciones *G92* pueden estar vigentes cuando se llama a *G92*. Si este es el caso, el offset se reemplaza con un nuevo offset que hace que el punto actual se convierta al valor especificado.

Es un error si:

- se omiten todas las palabras del eje.

LinuxCNC almacena las compensaciones G92 y las reutiliza en la próxima ejecución de un programa. Para evitar esto, puede programar un G92.1 (para borrarlos), o un G92.2 (para quitarlos, pero quedan almacenados).

nota
El comando *G52* también se puede usar para cambiar este offset; ver la sección Offsets para más detalles sobre *G92* y *G52* y cómo interactúan.

Consulte la sección Sistema de coordenadas para obtener un resumen de los sistemas de coordenadas.

Consulte la sección parámetros para obtener más información.

5.2.53. G92.1, G92.2 Restablecer compensaciones G92

- *G92.1* - apaga las compensaciones G92 y restablece parámetros 5211 - 5219 a cero.
- *G92.2* - apaga las compensaciones de G92 pero mantén parámetros 5211 - 5219 disponibles.

nota
G92.1 solo borra las compensaciones G92, para cambiar las compensaciones del sistema de coordenadas G53-G59.3 en el código G use G10 L2 o G10 L20.

5.2.54. G92.3 Restaurar compensaciones G92

- *G92.3* - establece el offset de G92 a los valores guardados en los parámetros 5211 a 5219

Puede establecer offsets de eje en un programa y utilizar los mismos offsets en otro programa Programa *G92* en el primer programa. Esto establecerá parámetros 5211 a 5219. No utilice *G92.1* en el resto del primer programa Los valores de los parámetros se guardarán cuando el primer el programa sale y se restaura cuando se inicia el segundo. Use *G92.3* cerca del comienzo del segundo programa. Eso restaurará las compensaciones guardadas en el primer programa.

5.2.55. Modo de velocidad de alimentación G93, G94, G95

- *G93* - Modo de Tiempo Inverso. En modo de velocidad de avance de tiempo inverso, la palabra F significa que el movimiento debe completarse en [uno dividido por el número F] minutos. Por ejemplo, si el número F es 2.0, el movimiento debería ser completado en medio minuto.

  Cuando el modo de velocidad de avance de tiempo inverso está activo, debe aparecer una palabra F en cada línea que tiene un movimiento G1, G2 o G3. Si la linea no tiene G1, G2 o G3, F se ignora. Estar en modo de velocidad de tiempo inverso no afecta a los movimientos G0.

- *G94* - Modo Unidades por Minuto. En el modo de alimentación de unidades por minuto, una palabra F significa que el punto controlado debe moverse a un cierto número de pulgadas por minuto, milímetros por minuto o grados por minuto, dependiendo de qué unidades de longitud se utilizan y qué eje o ejes se mueven.

- *G95* - Modo Unidades por Revolución En modo de unidades por revolución, una palabra F se interpreta como que el punto controlado debe moverse un cierto número de pulgadas por revolución del husillo, dependiendo de qué unidades de longitud se estén utilizando y cuáles eje o ejes se mueven. G95 no es adecuado para roscar; use G33 o G76. G95 requiere que se conecte spindle.N.speed-in. El husillo real con el que se sincroniza la alimentacion se elige con el parámetro $

Es un error si:

- El modo de alimentación de tiempo inverso está activo y una línea con G1, G2 o G3 (explícita o implícitamente) no tiene una palabra F.
- No se especifica una nueva velocidad de alimentación después de cambiar a G94 o G95

5.2.56. G96, G97 Modo de control del husillo

```
G96 <D-> S- <$ -> (Modo de velocidad de superficie constante)
G97 S- <$ -> (Modo RPM)
```

- *D* - RPM máximas del husillo
- *S* - velocidad de superficie
- ’$ "- el husillo del cual se variará la velocidad.
- *G96 D- S-* - selecciona la velocidad superficial constante de *S* pies por minuto (si G20 está en vigor) o metros por minuto (si G21 está en vigor). D- es opcional.

  Cuando use G96, asegúrese de que X0 en el sistema de coordenadas actual (incluidos los offsets y las longitudes de herramienta) es el centro de rotación o LinuxCNC no proporcionará la velocidad de superficie deseada. G96 no se ve afectado por el modo de radio o diámetro.

Para lograr el modo CSS en los husillos seleccionados, ejecute comandos G96 sucesivos para cada husillo antes de emitir M3.

- *G97* selecciona el modo RPM.

Línea de ejemplo G96

```
G96 D2500 S250 (establece CSS con rpm máximo de 2500 y una velocidad de superficie de 250)
```

Es un error si:

- S no se especifica con G96
- Se especifica un movimiento de avance en el modo G96 mientras el husillo no gira

5.2.57. G98, G99 Nivel de retorno del ciclo fijo

- *G98* - se retrocede a la posición en la que se encontraba el eje justo antes de una serie de uno o más ciclos contiguos.
- *G99* - retrocede a la posición especificada por la palabra R del ciclo fijo.

Programe un *G98* y el ciclo fijo usará la posición Z antes del ciclo fijo como la posición de retorno Z si es más alto que el valor R especificado en el ciclo. Si es menor, se usará el valor R. La palabra R tiene diferentes significados en modo de distancia absoluta y en modo de distancia incremental.

G98 Retraer al origen

```
G0 X1 Y2 Z3
G90 G98 G81 X4 Y5 Z-0.6 R1.8 F10
```

G98 en la segunda línea significa que el movimiento de retorno será al valor de Z en la primera línea ya que es más alto que el valor R especificado.

El plano *inicial* (G98) se restablece cada vez que el modo de movimiento del ciclo es abandonado, ya sea explícitamente (G80) o implícitamente (cualquier código de movimiento que no sea un ciclo). Cambiar entre modos de ciclo (por ejemplo, G81 a G83) NO restablece el plano *inicial*. Es posible cambiar entre G98 y G99 durante una serie de ciclos.

5.3. Códigos M

5.3.1. Tabla de referencia rápida de códigos M

Código	Descripción
M0 M1	Pausa del programa
M2 M30	Fin del programa
M60	Pausa Cambio Palet
M3 M4 M5	Control de husillo
M6	Cambio de herramienta
M7 M8 M9	Control de refrigerante
M19	Orientacion de Husillo
M48 M49	Activar/Desactivar Ajustes de avance y husillo
M50	Control de ajuste de alimentación
M51	Control de ajuste del husillo
M52	Control adaptativo de alimentación
M53	Control de parada de alimentación
M61	Establecer número de herramienta actual
M62-M65	Control de salida
M66	Control de entrada
M67	Control de salida analógica sicronizada
M68	Control de salida analógica inmediata
M70	Guardar estado modal
M71	Invalidar estado modal almacenado
M72	Restaurar estado modal
M73	Guardar estado modal Autorestore
M98 M99	Llamar y regresar de subprograma
M100-M199	Códigos M definidos por el usuario

5.3.2. M0, M1 Pausa del programa

- *M0* - pausa temporalmente un programa en ejecución. LinuxCNC permanece en el modo automático y MDI y otras acciones manuales no están habilitadas. Presionando el botón de reanudacion se reiniciará el programa en la siguiente línea.

- *M1* - pausa temporalmente un programa en ejecución si el interruptor de parada opcional está activado. LinuxCNC permanece en el modo automático, por lo que MDI y otras acciones manuales no están disponibles. Al presionar el botón de reanudacion se reiniciará el programa en la línea siguiente.

nota
Está bien programar *M0* y *M1* en modo MDI, pero el efecto probablemente no se notará porque el comportamiento normal en modo MDI es detenerse de todos modos después de cada línea de entrada.

5.3.3. M2, M30 Fin del Programa

- *M2* - finaliza el programa. Al presionar `Ejecutar Programa` ("R"en la GUI Axis) reiniciará el programa desde el comienzo del archivo.
- *M30* - intercambia lanzaderas de palets y finaliza el programa. Presionando *Ejecutar Programa* comenzará el programa desde el principio del archivo.

Ambos comandos tienen los siguientes efectos:

1. Cambia del modo automático al modo MDI.
2. Los offsets de origen se establecen en el valor predeterminado (como *G54*).
3. El plano seleccionado se establece en el plano XY (como *G17*).
4. El modo de distancia se establece en modo absoluto (como *G90*).
5. El modo de velocidad de avance se establece en unidades por minuto (como *G94*).
6. Los ajustes de avance y velocidad están activadas (como *M48*).
7. La compensación del cortador está desactivada (como *G40*).
8. El husillo se detiene (como *M5*).
9. El modo de movimiento actual está configurado para alimentacion (como *G1*).
10. El refrigerante está apagado (como *M9*).

nota
Las líneas de código después de M2/M30 no se ejecutarán. Presionando `Ejecutar Programa` iniciará el programa desde el comienzo del archivo.

aviso
Usar % para terminar el código G no hace lo mismo que un *Fin de programa*. Ver Requisitos de archivo para obtener más información sobre lo que % no hace.

5.3.4. Pausa de cambio de palet M60

- *M60* - intercambia lanzaderas de palets y luego detiene un programa en ejecución temporalmente (independientemente de la configuración de parada opcional). Presionando el botón de inicio del ciclo reiniciará el programa en la siguiente línea.

5.3.5. M3, M4, M5 Control de husillo

- *M3 [$n]* - inicia el husillo seleccionado en sentido horario a la velocidad *S*.
- *M4 [$n]* - inicia el husillo seleccionado en sentido antihorario a la velocidad *S*.
- *M5 [$n]* - detiene el husillo seleccionado.

Use $ para operar en husillos específicos. Si se omite $, entonces los comandos operan en el husillo 0 predeterminado. Use $-1 para operar en todos los husillos activos.

Este ejemplo iniciará los husillos 0, 1 y 2 simultáneamente a diferentes velocidades:

```
S100 $0
S200 $1
S300 $2
M3 $-1
```

Este ejemplo invertirá el husillo 1 pero dejará los otros husillos. girando hacia adelante:

```
M4 $1
```

Y esto detendrá el husillo 2 y dejará rotar a los otros husillos:

```
M5 $2
```

Si se omite $, entonces el comportamiento es exactamente el normal para una máquina de un solo husillo

Está bien usar *M3* o *M4* si la velocidad del husillo S se establece a cero. Si se hace esto, (o si el interruptor de ajuste de velocidad está habilitado y configurado en cero), el husillo no comenzará a girar. Si, más tarde, la velocidad del husillo se establece por encima de cero (o el interruptor de ajuste está activado), el husillo comenzará a girar. Está bien usar *M3* o *M4* cuando el husillo ya está girando o usar *M5* cuando el husillo ya está parado.

5.3.6. Cambio de herramienta M6

5.3.6.1. Cambio manual de herramienta

Si el componente HAL hal_manualtoolchange está cargado, M6 detendrá el husillo y le pedirá al usuario que cambie la herramienta basado en el último número *T*- programado. Para más información sobre hal_manualtoolchange ver la sección Cambio Manual de Herramienta.

5.3.6.2. Cambiador de herramientas

Para cambiar una herramienta en el husillo por la herramienta seleccionada más recientemente (usando una palabra T - vea la Sección Seleccionar Herramienta), programar *M6*. Cuando se completa el cambio de herramienta:

- El husillo se detendrá.
- La herramienta seleccionada (por una palabra T en la misma línea o en cualquier línea después del cambio de herramienta anterior) estará en el husillo.
- Si la herramienta seleccionada no estaba en el husillo antes del cambio de herramienta, la herramienta que estaba en el husillo (si hubiera) se colocará de nuevo en el cambiador de herramientas.
- Si está configurado en el archivo .ini, algunas posiciones de eje pueden moverse cuando se emite un M6. Vea la sección EMCIO para más información sobre opciones de cambio de herramienta.
- No se realizarán otros cambios. Por ejemplo, el refrigerante continuará fluyendo durante el cambio de herramienta a menos que haya sido desactivado por un *M9*.

aviso
El offset de la longitud de la herramienta no cambia con *M6*, use *G43* después de *M6* para cambiar el offset de longitud de la herramienta.

El cambio de herramienta puede incluir movimiento de ejes. Está bien (pero no es útil) programar un cambio en la herramienta que ya está en el husillo. Está bien si no hay ninguna herramienta en la ranura seleccionada; en ese caso, el husillo estará vacío después del cambio de herramienta. Si se seleccionó por última vez la ranura cero, definitivamente no habrá herramienta en el husillo después de un cambio de herramienta. El cambiador tendrá que estar configurado para realizar el cambio de herramienta en hal y posiblemente con classicladder.

5.3.7. M7, M8, M9 Control de refrigerante

- *M7* - activa el refrigerante de niebla. M7 controla el pin iocontrol.0.coolant-mist.
- *M8* - activa el refrigerante de inundación. M8 controla el pin iocontrol.0.coolant-flood iocontrol.0.
- *M9* - apaga M7 y M8.

Conecte uno o ambos pines de control de refrigerante en HAL antes de que M7 o M8 puedan controlar una salida. M7 y M8 se pueden usar para activar cualquier salida a través del código G.

Está bien usar cualquiera de estos comandos, independientemente del estado actual del refrigerante.

5.3.8. M19 Orientacion del Husillo

- *M19 R- Q- [P-] [$-]*
- Posición *R* para rotar desde 0; el rango válido es 0-360 grados
- *Q* Número de segundos de espera hasta que se complete la orientación. Si spindle.N.is-oriented no se hace true dentro del tiempo de espera Q, se produce un error
- Dirección *P* para rotar a la posición.
  - *0* girar segun el movimiento angular más pequeño (predeterminado)
  - *1* siempre girar en sentido horario (igual que la dirección M3)
  - *2* siempre girar en sentido antihorario (igual que en la dirección M4)
- *$* El husillo a orientar (en realidad solo determina a qué pines HAL llevar los comandos de posición del husillo)

M19 es eliminado por cualquiera de M3, M4, M5.

La orientación del husillo requiere un codificador de cuadratura con un índice para detectar la posición del eje del husillo y la dirección de rotación.

Configuración INI en la sección [RS274NGC].

ORIENT_OFFSET = 0-360 (offset fijo en grados agregado a la palabra M19 R)

Pines HAL

- *spindle.N.orient-angle* (out float) Orientación deseada del husillo para M19. Valor del parámetro de palabra M19 R más el valor del parámetro ini [RS274NGC]ORIENT_OFFSET.
- *spindle.N.orient-mode* (out s32) Modo de rotación deseado del husillo. Refleja la palabra del parámetro M19 P, Predeterminado = 0

- *spindle.N.orient* (out bit)) Indica el inicio del ciclo de orientación del husillo. Establecido por M19. Autorizado por cualquiera de M3, M4, M5. Si spindle-orient-fault no es cero cuando spindle-orient es true, el comando M19 falla con un mensaje de error.
- *spindle.N.is-oriented* (in bit) Pin de reconocimiento para orientacion del husillo. Completa el ciclo de orientación. Si spindle-orient era true cuando se afirmó spindle-is-oriented, el pin spindle-orient se limpia y se se afirma el pin spindle-locked. Además, se afirma el pin spindle-brake.
- *spindle.N.orient-fault* (in s32) Entrada del código de fallo para el ciclo de orientación. Cualquier valor que no sea cero hace que el ciclo de orientación se cancele.
- *spindle.N.locked* (out bit) Pin de orientacion de husillo completa. Limpiado por cualquiera de M3, M4, M5.

5.3.9. M48, M49 Control de ajustes de velocidad y alimentación

- *M48* - habilita los controles de ajuste de velocidad del husillo y velocidad de avance.
- *M49* - deshabilita ambos controles.

Estos comandos también toman un parámetro $ opcional para determinar qué husillo es el que operan.

Está bien habilitar o deshabilitar los controles cuando ya están habilitados o deshabilitados. Consulte la sección velocidad de alimentación para obtener más detalles.

5.3.10. M50 Control de ajuste de alimentación

- *M50 <P1>* - habilita el control de ajuste de la velocidad de alimentación. El P1 es opcional.
- *M50 P0* - deshabilita el control de velocidad de avance. Mientras esté desactivado, el ajuste de alimentación no tendrá influencia, y el movimiento se ejecutará a la velocidad de avance programada. (a menos que haya una ajuste de velocidad de alimentación adaptativa activo).

5.3.11. M51 Control de ajuste de velocidad del husillo

- *M51 <P1> <$->* - habilita el control de ajuste de velocidad del husillo para el husillo seleccionado. P1 es opcional.
- *M51 P0 <$->* - deshabilita el programa de control de ajuste de velocidad del husillo. Mientras está deshabilitado, el ajuste de velocidad del husillo no tendrá influencia, y la velocidad del husillo tendrá la programada por el valor especificado de la palabra S (descrito en la seccion velocidad del husillo).

5.3.12. M52 Control de alimentación adaptable

- *M52 <P1>* - utilice una alimentación adaptativa. P1 es opcional.
- *M52 P0* - dejar de usar alimentación adaptativa. Cuando la alimentación adaptativa está habilitada, algunos valores de entrada externa se usan junto con el valor de ajuste de alimentación de la interfaz de usuario y la velocidad de alimentación ordenada para establecer la velocidad de alimentación real. En LinuxCNC, el pin HAL *motion.adaptive-feed* se utiliza para este propósito. Los valores en *motion.adaptive-feed* deberían variar de -1 (velocidad programada en reversa) a 1 (velocidad máxima). 0 es equivalente a mantener avance.

---

nota
El uso de alimentación adaptativa negativa para marcha inversa es una nueva característica y aún no está muy bien probada. El uso previsto es para cortadores de plasma y electroerosion, pero no se limita a tales aplicaciones.

5.3.13. M53 Control de parada de alimentación

- *M53 <P1>* - activa el interruptor de parada de alimentación. P1 es opcional. Habilitar el interruptor de parada de alimentación permitirá que el movimiento sea interrumpido por medio del control de parada de alimentación. En LinuxCNC, el pin HAL *motion.feed-hold* se usa para este propósito. El valor true hará que el movimiento se detenga cuando *M53* esté activo.
- *M53 P0* - deshabilita el interruptor de parada de alimentación. El estado de *motion.feed-hold* no tendrá ningún efecto en la alimentación cuando M53 no esté activo.

5.3.14. M61 Establecer herramienta actual

- *M61 Q-* - cambia el número de herramienta actual mientras está en modo MDI o Manual sin cambio de herramienta. Un uso es cuando enciende LinuxCNC con una herramienta actualmente en el husillo; puede establecer ese número de herramienta sin hacer un cambio de herramienta.

aviso
El desplazamiento de la longitud de la herramienta no cambia con *M61*, use *G43* después de *M61* para cambiar el offset de longitud de la herramienta.

Es un error si:

- Q- no es 0 o mayor

5.3.15. M62 - Control de salida digital M65

- *M62 P-* - activa la salida digital sincronizada con el movimiento. La palabra P especifica el número de salida digital.
- *M63 P-* - apaga la salida digital sincronizada con el movimiento. La palabra P especifica el número de salida digital.
- *M64 P-* - activa la salida digital de inmediato. La palabra P especifica el número de salida digital.
- *M65 P-* - apaga la salida digital inmediatamente. La palabra P especifica el número de salida digital.

La palabra P varía de 0 a un valor predeterminado de 3. Si es necesario, el número de E/S se puede aumentar utilizando el parámetro num_dio al cargar el controlador de movimiento. Vea la Sección de movimiento para más información.

Los comandos M62 y M63 se pondrán en cola. Comandos posteriores referentes al mismo número de salida sobrescribirá la configuración anterior. Se puede especificar mas de un cambio de salida emitiendo más de un comando M62/M63.

El cambio real de las salidas especificadas ocurrirá en el comienzo del siguiente comando de movimiento. Si no hay movimiento posterior, los cambios de salida en cola no sucederán. Lo mejor es siempre programar un código G de movimiento (G0, G1, etc.) justo después del M62/63.

M64 y M65 suceden inmediatamente cuando son recibidos por el controlador. No están sincronizados con el movimiento, y haran romper la mezcla

nota
M62-65 no funcionará a menos que los pines motion.digital-out-nn apropiados sean conectado en su archivo hal a las salidas.

5.3.16. M66 Esperar en entrada

```
M66 P- | E- <L->
```

- *P-* - especifica el número de entrada digital de 0 a 3.
- *E-* - especifica el número de entrada analógica de 0 a 3.
- *L-* - especifica el tipo de modo de espera.
  - *Modo 0: IMMEDIATE* - sin esperas, regresa de inmediato. El valor actual de la entrada se almacena en el parámetro #5399
  - *Modo 1: RISE* - espera a que la entrada seleccionada realize un evento de subida.
  - *Modo 2: FALL* - espera a que la entrada seleccionada realize un evento de bajada.
  - *Modo 3: HIGH* - espera a que la entrada seleccionada pase al estado ALTO.
  - *Modo 4: LOW* - espera a que la entrada seleccionada pase al estado BAJO.
- *Q-* - especifica el tiempo de espera en segundos. Si el tiempo de espera es excedido, la espera se interrumpe y la variable #5399 se mantendrá en el valor -1. El valor Q se ignora si la palabra L es cero (INMEDIATO). Un valor Q de cero es un error si la palabra L no es cero.
- El modo 0 es el único permitido para una entrada analógica.

Líneas de ejemplo M66

```
M66 P0 L3 Q5 (espere hasta 5 segundos para que se active la entrada digital 0)
```

M66 detiene la ejecución posterior del programa, hasta que se produce el evento seleccionado (o el tiempo de espera programado).

Es un error programar M66 con una palabra P y una palabra E (por lo tanto seleccionando una entrada analógica y una digital). En LinuxCNC las entradas no se controlan en tiempo real y, por lo tanto, no se deben utilizar para aplicaciones de tiempo crítico.

El número de E/S se puede aumentar utilizando el parámetro num_dio o num_aio al cargar el controlador de movimiento. Vea la Sección de movimiento para más información.

nota
M66 no funcionará a menos que los pines motion.digital-in-nn o motion.analog-in-nn estén conectados en su archivo hal a una entrada.

Ejemplo de conexión HAL

```
net signal-name motion.digital-in-00 <= parport.0.pin10-in
```

5.3.17. M67 Salida analógica, sincronizada

```
M67 E- Q-
```

- *M67* - establece una salida analógica sincronizada con el movimiento.
- *E-* - número de salida que va de 0 a 3.
- *Q-* - es el valor a configurar (establecer a 0 para desactivar).

El cambio real de las salidas especificadas ocurrirá en el comienzo del siguiente comando de movimiento. Si no hay comando de movimiento posterior, los cambios de salida en cola no sucederán. Lo mejor es siempre programar un código G de movimiento (G0, G1, etc.) justo después de M67. Las funciones M67 son las mismas que las de M62-63.

El número de E/S se puede aumentar utilizando el parámetro num_dio o num_aio al cargar el controlador de movimiento. Vea la Sección de movimiento para más información.

nota
M67 no funcionará a menos que los pines motion.analog-out-nn apropiados sean conectado en su archivo hal a las salidas.

5.3.18. M68 Salida analógica, inmediata

```
M68 E- Q-
```

- *M68* - establece una salida analógica de inmediato.
- *E-* - número de salida que va de 0 a 3.
- *Q-* - es el valor a configurar (establecer a 0 para desactivar).

La salida M68 ocurre inmediatamente cuando son recibidos por el controlador. No están sincronizados con el movimiento, y harán romper la mezcla. M68 funciona igual que M64-65.

El número de E/S se puede aumentar utilizando el parámetro num_dio o num_aio al cargar el controlador de movimiento. Vea la Sección de movimiento para más información.

nota
M68 no funcionará a menos que los pines motion.analog-out-nn apropiados sean conectado en su archivo hal a las salidas.

5.3.19. M70 Guardar estado modal

Para guardar explícitamente el estado modal en el nivel de llamada actual, programe *M70*. Una vez que el estado modal se ha guardado con *M70*, se puede restaurar exactamente a ese estado ejecutando un *M72*.

Un par de instrucciones *M70* y *M72* generalmente se utilizarán para proteger un programa contra cambios modales involuntarios dentro de subrutinas

El estado guardado consiste en:

- configuración actual de G20/G21 (imperial/métrica)
- plano seleccionado (G17/G18/G19 G17.1, G18.1, G19.1)
- estado de la compensación del cortador (G40, G41, G42, G41.1, G42,1)
- modo distancia - relativo/absoluto (G90/G91)
- modo de alimentación (G93/G94, G95)
- sistema de coordenadas actual (G54-G59.3)
- estado de offset de longitud de herramienta (G43, G43.1, G49)
- modo de retracción (G98, G99)
- modo de husillo (G96-css o G97-RPM)

- modo de distancia de arco (G90.1, G91.1)
- modo de radio/diámetro de torno (G7, G8)
- modo de control de ruta (G61, G61.1, G64)
- avance y velocidad actuales (valores *F* y *S*)
- estado del husillo (M3, M4, M5) - encendido/apagado y dirección
- estado de niebla (M7) e inundación (M8)
- configuración de ajuste de velocidad (M51) y ajuste de alimentación (M50)
- ajuste de alimentación adaptativa (M52)
- ajuste de retención de alimentación (M53)

Tenga en cuenta que, en particular, el modo de movimiento (G1, etc.) NO se restaura.

nivel de llamada actual significa:

* ejecutando en el programa principal. Hay una única ubicación de almacenamiento para el estado en el nivel principal del programa; si varias instrucciones *M70* se ejecutan a la vez, solo se restaura el estado guardado más recientemente cuando se ejecute *M72*.

* ejecutando dentro de una subrutina de código G. El estado guardado con *M70* dentro de una subrutina se comporta exactamente como un parámetro con nombre local: solo se puede hacer referencia a esta invocación de subrutina con un *M72* y cuando la subrutina sale, el parámetro desaparece.

Una invocación recursiva de una subrutina introduce un nuevo nivel de llamada.

5.3.20. M71 Invalidar estado modal almacenado

Se invalida el estado modal guardado con un *M70*, o por un *M73* en la llamada actual (ya no se puede restaurar).

Un *M72* posterior en el mismo nivel de llamada fallará.

Si se ejecuta en una subrutina que protege el estado modal mediante un *M73*, return o endsub **no** restaurará el estado modal.

La utilidad de esta función es dudosa. No se debe confiar en ella.

5.3.21. M72 Restaurar estado modal

El estado modal guardado con un código M70 puede ser restaurado ejecutando un *M72*.

El manejo de G20/G21 se trata especialmente a medida que se interpretan los avances de manera diferente dependiendo de G20/G21: si las unidades de longitud (mm/in) están a punto de ser cambiadas por la operación de restauración, *M72* restaurará primero el modo distancia y luego todos los demás estados, incluido el avance, para asegurarse de que el valor de alimentación se interpreta en la configuración correcta de la unidad.

Es un error ejecutar un *M72* sin guardar con *M70* previo a ese nivel.

El siguiente ejemplo demuestra como guardar y restaurar explícitamente el estado modal alrededor de una llamada de subrutina usando *M70* y *M72*. Tenga en cuenta que la subrutina *imperialsub* no es "consciente" de las características de M7x y puede ser usado sin modificar:

```
O <showstate> sub
(DEBUG, imperial=#<_imperial> absoluto=#<_absolute> avance=#<_feed> rpm=#<_rpm>)
O <showstate> endsub

O <imperialsub> sub
g20 (imperial)
g91 (modo relativo)
```

```
F5 (alimentación baja)
S300 (bajas rpm)
(debug, in subroutine, state now:)
o<showstate> call
O<imperialsub> endsub

; programa principal
g21 (métrico)
g90 (absoluto)
f200 (velocidad rápida)
S2500 (rpm altas)

(debug, in main, state now:)
o<showstate> call

M70 (guardar el estado de la llamada en el nivel global)
O<imperialsub> call
M72 (restaurar explícitamente el estado)

(debug, back in main, state now:)
o<showstate> call
m2
```

5.3.22. M73 Guardar y Autorestaurar estado modal

Para guardar el estado modal dentro de una subrutina y restaurar el estado en *endsub* o cualquier ruta de *retorno*, programe *M73*.

Abortar un programa en ejecución en una subrutina que tiene un *M73* **no** restaurará el estado.

Además, el final normal (*M2*) de un programa principal que contiene un *M73* **no** restaurará el estado.

El uso sugerido es al comienzo de una subrutina O-word como en el siguiente ejemplo. Usar *M73* de esta manera permite diseñar subrutinas que necesitan modificar el estado modal pero protegerán el programa de llamada contra cambios modales involuntarios. Tenga en cuenta el uso de parámetros con nombre predefinidos en la subrutina *showstate*.

```
O<showstate> sub
(DEBUG, imperial=#<_imperial> absoluto=#<_absolute> avance=#<_feed> rpm=#<_rpm>)
O<showstate> endsub

O<imperialsub> sub
M73 (guardar el estado de la llamada en el contexto de llamada actual, restaurar en return ↩
    o endsub)
g20 (imperial)
g91 (modo relativo)
F5 (alimentación baja)
S300 (bajas rpm)
(depuración, en subrutina, estado ahora :)
o<showstate> call

; nota: no se necesita M72 aquí - el siguiente endsub o un
; 'return' explícito restaurará el estado del llamador
O<imperialsub> endsub

; programa principal
g21 (métrico)
g90 (absoluto)
f200 (velocidad rápida)
S2500 (rpm altas)
(depuración, en estado principal, ahora:)
o<showstate> call
o<imperialsub> call
```

```
(depuración, de nuevo en main, estado ahora:)
o<showstate> call
m2
```

5.3.23. M98 y M99

El intérprete admite programas principales y subprogramas de estilo Fanuc con códigos *M98* y *M99*. Ver Programas estilo Fanuc.

5.3.23.1. Restauración selectiva del estado modal

Ejecutar un *M72* o regresar de una subrutina que contiene un *M73* restaurará todo estado modal guardado.

Si solo se deben preservar algunos aspectos del estado modal, una alternativa es el uso de parámetros con nombre predefinidos, parámetros locales y declaraciones condicionales. La idea es recordar los modos que se restaurarán en el comienzo de la subrutina y restaurar estos antes de salir. Aquí está un ejemplo, basado en el fragmento de *nc_files/tool-length-probe.ngc*:

```
O<measure> sub (medida de herramienta de referencia)
;
#<absolute>=#<_absolute> (recuerda en la variable local si se configuró G90)
;
g30 (interruptor)
g38.2 z0 f15 (medida)
g91 g0z.2 (fuera del interruptor)
#1000=#5063 (guardar la longitud de la herramienta de referencia)
(print, la longitud de referencia es #1000)
;
O<restore_abs> if [#<absolute>]
~~~~g90 (restaurar G90 solo si se configuró en la entrada:)
O<restore_abs> endif
;
O<measure> endsub
```

5.3.24. M100 - M199 Comandos definidos por el usuario

```
M1-- <P- Q->
```

- *M1--* - un entero en el rango de 100 a 199.
- *P-* - un número pasado al archivo como primer parámetro.
- *Q-* - un número pasado al archivo como segundo parámetro.

---

nota
Después de crear un nuevo archivo *M1nn* debe reiniciar la GUI para que tenga en cuenta el nuevo archivo, de lo contrario obtendrá un error de *Código M desconocido*.

El programa externo llamado *M100* a *M199* (sin extensión y M mayuscula) se ejecuta con los valores opcionales P y Q como sus dos argumentos. La ejecución del archivo de código G se detiene hasta que sale del programa externo. Se puede usar cualquier archivo ejecutable válido. El archivo debe estar ubicado en la ruta de búsqueda especificada en la configuración del archivo ini. Ver la Seccion Display para obtener más información sobre las rutas de búsqueda.

aviso
No use un procesador de textos para crear o editar los archivos. Un procesador de textos dejará códigos invisibles que causarán problemas y pueden hacer que un bash o un archivo Python no trabaje. Use un editor de texto como Gedit o Notepad++ para crear o editar los archivos.

El error *Código M desconocido utilizado* denota uno de los siguientes

- El comando definido por el usuario no existe
- El archivo no es un archivo ejecutable
- El nombre del archivo tiene una extensión
- El nombre del archivo no sigue este formato M1nn donde nn = 00 a 99
- El nombre del archivo utiliza una M minúscula

Por ejemplo, para abrir y cerrar una pinza que está controlada por un pin de puerto paralelo usando un archivo de script bash usando M101 y M102. Crea dos archivos llamados M101 y M102. Establecerlos como archivos ejecutables (generalmente clic derecho/propiedades/permisos) antes de ejecutar LinuxCNC. Asegúrar que el pin del puerto paralelo no está conectado a nada en el archivo HAL.

Archivo de ejemplo M101

```
#!/bin/bash
# archivo para activar parport pin 14 para abrir la pinza
halcmd setp parport.0.pin-14-out True
exit 0
```

Archivo de ejemplo M102

```
#!/bin/bash
# archivo para apagar parport pin 14 para abrir pinza
halcmd setp parport.0.pin-14-out False
exit 0
```

Para pasar una variable a un archivo M1nn, use la opción P y Q de esta manera:

```
M100 P123.456 Q321.654
```

Archivo de ejemplo M100

```
#!/bin/bash
voltaje=$1
avance=$2
halcmd setp thc.voltage $voltaje
halcmd setp thc.feedrate $avance
exit 0
```

Para mostrar un mensaje gráfico y detener hasta que se cierre la ventana del mensaje use un programa de visualización gráfica como Eye of Gnome para mostrar el archivo gráfico. Cuando lo cierre, el programa se reanudará.

Archivo de ejemplo M110

```
#!/bin/bash
eog /home/john/linuxcnc/nc_files/message.png
exit 0
```

Para mostrar un mensaje gráfico y continuar procesando el archivo de código G sufije un ampersand al comando.

Ejemplo de visualización y sigue adelante M110

```
#!/bin/bash
eog /home/john/linuxcnc/nc_files/message.png &
exit 0
```

5.4. Códigos O

Los códigos O proporcionan control de flujo en programas NC. Cada bloque tiene un número asociado, que es el número utilizado después de O. Se debe tener cuidado para que coincidan adecuadamente los números O. Los códigos O usan la letra *O* no el número cero como el primer carácter en el número como O100 u o100.

5.4.1. Numeración

Los códigos O numerados deben tener un número único para cada subrutina, .Ejemplo de numeración

```
(el comienzo de o100)
o100 sub
(observe que el bloque if-endif usa un número diferente)
~~(el comienzo de o110)
~~o110 if [#2 GT 5]
~~~~(algún código aquí)
~~(el final de o110)
~~o110 endif
~~(un poco más de código aquí)
(el final de o100)
o100 endsub
```

5.4.2. Comentarios

No deben usarse comentarios en la misma línea que la palabra O, ya que el comportamiento puede cambiar en el futuro.

El comportamiento es indefinido si:

- Se utiliza el mismo número para más de un bloque.
- Se usan otras palabras en una línea con una palabra O
- Se usan comentarios en una línea con una palabra O

nota
El uso de minúsculas o hace que sea más fácil distinguir de un 0 que podría haber sido mal escrito. Por ejemplo, que no es un 0 es más fácil de ver en o100 que en O100.

5.4.3. Subrutinas

Las subrutinas comienzan en *Onnn sub* y terminan en *Onnn endsub*. Las líneas entre *Onnn sub* y *Onnn endsub* no se ejecutan hasta que se llama a la subrutina con *Onnn call*. Cada subrutina debe usar un número único.

Ejemplo de subrutina

```
o100 sub
~~G53 G0 X0 Y0 Z0 (rápido al home de la máquina)
o100 endsub

(se llama la subrutina)
o100 call
M2
```

Consulte las secciones G53 , G0 y M2 para obtener más información.

O- Return Dentro de una subrutina, se puede ejecutar *O- return*. Esto vuelve al código de llamada inmediatamente, como si se encontrara *O-endsub*.

O- Ejemplo de return

```
o100 sub
~~(prueba si el parámetro #2 es mayor que 5)
~~o110 if [#2 GT 5]
~~~~(regrese al inicio de la subrutina si la prueba es verdadera)
~~~~o100 return
~~o110 endif
~~~~(esto solo se ejecuta si el parámetro #2 no es mayor que 5)
~~~~(DEBUG, el parámetro 2 es [#1])
o100 endsub
```

Consulte las secciones operadores binarios y parámetros para obtener más información.

O- Call *O- Call* toma hasta 30 argumentos opcionales, que se pasan a la subrutina como *#1*, *#2*, . . . , #N. Los parámetros de #N+1 a #30 tienen el mismo valor como en el contexto de llamada. Al regresar de la subrutina, los valores de los parámetros del 1 al 30 (independientemente del número de argumentos) serán restaurados a los valores que tenían antes de la llamada. Los parámetros #1 - #30 son locales a la subrutina.

Como *1 2 3* se analiza como el número 123, los parámetros deben estar entre corchetes. Lo siguiente llama a una subrutina con 3 argumentos:

O- Ejemplo de llamada

```
o100 sub
~~(prueba si el parámetro #2 es mayor que 5)
~~o110 if [#2 GT 5]
~~~~(regresa al inicio de la subrutina si la prueba es verdadera)
~~~~o100 return
~~o110 endif
~~~~(esto solo se ejecuta si el parámetro #2 no es mayor que 5)
~~~~(DEBUG, el parámetro 1 es [#1])
~~~~(DEBUG, el parámetro 3 es [#3])
o100 endsub

o100 call [100] [2] [325]
```

Los cuerpos de subrutina no pueden estar anidados. Solo pueden llamarse después de ser definidos. Pueden ser llamados desde otras funciones, y pueden llamarse a sí mismos recursivamente si tiene sentido hacerlo. El maximo nivel de anidación de subrutinas es 10.

Las subrutinas pueden cambiar el valor de los parámetros por encima de #30 y esos cambios serán visible para el código de llamada. Las subrutinas también pueden cambiar el valor de parámetros globales con nombre.

5.4.3.1. Programas numerados al estilo Fanuc

Programas numerados (tanto principales como subprogramas), la llamada *M98* y *M99* devuelven códigos M, y sus respectivas diferencias semánticas son un alternativa a las subrutinas rs274ngc descritas anteriormente, proporcionadas para compatibilidad con Fanuc y otros controladores de máquina.

Los programas numerados están habilitados de manera predeterminada y pueden deshabilitarse colocando `DISABLE_FANUC_STYLE_SUB =1` en la sección `[RS274NGC]` del archivo `.ini`.

nota
Las definiciones y llamadas principales y subprogramas numerados difieren de rs274ngc tradicional tanto en sintaxis como en ejecución. Para reducir la posibilidad de confusión, el intérprete generará un error si las definiciones de un estilo se mezclan con las llamadas de otro.

Ejemplo simple de subprograma numerado

```
o1 (Ejemplo 1)          ; Programa principal 1, "Ejemplo 1"
M98 P100                        ; Llama al subprograma 100
M30                                     ; Fin del programa principal

o100                            ; Comienzo del subprograma 100
~~G53 G0 X0 Y0 Z0     ; Rápido a home máquina
M99                                     ; Regresar del subprograma 100
```

o1 (Título) El bloque inicial opcional del programa principal le da al programa principal el número `1`. Algunos controladores tratan un seguiente comentario opcional entre paréntesis como título del programa, `Ejemplo 1` en este ejemplo, pero esto no tiene un significado especial en el intérprete rs274ngc.

M98 P- <L\-> Llama a un subprograma numerado. El bloque `M98 P100` es análogo a la sintaxis tradicional `o100 call`, pero solo se puede usar para llamar a un subprograma numerado definido con `o100` ...` M99`. Una palabra opcional *L-* especifica un recuento de bucles.

M30 El programa principal debe terminarse con `M02` o `M30` (o `M99`; consulte abajo).

O- Inicio de definición de subprograma Marca el inicio de una definición de subprograma numerado. El bloque `O100` es similar a `o100 sub`, excepto que debe colocarse más adelante en el archivo que el bloque de llamada `M98 P100`.

M99 Return de la subrutina numerada El bloque `M99` es análogo a la sintaxis tradicional` o100 endsub`, pero solo puede terminar un programa numerado (`o100` en este ejemplo), y no puede terminar una subrutina que comience con sintaxis `o100 sub`.

La llamada de subprograma `M98` difiere de rs274ngc `O call` en lo siguiente:

- El subprograma numerado debe seguir la llamada `M98` en el archivo de programa. El intérprete arrojará un error si el subprograma precede al bloque de llamada.
- Los parámetros `#1`,` #2`, ..., `#30` son globales y accesibles en subprogramas numerados, similares a los parámetros numerados más altos en llamadas de estilo tradicional. Modificaciones a estos parámetros dentro de un el subprograma son modificaciones globales y persistirán después del retorno de subprograma.
- Las llamadas `M98` a subprograma no tienen valor de retorno.
- Los bloques de llamadas del subprograma `M98` pueden contener una palabra L opcional especificando un recuento de repetición de bucle. Sin la palabra L, el subprograma se ejecutará solo una vez (equivalente a `M98 L1`). Un bloque `M98 L0` no ejecutará el subprograma.

En casos raros, el código `M99` puede usarse para terminar el programa principal, donde indica un *programa sin fin*. Cuando el el intérprete alcanza un `M99` en el programa principal, saltará de nuevo al comienzo del archivo y reanudara la ejecución en la primera línea. Un ejemplo de un programa sin fin es en un ciclo de calentamiento de la máquina; una bloque final de programa con eliminacion `/M30` puede usarse para detener el ciclo en un punto ordenado cuando el operador está listo.

Ejemplo completo de subprograma numerado

```
O1                              ; Programa principal 1
  #1 = 0
  (PRINT,X MAIN BEGIN:  1=#1)
  M98 P100 L5                   ; Llame al subprograma 100
  (PRINT,X MAIN END:  1=#1)
M30                             ; Fin del programa principal

O100                            ; Subprograma 100
  #1 = [#1 + 1]
  M98 P200 L5                   ; Llamada a subprograma 200
  (PRINT,>> O100:  #1)
M99                             ; Return desde Subprograma 100

O200                            ; Subprograma 200
```

```
  #1 = [#1 + 0.01]
  (PRINT,>>>> O200:  #1)
M99                           ; Return desde Subprograma 200
```

En este ejemplo, el parámetro `#1` se inicializa a `0`. El subprograma `O100` se llama cinco veces en un bucle. Anidado dentro de cada llamada a `O100`, el subprograma `O200` se llama cinco veces en un ciclo; 25 veces en total.

Tenga en cuenta que el parámetro `#1` es global. Al final del programa principal, después de las actualizaciones dentro de `O100` y` O200`, su valor será igual a `5.25`.

5.4.4. Bucles

El *bucle while* tiene dos estructuras: *while/endwhile* y *do/while*. En cada caso, el ciclo se cierra cuando la condición *while* se evalúa como falso. La diferencia es cuando se realiza la condición de prueba. El bucle *do/while* ejecuta el código en el bucle y luego verifica la condición de prueba. El bucle *while/endwhile* hace la prueba primero.

Ejemplo While/Endwhile

```
(dibuja una forma de diente de sierra)
G0 X1 Y0 (mover a la posición inicial)
#1 = 0 (asigne al parámetro # 1 el valor de 0)
F25 (establecer una velocidad de alimentación)
o101 while [#1 LT 10]
~~G1 X0
~~G1 Y[#1/10] X1
~~#1 = [#1+1] (incrementar el contador de prueba)
o101 endwhile
M2 (final del programa)
```

Ejemplo Do/While

```
#1 = 0 (asigne al parámetro # 1 el valor de 0)
o100 do
~~(debug, parámetro 1 = #1)
~~o110 if [#1 EQ 2]
~~~~#1 = 3 (asigne el valor de 3 al parámetro #1)
~~~~(msg, #1 se le ha asignado el valor de 3)
~~~~o100 continue (saltar al inicio del bucle)
~~o110 endif
~~(algún código aquí)
~~#1 = [#1+1] (incrementar el contador de prueba)
o100 while [#1 LT 3]
(msg, bucle hecho!)
M2
```

Dentro de un ciclo while, *O- break* sale inmediatamente del ciclo, y *O- continue* salta inmediatamente a la próxima evaluación de la condición *while*. Si aún es cierta, el ciclo comienza nuevamente en la parte superior. Si es falsa, sale del bucle.

5.4.5. Condicionales

El condicional *if* consiste en un grupo de declaraciones con el mismo número *o* que comienzan con *if* y terminan con *endif*. Condiciones opcionales *elseif* y *else* puede estar entre el inicio *if* y el final *endif*.

Si el condicional *if* se evalúa como verdadero, entonces se ejecuta el grupo de declaraciones siguiendo al *if* hasta la siguiente línea condicional.

Si el condicional *if* se evalúa como falso, entonces las condiciones *elseif* son evaluadas en orden hasta que una evalúa como verdadera. Si la condición *elseif* es cierta entonces se ejecutan las declaraciones que siguen al *elseif* hasta el próximo condicional. Si ninguna de las condiciones *if* o *elseif* se evalúa como verdadera, entonces se ejecutan las declaraciones que siguen al *else*. Cuando una condición es evaluada como verdadero, no se evalúan más condiciones en el grupo.

Ejemplo If / Endif

```
(si el parámetro #31 es igual a 3, configure S2000)
o101 if [#31 EQ 3]
~~S2000
o101 endif
```

Ejemplo If ElseIf Else EndIf

```
(si el parámetro #2 es mayor que 5, configure F100)
o102 if [#2 GT 5]
~~F100
o102 elseif [#2 LT 2]
(de lo contrario, si el parámetro #2 es menor que 2, configure F200)
~~F200
(de lo contrario, si el parámetro #2 es de 2 al 5, configure F150)
o102 else
~~F150
o102 endif
```

Se pueden probar varias condiciones mediante declaraciones *elseif* hasta que la ruta *else* finalmente se ejecuta si todas las condiciones anteriores son falsas:

Ejemplo If Elseif Else Endif

```
(si el parámetro #2 es mayor que 5, configure F100)
O102 if [#2 GT 5]
~~F100
(de lo contrario, si el parámetro #2 es inferior a 2, configure F200)
O102 elseif [#2 LT 2]
~~F20
(el parámetro #2 está entre 2 y 5)
O102 else
~~F200
O102 endif
```

5.4.6. Repeat

repeat ejecutará las declaraciones dentro de repeat/endrepeat el número especificado de veces. El ejemplo muestra cómo puede fresar una serie diagonal de formas a partir de la presente posición.

Ejemplo de repeat

```
(Fresar 5 formas diagonales)
G91 (modo incremental)
o103 repeat [5]
... (inserte el código de fresado aquí)
G0 X1 Y1 (movimiento diagonal a la siguiente posición)
o103 endrepeat
G90 (modo absoluto)
```

5.4.7. Indirección

El O-número puede ser dado por un parámetro y/o cálculo.

Ejemplo de Indirección

```
o[#101+2] call
```

Calculando valores en O-palabras Para obtener más información sobre los valores, consulte las siguientes secciones

- parámetros
- expresiones
- operadores binarios
- funciones

5.4.8. Llamando a archivos

Para llamar a un archivo separado con una subrutina, nombre el archivo igual que su llamada e incluya un sub y endub en el archivo. El archivo debe estar en el directorio señalado por *PROGRAM_PREFIX* o *SUBROUTINE_PATH* en el archivo ini. El nombre del archivo puede incluir **letras minúsculas**, números, guiones y guiones bajos solamente. Un archivo de subrutina con nombre solo puede contener una única definición de subrutina.

Ejemplo de archivo con nombre

```
o<myfile> call
```

Ejemplo de archivo numerado

```
o123 call
```

En el archivo llamado, debe incluir el sub y el endsub oxxx y el archivo debe ser un archivo válido.

Ejemplo de archivo llamado

```
(nombre de archivo myfile.ngc)
o<myfile> sub
~~(código aquí)
o<myfile> endsub
M2
```

nota
Los nombres de los archivos son solo letras minúsculas, por lo que *o <MyFile>* se convierte en *o <myfile>* por el intérprete. Más información sobre la ruta de búsqueda y las opciones para la ruta de búsqueda se encuentra en la sección de configuración INI.

5.4.9. Valores de retorno de subrutina

Las subrutinas pueden devolver opcionalmente un valor mediante una expresión opcional en una declaración *endsub* o *return*.

Ejemplo de valor de retorno

```
o123 return [#2 *5]
...
o123 endub [3 * 4]
```

Un valor de retorno de subrutina se almacena en el parámetro con nombre predefinido *<_value>* y el parámetro predefinido *<_value_returned>* se establece en 1, para indicar que se ha devuelto un valor. Ambos parámetros son globales y se borran solo antes de la próxima llamada de subrutina.

5.4.10. Errores

Las siguientes declaraciones provocan un mensaje de error y abortan el interprete:

- un `return` o` endsub` no dentro de una subdefinición - una etiqueta en *repeat* que se define en otra parte - una etiqueta en `while` que se define en otro lugar y no se refiere a un` do` - una etiqueta en `if` definida en otra parte - una etiqueta indefinida en `else` o` elseif` - una etiqueta en `else`,` elseif` o `endif` no apunta a un` if` coincidente - una etiqueta en `break` o` continue` que no apunta a una coincidencia `while` o` do` - una etiqueta en `endrepeat` o` endwhile` sin hacer referencia a un `while` o` repeat` correspondiente Para hacer estos errores advertencias no fatales en stderr, establezca el bit 0x20 en la opción mask [[RS274NGC]FEATURE=` del .ini.

5.5. Otros códigos

5.5.1. F: Establecer velocidad de alimentación

Fx - establece la velocidad de avance a *x*. *x* usualmente está en unidades máquina (pulgadas o milímetros) por minuto.

La aplicación de la velocidad de alimentación es como se describe en la Sección Velocidad de alimentación, a menos que estén vigentes *modo de velocidad de alimentación de tiempo inverso* o *modo de avance por revolución*, en cuyo caso la velocidad de avance es como se describe en la Sección G93 G94 G95.

5.5.2. S: Establecer la velocidad de husillo

Sx [$n] - establece la velocidad del husillo a *x* revoluciones por minuto (RPM). Con el $ opcional establece la velocidad del husillo para un husillo específico. Sin el $, el comando pasará por defecto a spindle.0

Los husillos o husillo seleccionado girarán a esa velocidad cuando un *M3* o *M4* esté en efecto. Está bien programar una palabra S si el husillo está girando o no. Si el interruptor de ajuste de velocidad está habilitado y no está configurado al 100 %, la velocidad será diferente de lo programado.

Está bien programar S0; el husillo no girará.

Es un error si:

- el número S es negativo.

5.5.3. T: Seleccionar herramienta

Tx: prepára el cambio a la herramienta *x*.

La herramienta no se cambia hasta que se programa un *M6* (consulte la sección M6). La palabra T puede aparecer en la misma línea que el *M6* o en una línea anterior. Está bien si las palabras T aparecen en dos o más líneas sin cambio de herramienta. Solo la palabra T más reciente tomará efecto en el siguiente cambio de herramienta.

NOTA: Cuando LinuxCNC está configurado para un cambiador de herramientas no aleatorio (consulte la entrada para RANDOM_TOOLCHANGER en EMCIO), *T0* tiene un manejo especial: no se seleccionará ninguna herramienta. Esto es útil si desea que el husillo quede vacío después de un cambio de herramienta.

NOTA: Cuando LinuxCNC está configurado para un cambiador de herramientas aleatorio (consulte la entrada para RANDOM_TOOLCHANGER en EMCIO), *T0* no recibe ningún tratamiento especial: T0 es una herramienta válida como cualquier otra. Es costumbre usar T0 en un cambiador de herramientas aleatorio para rastrear una ranura vacía, para que se comporte como cambiador de herramientas no aleatorio y descargue el husillo.

Es un error si:

- se usa un número T negativo,
- Se utiliza un número T que no aparece en el archivo de la tabla de herramientas (con la excepción de que T0 en cambiadores de herramientas no aleatorios, que **es** aceptado como se señaló anteriormente).

En algunas máquinas, el carrusel se moverá cuando se programe una palabra T, Al mismo tiempo, se está mecanizando. En tales máquinas, la programación de la palabra T varias líneas antes de un cambio de herramienta ahorrará tiempo. Una práctica común de programación para tales máquinas es poner la palabra T para la siguiente herramienta que se utilizará en la línea después de un cambio de herramienta. Esto maximiza el tiempo disponible para que el carrusel se mueva.

Los movimientos rápidos después de una *T<n>* no se mostrarán en la vista previa de AXIS hasta después un movimiento de alimentación. Esto es para máquinas que viajan largas distancias para cambiar la herramienta, como un torno. Esto puede ser muy confuso al principio. Para desactivar esta característica para la herramienta actual, programe un G1 sin ningún movimiento después de *T<n>*.

5.6. Ejemplos de código G

Después de instalar LinuxCNC, se colocan varios archivos de muestra en la carpeta /nc_files. Asegúrese de que el archivo de muestra sea apropiado para su máquina antes de correrlo.

5.6.1. Ejemplos de fresado

5.6.1.1. Fresado helicoidal

Nombre de archivo: useful-subroutines.ngc

Descripción: Subrutina para fresar un agujero utilizando parámetros.

5.6.1.2. Ranurado

Nombre de archivo: useful-subroutines.ngc

Descripción: Subrutina para fresar una ranura usando parámetros.

5.6.1.3. Sondeo en cuadrícula

Nombre de archivo: gridprobe.ngc

Descripción: sondeo rectangular

Este programa prueba repetidamente en una cuadrícula XY normal y escribe la ubicación sondeada al archivo *probe-results.txt* en el mismo directorio que el archivo .ini.

5.6.1.4. Sonda inteligente

Nombre de archivo: smartprobe.ngc

Descripción: sondeo rectangular

Este programa prueba repetidamente en una cuadrícula XY normal y escribe la ubicación sondeada al archivo *probe-results.txt* en el mismo directorio que el archivo .ini. Es una mejora del archivo de sondeo de cuadrícula.

5.6.1.5. Sondeo de longitud de herramienta

Nombre de archivo: tool-length-probe.ngc

Descripción: Sondeo de longitud de herramienta

Este programa muestra un ejemplo de cómo medir longitudes de herramientas automáticamente usando un interruptor conectado a la entrada de la sonda. Esto es útil para máquinas sin portaherramientas, donde la longitud de una herramienta es diferente cada vez que se inserta.

5.6.1.6. Sonda de agujero

Nombre de archivo: probe-hole.ngc

Descripción: Encontrar el centro y el diámetro de un agujero.

El programa muestra cómo encontrar el centro de un agujero, medir el diámetro del agujero y registrar los resultados.

5.6.1.7. Compensación del cortador

Para ser agregado

5.6.2. Ejemplos de torno

5.6.2.1. Roscado

Nombre de archivo lathe-g76.ngc

Descripción: Refrentado, roscado y tronzado.

Este archivo muestra un ejemplo de roscado en un torno utilizando parámetros.

5.7. Diferencias RS274/NGC

5.7.1. Cambios desde RS274/NGC

DIFERENCIAS QUE CAMBIAN EL SIGNIFICADO DE LOS PROGRAMAS RS274/NGC

Ubicación después de un cambio de herramienta
En LinuxCNC, la máquina no vuelve a su posición original después de un cambio de herramienta. Este cambio se realizó porque la nueva herramienta podría ser más larga que la herramienta anterior y, por lo tanto, el movimiento a la posición original de la máquina podría dejar la punta de la herramienta demasiado baja.

Los parámetros de offset están en unidades de archivo ini
En LinuxCNC, los valores almacenados en parámetros para las ubicaciones home G28 y G30, , los sistemas de coordenadas P1 … P9 y el desplazamiento G92 son en "unidades de archivo ini". Este cambio se realizó porque de lo contrario el significado de una ubicación cambiaba dependiendo de si estaba activo G20 o G21 cuando G28, G30, G10 L2 o G92.3 se programa.

Las longitudes/diámetros de la tabla de herramientas están en unidades de archivo ini
En LinuxCNC, las longitudes de herramienta (compensaciones) y diámetros en la herramienta la tabla se especifica solo en unidades de archivo ini. Este cambio fue hecho porque de lo contrario la longitud de una herramienta y su diámetro cambiaban con G20 o G21 al iniciar los modos G43, G41, G42. Esto hizo imposible ejecutar el código G en unidades no nativas de la máquina, incluso cuando el código G era simple y bien formado (comenzando con G20 o G21, y sin cambió de unidades durante todo el programa), sin cambiar la tabla de herramientas.

G84, G87 no implementado
G84 y G87 no se implementan actualmente, pero se pueden agregar a un lanzamiento futuro de LinuxCNC.

G28, G30 con palabras de eje
Cuando G28 o G30 se programan con solo algunas palabras de eje presentes, LinuxCNC solo mueve los ejes nombrados. Esto es común en otros controles de máquinas. Para mover algunos ejes a un punto intermedio y luego mover todos los ejes al punto predefinido, escriba dos líneas de código G:

G0 X- Y- (ejes para mover al punto intermedio) G28 (mover todos los ejes al punto predefinido)

5.7.2. Adiciones a RS274/NGC

DIFERENCIAS QUE NO CAMBIAN EL SIGNIFICADO DE LOS PROGRAMAS RS274 / NGC

Códigos de roscado G33, G76
Estos códigos no están definidos en RS274/NGC.

G38.2
La punta de la sonda no se retrae después de un movimiento G38.2. Este movimiento de retracción puede agregarse en una versión futura de LinuxCNC.

G38.3 … G38.5
Estos códigos no están definidos en RS274/NGC

O-codes
Estos códigos no están definidos en RS274/NGC

M50 … M53 ajustes
Estos códigos no están definidos en RS274/NGC

M61..M66
Estos códigos no están definidos en RS274/NGC

G43, G43.1
Longitudes de herramienta negativas

La especificación RS274 / NGC dice "se espera que" todas las longitudes de herramienta sean positivas. Sin embargo, G43 funciona para longitudes de herramienta negativas.

Herramientas de torno

La compensación de longitud de herramienta G43 puede compensar la herramienta tanto en la X y la Z. Esta característica es principalmente útil en tornos.

Longitudes de herramientas dinámicas

LinuxCNC permite la especificación de una longitud de herramienta calculada a través de G43.1 I K.

G41.1, G42.1
LinuxCNC permite la especificación de un diámetro de herramienta y, si está en modo torno, orientación en el código G. El formato es G41.1 / G42.1 D L, donde D es el diámetro y L (si se especifica) es la orientación de la herramienta de torno.

G43 sin palabra H
En ngc, esto no está permitido. En LinuxCNC, establece compensaciones de longitud para la herramienta cargada actualmente. Si no hay una herramienta cargada actualmente, es un error. Este cambio se realizó para que el usuario no tenga que especificar el número de herramienta en dos lugares para cada cambio de herramienta, y porque es consistente con la forma en que funciona G41/G42 cuando la palabra D no está especificada

Ejes U, V y W
LinuxCNC permite máquinas con hasta 9 ejes definiendo un conjunto adicional de 3 ejes lineales conocidos como U, V y W

Capítulo 6

Compensacion de Herramientas

:lang:es

6.1. Compensación de herramientas

6.1.1. Compensación de longitud de herramienta

6.1.1.1. Touch Off

Usando Touch Off en la interfaz AXIS puede actualizar la tabla de herramientas de forma automática.

Los pasos típicos para actualizar la tabla de herramientas son:

- Después de homing, cargue una herramienta con *Tn M6* donde *n* es el número de herramienta.
- Mueva la herramienta a un punto establecido utilizando un calibre o realice un corte de prueba y medida.
- Haga clic en el botón "Touch Off" en la pestaña Control Manual (o presione el botón Fin en su teclado).
- Seleccione *Tabla de herramientas* en el cuadro desplegable Sistema de Coordenadas.
- Ingrese el valor calibrado o la dimensión medida y seleccione OK.

La tabla de herramientas se cambiará con la longitud Z correcta para que el DRO muestre la posición Z correcta y se emitirá un comando G43 para para que la longitud Z de la nueva herramienta esté vigente. Touch Off de la tabla de herramientas solo está disponible cuando una herramienta se carga con *Tn M6*.

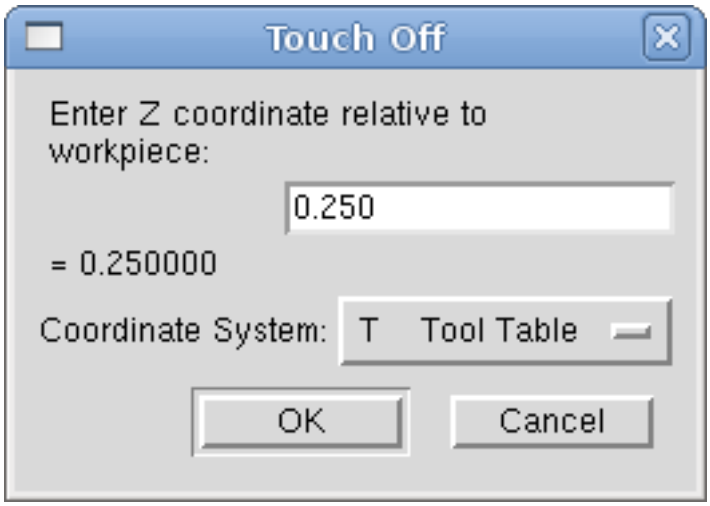


Figura 6.1: Touch Off de tabla de herramientas

6.1.1.2. Usando G10 L1/L10/L11

Los comandos G10 L1/L10/L11 se pueden usar para establecer las compensaciones de la tabla de herramientas:

- *G10 L1 Pn* - Establece los offsets en un valor. Posición actual irrelevante. (vea G10 L1 para más detalles)
- *G10 L10 Pn* - Establece los offsets de modo que la posición actual respecto a fijaciones 1-8 se convierta en un valor. (vea G10 L10 para más detalles)
- *G10 L11 Pn* - Establece los offsets de modo que la posición actual respecto a fijacion 9 se convierta en un valor. (vea G10 L11 para más detalles)

6.1.2. Tabla de herramientas

La *Tabla de herramientas* es un archivo de texto que contiene información sobre cada herramienta. El archivo está ubicado en el mismo directorio que su configuración y se llama *tool.tbl*. Las herramientas pueden estar en un cambiador de herramientas o simplemente ser cambiadas manualmente. El archivo puede editarse con un editor de texto o ser actualizado con G10 L1. Vea la sección tabla de herramientas de torno para un ejemplo del formato de tabla de herramientas de torno. El número máximo de entradas en la tabla de herramientas es 1000. El número máximo de herramienta y de ranura es 99999.

Tool Editor o un editor de texto se pueden usar para editar la tabla de herramientas. Si utiliza un editor de texto, asegúrese de volver a cargar la tabla en la GUI

6.1.2.1. Formato de tabla de herramientas

Cuadro 6.1: Formato de tabla de herramientas

T#	P#	X	Y	Z	A	B	C	U	V	W	Dia	AF	AP	Ori	Com
;	(sin datos después de punto y coma)														
T1	P17	X0	Y0	Z0	A0	B0	C0	U0	V0	W0	D0	I0	J0	Q0	;com
T2	P5	X0	Y0	Z0	A0	B0	C0	U0	V0	W0	D0	I0	J0	Q0	;com
T3	P12	X0	Y0	Z0	A0	B0	C0	U0	V0	W0	D0	I0	J0	Q0	;com

En general, el nuevo formato de línea de la tabla de herramientas es:

-; - punto y coma de apertura, sin datos - T - número de herramienta, 0-99999 (los números de herramienta deben ser únicos) - P - número de ranura, 1-99999 (los números de ranura deben ser únicos) - X..W - offset de herramienta en el eje especificado - punto flotante - D - diámetro de la herramienta - coma flotante, valor absoluto - I - ángulo frontal (solo torno) - punto flotante - J - ángulo posterior (solo torno) - punto flotante - Q - orientación de la herramienta (solo torno) - entero, 0-9 - ; - comienzo del comentario o anotacion - texto

El archivo consta de un punto y coma de apertura en la primera línea, seguido de hasta un máximo de 1000 entradas de herramientas.

nota
Aunque se permiten números de herramienta hasta 99999, el número de entradas en la tabla de herramientas, por el momento, todavía está limitada a un máximo de 1000 herramientas por razones técnicas. Los desarrolladores de LinuxCNC planean eliminar esa limitación. Si tiene un cambiador de herramientas muy grande, sea paciente.

Las versiones anteriores de LinuxCNC tenían dos formatos de tabla de herramientas diferentes para fresadoras y tornos, pero desde la versión 2.4.x, el mismo formato de tabla de herramientas se utiliza para todas las máquinas. Simplemente ignore las partes de la tabla de herramientas que no pertenecen a su máquina, o que no necesita usar.

Cada línea del archivo de tabla de herramientas después del punto y coma de apertura contiene los datos para una herramienta. Una línea puede contener hasta 16 entradas, pero probablemente contendrá muchas menos.

Las unidades utilizadas para la longitud, diámetro, etc., están en unidades de máquina.

Probablemente desee mantener las entradas de la herramienta en orden ascendente, especialmente si va a usar un cambiador de herramientas aleatorio, aunque la tabla de herramientas permite números de herramienta en cualquier orden.

Cada línea puede tener hasta 16 entradas. Se requieren las dos primeras entradas. La última entrada (un comentario o anotación, precedido por un punto y coma) es opcional. La lectura se facilita si las entradas están organizadas en columnas, como se muestra en la tabla, pero el único requisito de formato es que haya al menos un espacio o tabulador después de cada una de las entradas en una línea y un carácter de nueva línea al final de cada entrada.

Los significados de las entradas y el tipo de datos que se colocarán en cada uno son como sigue:

Número de herramienta (requerido) La columna *T* contiene el número (entero sin signo) que representa un número de código para la herramienta. El usuario puede usar cualquier código para cualquier herramienta, siempre que los códigos sean enteros sin signo.

Número de ranura (requerido) La columna *P* contiene el número (entero sin signo) que representa el número de ranura del cambiador de herramientas donde se puede encontrar la herramienta. Las entradas en esta columna deben ser todas diferentes.

Los números de ranura generalmente comenzarán en 1 y subirán hasta la ranura más alta disponible en su cambiador de herramientas. Pero no todos los cambiadores de herramientas siguen este patrón. Sus números de ranura serán determinados por los números. que su cambiador de herramientas utiliza para referirse a las ranuras. Por tanto, los números de ranura que use estarán determinados por el esquema de numeración utilizado en su cambiador de herramientas, y los números de ranura que usa deben tener sentido en su máquina.

Números de datos de offsets (opcional) Las columnas *Datos de Offset* (XYZABCUVW) contienen números reales que representar los offsets de herramientas en cada eje. Este número se usará si la herramienta utiliza offset de longitud y está seleccionada. Estos números pueden ser positivos, cero o negativos, y de hecho son completamente opcionales. Aunque probablemente querrá hacer al menos una entrada aquí, de lo contrario no tendría mucho sentido hacer un entrada en la tabla de herramientas para empezar.

En una fresadora típica, probablemente desee una entrada para Z (offset de longitud de la herramienta). En un torno típico, probablemente desee una entrada para X (offset de herramienta X) y Z (offset de herramienta Z). En una fresadora típica usando compensación del diámetro del cortador (comp. del cortador), probablemente también desee agregar una entrada para D (diámetro del cortador). En un torno típico usando compensación del diámetro de la punta de la herramienta (comp. herramienta), probablemente también desee agregar una entrada para D (diámetro de la punta de la herramienta).

Un torno también requiere información adicional para describir la forma y orientación de la herramienta. Probablemente desee tener entradas para I (ángulo frontal) y J (ángulo posterior) de la herramienta. Probablemente también desee una entrada para Q (orientación de la herramienta).

Vea el capítulo Información del usuario de torno para mas detalles.

La columna *Diámetro* contiene un número real. Este número solo se usa si la compensación del cortador está activada con esta herramienta. Si la ruta programada durante la compensación es el borde del material que se está cortando, este debe ser un número real positivo que represente la medida del diámetro de la herramienta. Si la ruta programada durante la compensación es la trayectoria de una herramienta cuyo diámetro es nominal, este debe ser un número pequeño (positivo o negativo, pero cercano a cero) que representa solo la diferencia entre el diámetro medido de la herramienta y el diámetro nominal. Si la compensación del cortador no se usa con una herramienta, no importa qué número hay en esta columna.

La columna *Comentario* puede usarse opcionalmente para describir la herramienta. Cualquier tipo de descripción estará bien. Esta columna es solo para lectores humanos. El comentario debe ir precedido de un punto y coma.

6.1.2.2. Cambiadores de herramientas

LinuxCNC admite tres tipos de cambiadores de herramientas: *manual*, *ubicación aleatoria* y *ubicación fija*. Información sobre la configuración de un cambiador de herramientas LinuxCNC está en la Sección EMCIO del capítulo INI.

Cambiador de herramientas manual El cambiador manual de herramientas (cambiar la herramienta a mano) se trata como un cambiador de herramienta de ubicación fija y el número P se ignora. Utilizar el cambiador manual de herramientas solo tiene sentido si tiene portaherramientas que permanezcan con la herramienta (Cat, NMTB, Kwik Switch, etc.) cuando se cambia preservando así la ubicación de la herramienta en el husillo. Máquinas con R-8 o los portaherramientas de tipo collar de enrutadores no conservan la ubicación de la herramienta y el cambiador de herramientas manual no debe usarse.

Cambiadores de herramientas de ubicación fija Los cambiadores de herramientas de ubicación fija siempre devuelven las herramientas a una posición fija en el cambiador de herramientas. Esto también incluiría diseños como torretas de torno. Cuando LinuxCNC está configurado para un cambiador de herramientas de ubicación fija se ignora el número *P* (pero se lee, se conserva y se reescribe), por lo que puede usar P para cualquier número que quiera.

Cambiadores de herramientas de ubicación aleatoria. Los cambiadores de herramientas de ubicación aleatoria intercambian la herramienta en el eje con una del cambiador. Con este tipo de cambiador de herramientas, la herramienta siempre esta en un ranura diferente después de un cambio de herramienta. Cuando una herramienta se cambia, LinuxCNC reescribe el número de ranura para realizar un seguimiento de dónde están las herramientas. T puede ser cualquier número pero P debe ser un número que tenga sentido para la máquina.

6.1.3. Compensación del cortador

La compensación de cortador permite al programador programar la trayectoria de la herramienta sin conocer el diámetro exacto de la misma. La única advertencia es que el programador debe programar el movimiento de entrada para que sea al menos tan largo como el radio de herramienta más grande que podría usarse.

Hay dos caminos posibles que el cortador puede tomar segun la compensación esté en el lado izquierdo o derecho de una línea en la dirección del movimiento vista desde detrás del cortador. Para visualizar esto, imagine que está subido en la pieza caminando detrás de la herramienta mientras progresa su corte. G41 es su lado izquierdo de la línea y G42 es el lado derecho de la línea.

El punto final de cada movimiento depende del siguiente movimiento. Si el siguiente movimiento crea una esquina exterior, el movimiento será hasta el punto final de la línea de corte compensada. Si el siguiente movimiento crea una esquina interior el el movimiento se detendrá brevemente para no dañar la pieza. La siguiente figura muestra cómo el movimiento compensado se detendrá en diferentes puntos dependiendo del próximo movimiento

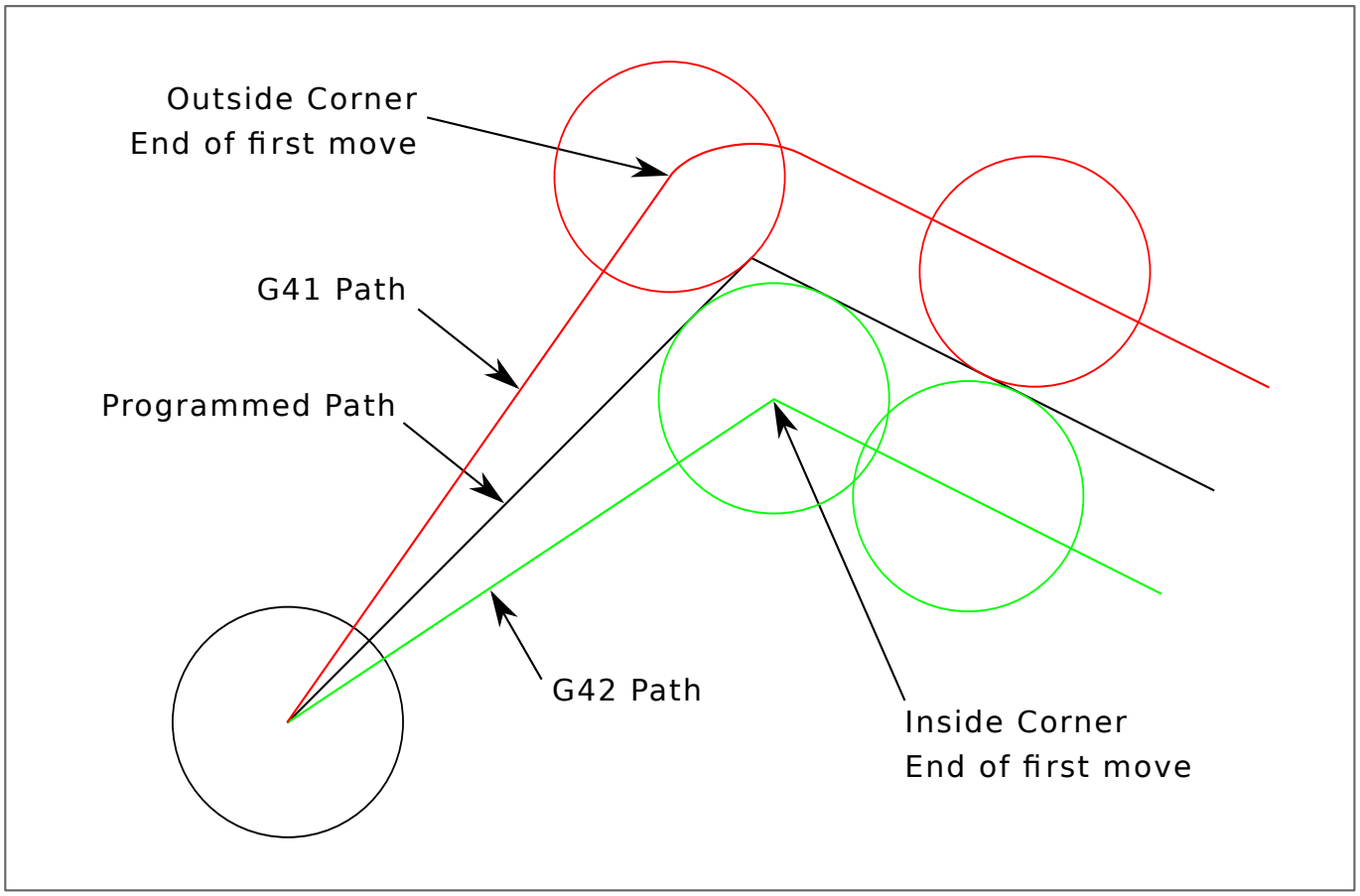


Figura 6.2: Punto final de compensación

6.1.3.1. Descripción general

Tabla de herramientas La compensación del cortador utiliza los datos de la tabla de herramientas para determinar el desplazamiento necesario. Los datos se pueden configurar en tiempo de ejecución con G10 L1

Programación de movimientos de entrada Cualquier movimiento que sea lo suficientemente largo como para realizar la compensación funcionará como movimiento de entrada. La longitud mínima es el radio de corte. Esto puede ser un movimiento rápido sobre la pieza de trabajo. Si se emiten varios movimientos rápidos después de un G41/42 solo el último moverá la herramienta a la posición compensada.

En la siguiente figura puede ver que el movimiento de entrada se compensa a la derecha de la línea. Esto coloca el centro de la herramienta a la derecha de X0 en este caso. Si tuviera que programar un perfil y el final está en X0, el perfil resultante dejaría una protuberancia debido al offset del movimiento de entrada.

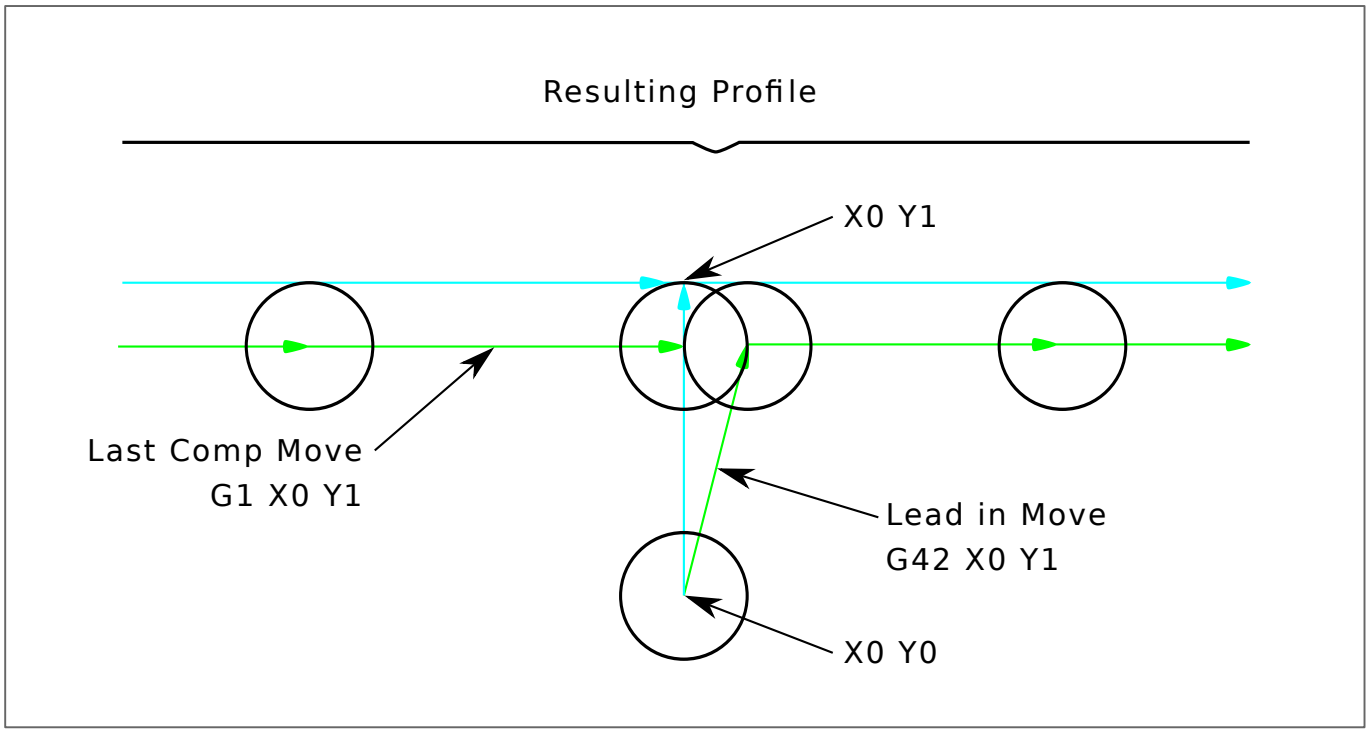


Figura 6.3: Movimiento de entrada

Movimientos Z El movimiento del eje Z puede tener lugar mientras se sigue el contorno en el plano XY. Se pueden omitir partes del contorno retrayendo el eje Z sobre la pieza y extendiendo el eje Z en el siguiente punto de inicio.

Movimientos rápidos Se pueden programar movimientos rápidos mientras la compensación está activada.

Buenas practicas - Inicie un programa con G40 para asegurarse de que la compensación esté desactivada.

6.1.3.2. Ejemplos

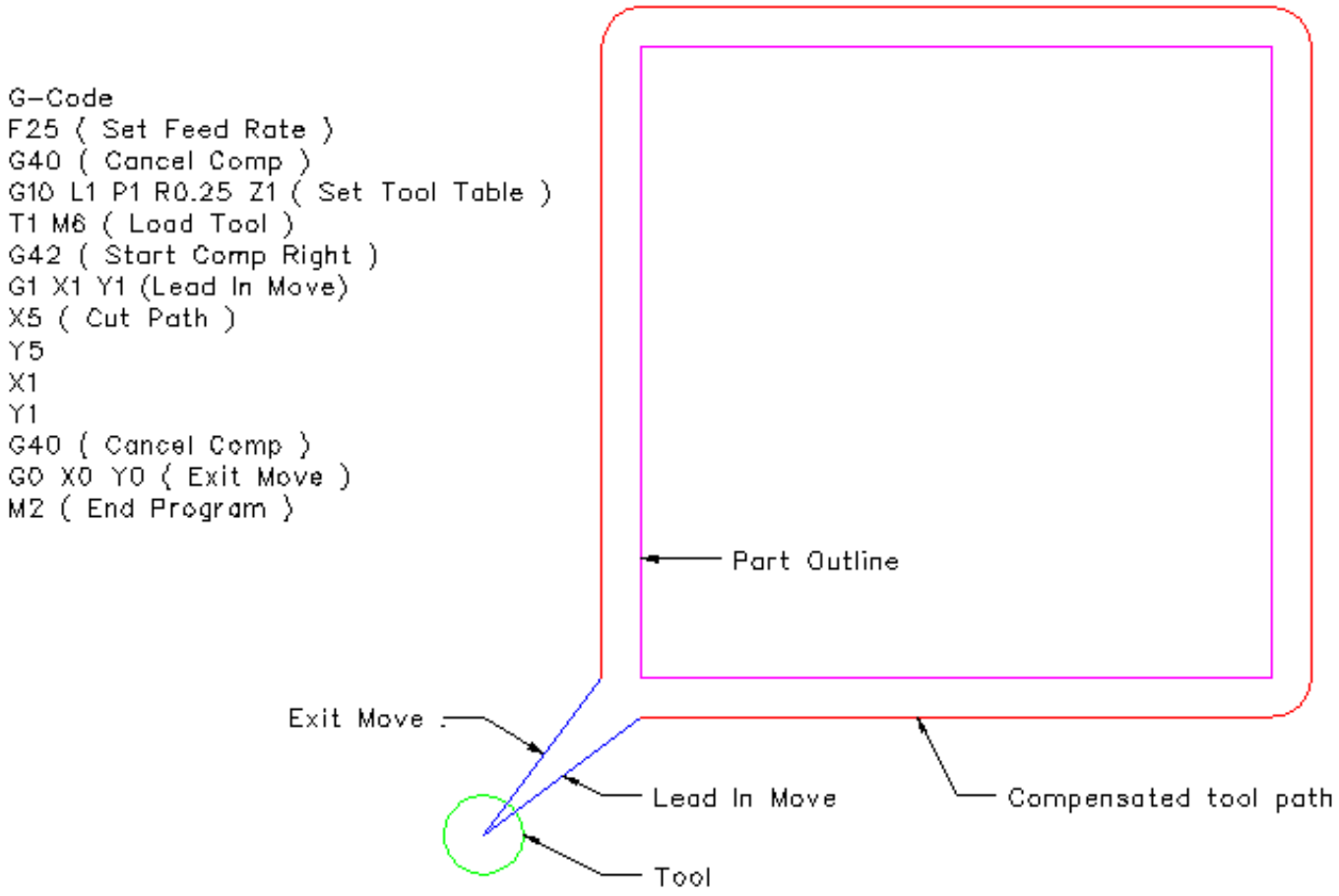


Figura 6.4: Perfil externo

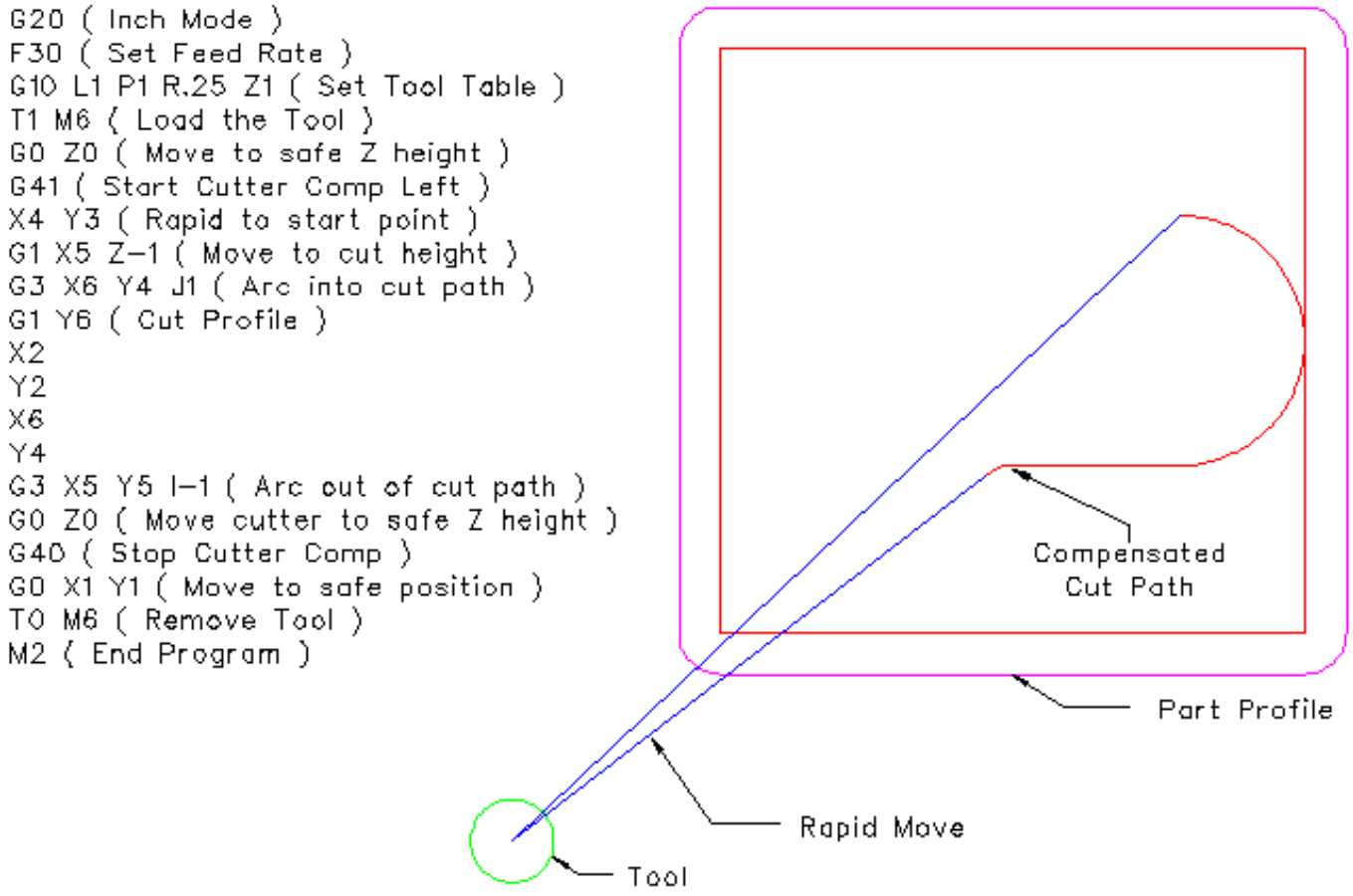


Figura 6.5: Perfil interno

6.2. GUI del Editor de Herramientas

6.2.1. Descripción general

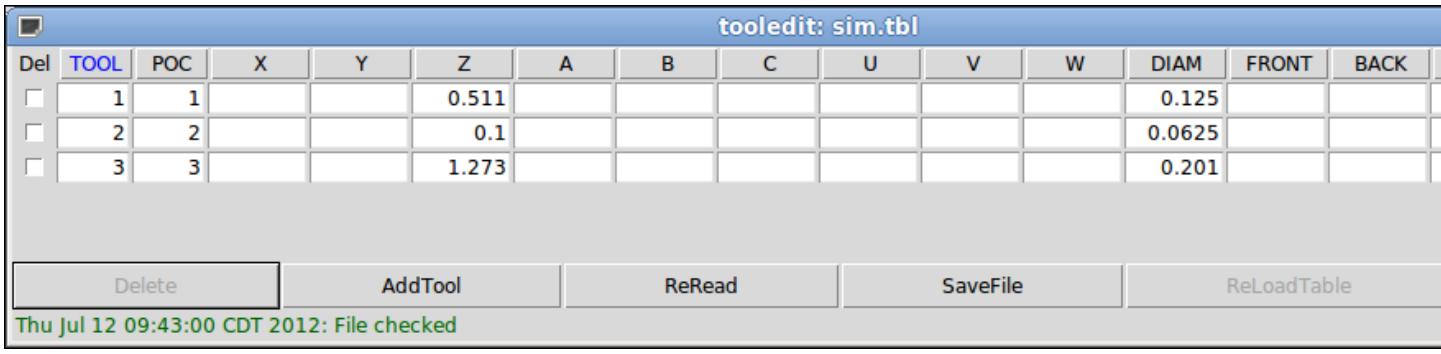


Figura 6.6: Tabla de herramientas

El programa *tooledit* puede actualizar el archivo de la tabla de herramientas con los cambios editados usando el botón Guardar Archivo. El botón Guardar Archivo actualiza el archivo del sistema pero se requiere una acción separada para actualizar los datos de la tabla de herramientas utilizados por una instancia de LinuxCNC en ejecución. Con la GUI Axis, tanto el archivo como los datos de la tabla de herramientas actuales utilizado por LinuxCNC se puede actualizar con el botón ReloadTable. Este botón está habilitado solo cuando la máquina está encendida e inactiva.

6.2.2. Ordenación por columnas

La visualización de la tabla de herramientas se puede ordenar por cualquier columna en forma ascendente haciendo clic en el encabezado de la columna. Un segundo clic hace la clasificacion en orden descendente. La clasificación de columnas requiere que la máquina se configure con la versión tcl mayor o igual a la 8.5.

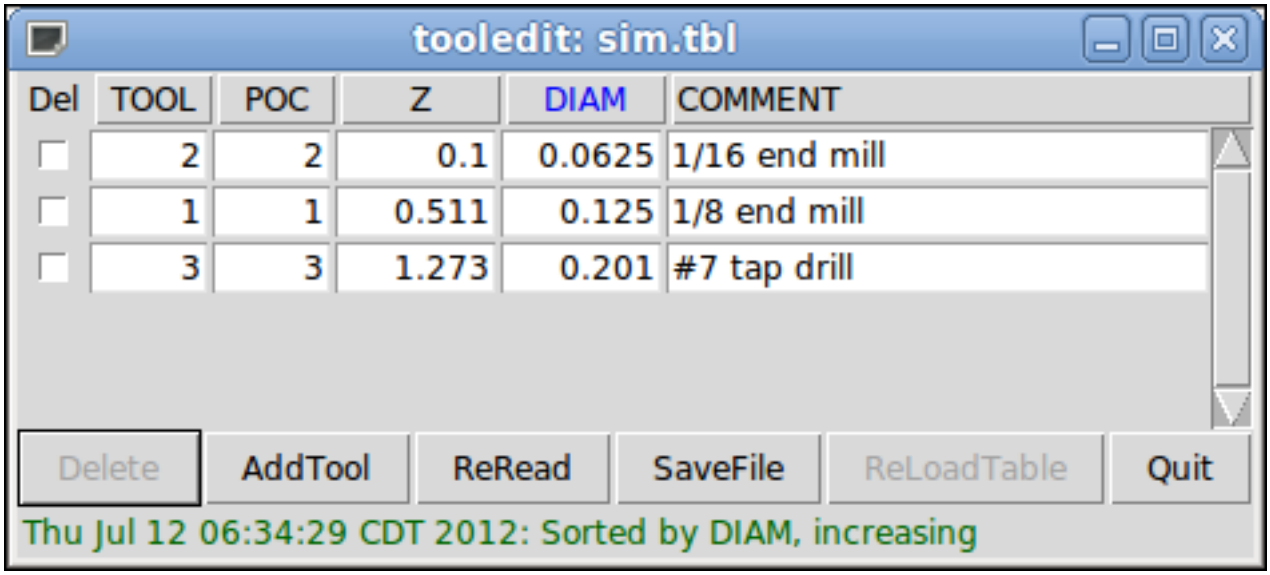


Figura 6.7: Clasificacion segun DIAM

Dependiendo de otras aplicaciones instaladas en el sistema, puede ser necesario habilitar tcl/tk8.5 con los comandos:

```
sudo update-alternative --config tclsh # selecciona la opción para tclsh8.5
sudo update-alternative --config wish  # selecciona la opción para wish8.5
```

6.2.3. Selección de columnas

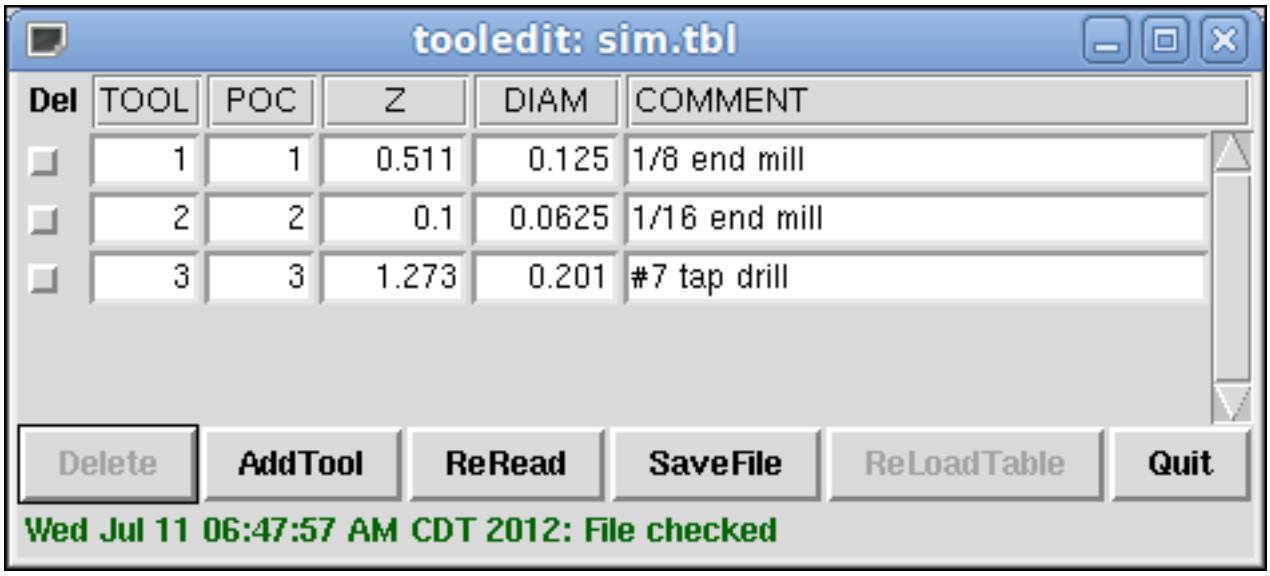
Por defecto, el programa *tooledit* mostrará todas las posibles columnas de parámetros de la tabla de herramientas. Dado que pocas máquinas utilizan todos los parámetros, las columnas mostradas pueden ser limitadas con el siguiente ajuste de archivo ini:

Sintaxis en archivo .INI

```
[DISPLAY]
TOOL_EDITOR = tooledit column_name column_name ...
```

Ejemplo para columnas Z y DIAM

```
[DISPLAY]
TOOL_EDITOR = tooledit Z DIAM
```



6.2.4. Uso independiente

El programa *tooledit* también se puede invocar como un programa independiente. Por ejemplo, si el programa está en la ruta de acceso del usuario, escribiendo *tooledit* se mostrará la sintaxis de uso:

```
Uso:
       tooledit nombre de archivo
       tooledit [column_1 ... column_n] nombre de archivo

Los nombres de columna permitidos son: x y z a b c u v w diam front back orien
```

Para sincronizar *tooledit* independiente con un LinuxCNC en ejecución, el nombre de archivo debe ser el mismo que el especificado en [EMCIO]TOOL_TABLE del archivo ini de LinuxCNC.

Cuando se usa el programa *tooledit* mientras se está ejecutando LinuxCNC, la ejecución de comandos gcode u otros programas pueden alterar el archivo de tabla de herramientas. Los cambios de archivos son detectados por *tooledit* y se muestra un mensaje:

```
Advertencia: Archivo modificado por otro proceso
```

La pantalla de la tabla de herramientas *tooledit* se puede actualizar para leer el archivo modificado con el botón Releer.

La tabla de herramientas se especifica en el archivo ini con una entrada:

```
[EMCIO]TOOL_TABLE = tool_table_filename
```

El archivo de la tabla de herramientas se puede editar con cualquier editor de texto simple (no un procesador de textos).

La gui Axis puede usar opcionalmente una configuración de archivo ini para especificar el programa editor de herramientas

```
[DISPLAY]TOOL_EDITOR = path_to_editor_programa
```

Por defecto, se usa el programa llamado *tooledit*. Este editor es compatible con todos los parámetros de la tabla de herramientas, permite la adición y eliminación de entradas de herramientas, y realiza una serie de verificaciones de validez en valores paramétricos.

Parte IV

Configuracion

Capítulo 7

Informacion General

7.1. Conceptos para Integradores

7.1.1. Ubicaciones de archivos

LinuxCNC busca la configuración y los archivos de código G en un lugar específico. La ubicación depende de cómo ejecute LinuxCNC.

7.1.1.1. LinuxCNC Instalado

Si está ejecutando LinuxCNC desde el LiveCD o si lo instaló a través de un deb y usa el selector de configuración *LinuxCNC* del menú LinuxCNC, mire en los siguientes directorios:

- El directorio LinuxCNC se encuentra en */home/dir-de-usuario/linuxcnc*.
- Los directorios de configuración se encuentran en */home/dir-de-usuario/linuxcnc/configs*.
  - Los archivos de configuración se encuentran en */home/dir-de-usuario/linuxcnc/configs/name-of-config*.
- Los archivos de código G se encuentran en /home/dir-de-usuario/linuxcnc/nc_files ’.

Por ejemplo, para una configuración llamada mill y nombre de usuario Fred, los directorios y la estructura de archivos se vería así.

- */home/fred/linuxcnc*
- */home/fred/linuxcnc/nc_files*
- */home/fred/linuxcnc/configs/mill*
  - */home/fred/linuxcnc/configs/mill/mill.ini*
  - */home/fred/linuxcnc/configs/mill/mill.hal*
  - */home/fred/linuxcnc/configs/mill/mill.var*
  - */home/fred/linuxcnc/configs/mill/tool.tbl*

7.1.1.2. LinuxCNC desde la Línea de Comandos

Si ejecuta LinuxCNC desde la línea de comandos y especifica el nombre y la ubicación del archivo INI, las ubicaciones de los archivos pueden estar en un lugar diferente. Para ver las opciones para ejecutar LinuxCNC desde la línea de comandos ejecute *linuxcnc -h*.

nota
Las ubicaciones opcionales para algunos archivos se pueden configurar en el archivo INI. Ver las secciones DISPLAY y RS274NGC.

7.1.2. Archivos

Cada directorio de configuración requiere al menos los siguientes archivos:

- Un archivo INI .ini
- Un archivo HAL .hal o un archivo HALTCL .tcl especificado en la sección HAL del archivo INI.

nota
Otros archivos pueden ser necesarios para algunas GUI.

Opcionalmente también puede tener:

- Un archivo de variables .var
  - Si omite un archivo .var en un directorio pero incluye [RS274NGC]PARAMETER_FILE=algunFichero.var, el archivo se creará para usted cuando se inicie LinuxCNC.
  - Si omite un archivo .var y omite el elemento [RS274NGC]PARAMETER_FILE, se creará el archivo var rs274ngc.var cuando se inicie LinuxCNC. Puede haber algunos mensajes confusos si se omite [RS274NGC]PARAMETER_FILE.
- Un archivo de tabla de herramientas .tbl si [EMCMOT]TOOL_TABLE ha sido especificado en el archivo INI. Algunas configuraciones no necesitan una tabla de herramientas.

7.1.3. Sistemas steppers

7.1.3.1. Período Base

BASE_PERIOD es el *latido* de su sistema LinuxCNC. [1] En cada período, el generador de pasos software decide si es hora de otro pulso de paso. Un período más corto le permitirá generar más pulsos por segundo, dentro de los limites. Pero si se hace muy corto, la computadora gastará mucho mucho tiempo generando pulsos de paso y todo lo demás se relentizara, o tal vez incluso se bloqueara. Los requisitos de latencia y de los drivers afectarán el período más corto que pueda usar.

Las latencias, en el peor de los casos, solo pueden ocurrir unas pocas veces por minuto, y las probabilidades de que ocurra una mala latencia justo cuando el motor está cambiando de dirección son bajas. Por lo tanto, puede obtener errores muy raros que arruinan una pieza de vez en cuando y son imposibles de solucionar.

La forma más sencilla de evitar este problema es elegir un BASE_PERIOD que sea la suma del requisito de tiempo más largo de su driver, y el peor caso de latencia de su computadora. Esto no siempre es la mejor opción. Por ejemplo, si está usando un driver con un requisito de tiempo de retención de señal de dirección de 20 us, y su prueba de latencia dice que tiene una latencia máxima de 11 us, entonces si configura BASE_PERIOD en 20 + 11 = 31 us obtendrá unos 32,258 pasos por segundo en un modo y 16,129 pasos por segundo en otro modo. Estos numeros son demasiado bajos.

[1] Esta sección se refiere al uso de **stepgen**, generador de pasos incorporado en LinuxCNC. Algunos dispositivos de hardware tienen su propio generador de pasos y no usan el incorporado de LinuxCNC. En ese caso, consulte su manual de hardware.

El problema está con el requisito de tiempo de espera de 20 us. Eso, más los 11 us de latencia es lo que nos obliga a usar un período lento de 31 us. Pero el generador de pasos de software LinuxCNC tiene algunos parámetros que le permiten aumentar de un período a varios. Por ejemplo, si *steplen* [2] es cambiado de 1 a 2, entonces habrá dos períodos entre el principio y final del pulso de paso. Del mismo modo, si *dirhold* [3] es cambiado de 1 a 3, habrá al menos tres períodos entre el pulso de paso y un cambio del pin de dirección.

Si podemos usar *dirhold* para cumplir con el requisito de tiempo de espera de 20 us, entonces el siguiente tiempo más largo es el tiempo alto de 4.5 us. Agregue la latencia de 11 us a los 4.5 us de tiempo en alto, y se obtiene un período mínimo de 15.5 us. Cuando se intenta 15.5 us, se vera que la computadora es lenta, por lo que se ajustará a 16 us. Si dejamos *dirhold* en 1 (el valor predeterminado), entonces el tiempo mínimo entre paso y dirección es el período de 16 us menos la latencia de 11 us = 5 us, lo cual no es suficiente, Necesitamos otros 15 us. Como el período es de 16 us, se necesita un período más. Entonces cambiamos *dirhold* de 1 a 2. Ahora el mínimo tiempo desde el final del pulso de paso hasta el cambio del pin de dirección es 5 + 16 = 21 us, y no tenemos que preocuparnos de que el driver tome una dirección incorrecta debido a la latencia.

Para obtener más información sobre stepgen, consulte la sección stepgen.

7.1.3.2. Temporizacion de pasos

Step Timing y Step Space son diferentes en algunos drivers. En este caso el punto de paso se vuelve importante. Si el driver manda pasos en borde descendente, entonces el pin de salida debe invertirse.

7.1.4. Servosistemas

7.1.4.1. Operación básica

Los servosistemas tienen mayor velocidad y precisión que los sistemas paso a paso equivalentes, pero son más costosos y complejos. A diferencia de los sistemas paso a paso, los servosistemas requieren algún tipo de dispositivo de retroalimentación de posición, y debe ajustarse o *tunearse*, ya que no funcionan directamente como un sistema paso a paso. Estas diferencias existen porque los servos son sistemas de *lazo cerrado*, a diferencia de los motores paso a paso que generalmente funcionan en *lazo abierto*. Que significa *lazo cerrado*?. Veamos un diagrama simplificado de cómo está conectado un sistema servomotor.

[2] steplen se refiere a un parámetro que ajusta el rendimiento del generador de pasos incorporado de LinuxCNC, *stepgen*, que es un componente HAL. Este parámetro ajusta la longitud del pulso de paso en sí mismo. Sigue leyendo, todo se explicará.

[3] dirhold se refiere a un parámetro que ajusta la duración del tiempo de retención de la dirección.

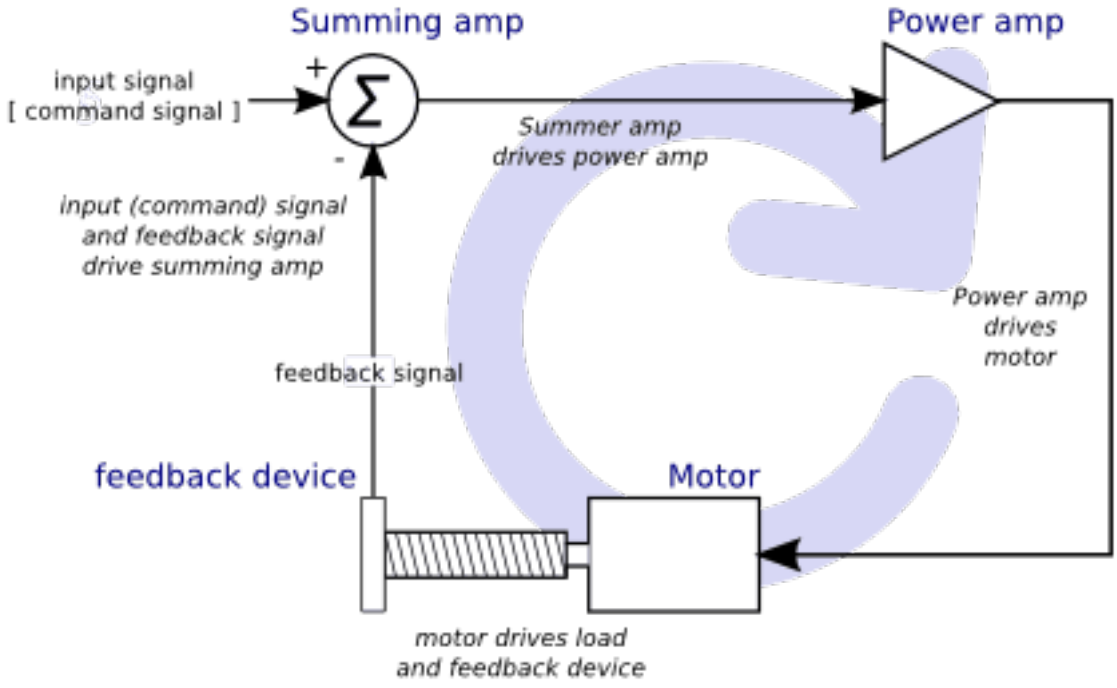


Figura 7.1: Servo Loop

Este diagrama muestra que la señal de entrada (y la señal de retroalimentación) alimentan un amplificador sumador, el amplificador sumador acciona el amplificador de potencia, el amplificador de potencia impulsa el motor, el motor impulsa la carga (y el dispositivo de retroalimentación) y el dispositivo de retroalimentación (y la señal de entrada) vuelven a manejar el motor. Esto se parece mucho a un círculo (un lazo cerrado) donde A controla B, B controla C, C controla D y D controla A.

Si no ha trabajado antes con servosistemas, sin duda esto parecerá una idea muy extraña al principio, especialmente en comparación con la electrónica normal de circuitos donde las entradas preceden a las salidas y nunca van hacia atras. [4] Si *todo* controla *todo lo demás*, cómo puede funcionar eso; ¿quién está a cargo?. La respuesta es que LinuxCNC *puede* controlar este sistema, pero tiene que hacerlo mediante algun método de control. El método de control que utiliza LinuxCNC, uno de los más simples y mejores, se llama PID.

PID significa Proporcional, Integral y Derivativo. El valor proporcional determina la reacción al error actual, el valor integral determina la reacción en función de la suma de los errores recientes, y el valor derivado determina la reacción en función de la velocidad a la que el error ha estado cambiando. Son tres técnicas matemáticas comunes que se aplican a la tarea de lograr que un proceso de trabajo sigan un punto de trabajo o consigna. En el caso de LinuxCNC, el proceso que queremos controlar es la posición real del eje y el punto de trabajo o consigna es la posición ordenada del eje.

[4] Si sirve de ayuda, el equivalente más cercano a esto en el mundo digital son las *máquinas de estado*, *máquinas secuenciales* y demás, donde lo que están haciendo las salidas *ahora*, depende de lo que las entradas (y las salidas) estaban haciendo *antes*. Si no ayuda, no importa.

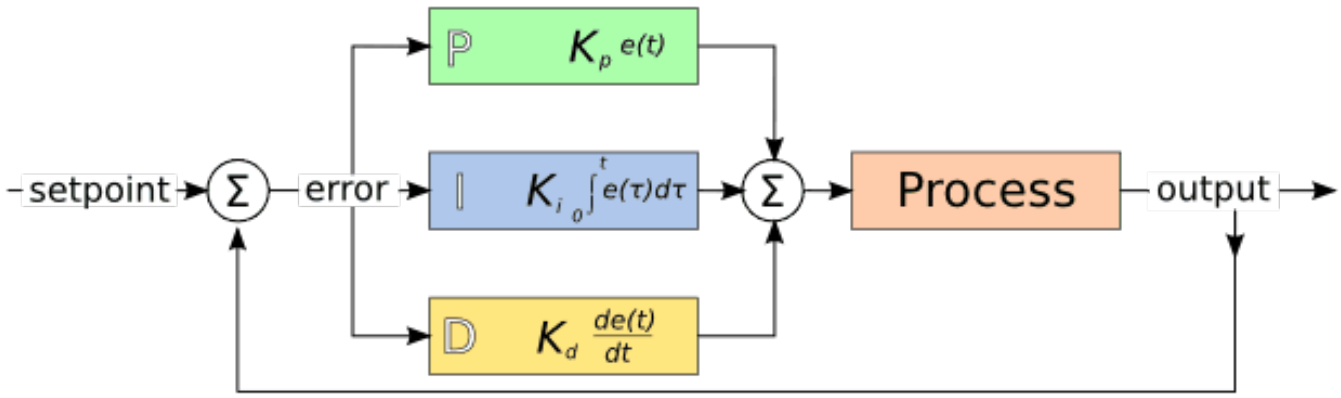


Figura 7.2: PID Loop

Al *ajustar* las tres constantes en el algoritmo del controlador PID, el el controlador puede proporcionar acciones de control diseñadas para requisitos de procesos específicos. La respuesta del controlador puede describirse en términos de la capacidad de respuesta frente a un error, el grado con el que el controlador sobrepasa el punto de ajuste (overshoot) y el grado de oscilación del sistema.

7.1.4.2. Término proporcional

El término proporcional (a veces llamado ganancia) hace un cambio en la salida que es proporcional al valor de error actual. Una ganancia proporcional alta resulta en un gran cambio en la salida para un determinado cambio en el error. Si la ganancia proporcional es demasiado alta, el sistema puede volverse inestable. En contraste, una pequeña ganancia resulta en una pequeña respuesta de salida a un gran error de entrada. Si la ganancia proporcional es demasiado baja, la acción de control puede ser demasiado pequeña al responder el sistema a los disturbios.

En ausencia de perturbaciones, el control proporcional puro no puede estabilizarse en su valor objetivo, pero retendrá un error de estado estable que es una función de la ganancia proporcional y la ganancia del proceso. A pesar del offset en estado estacionario, tanto la teoría de afinacion como la práctica industrial indican que es el término proporcional el que debe contribuir en mayor parte al cambio de la salida.

7.1.4.3. Término integral

La contribución del término integral (a veces llamado reset) es proporcional tanto a la magnitud del error como a la duración del mismo. Sumando el error instantáneo en el tiempo (integrando el error) se proporciona el offset acumulado que debería haberse corregido previamente. El error acumulado se multiplica por la ganancia integral y se agrega a la salida del controlador.

El término integral (cuando se agrega junto al término proporcional) acelera el movimiento del proceso hacia el punto de ajuste y elimina el error residual de estado estacionario que ocurre con una controlador solo proporcional. Sin embargo, dado que el término integral responde a errores acumulados del pasado, puede causar que el valor presente sobrepase el valor del punto de ajuste (cruzar sobre el punto de ajuste y luego crear una desviación en la otra dirección).

7.1.4.4. Término derivado

La tasa de cambio del error de proceso se calcula determinando la pendiente del error en el tiempo (es decir, su primera derivada con respecto al tiempo) y multiplicando esta tasa de cambio por la ganancia derivativa.

El término derivado reduce la velocidad de cambio de la salida del controlador y este efecto es más notable cerca del punto de ajuste del controlador. Por lo tanto, el control derivativo se utiliza para reducir la magnitud del sobreimpulso (overshoot) producido por el componente integral y mejorar estabilidad de la combinación proceso / controlador.

7.1.4.5. Ajuste de bucle

Si los parámetros del controlador PID (las ganancias de los términos proporcional, integral y derivado) se eligen incorrectamente, la entrada controlada del proceso puede ser inestable, es decir, su salida diverge, con o sin oscilación, y está limitada solo por saturación o rotura mecánica. Sintonizar un bucle de control es el ajuste de sus parámetros de control (ganancia / banda proporcional, ganancia integral / reset, ganancia derivada / tasa) a los valores óptimos para la respuesta de control deseada.

7.1.4.6. Sintonización manual

Un método de ajuste simple es establecer primero los valores I y D en cero. Aumente la P hasta que la salida del bucle oscile. Luego debe establecerse la P en aproximadamente la mitad de ese valor, buscando una respuesta del tipo *decaimiento de un cuarto de amplitud*. Despues se aumenta I hasta que cualquier offset se corrija en tiempo suficiente para el proceso. Sin embargo, demasiad a I causará inestabilidad. Finalmente, aumente D, si es necesario, hasta que el ciclo sea aceptablemente rápido para alcanzar su referencia después de una perturbación de carga. Sin embargo, demasiada D causará una respuesta excesiva y un sobredisparo. La sintonización del bucle PID generalmente sobredispara ligeramente para alcanzar el punto de ajuste más rapidamente; sin embargo, algunos sistemas no pueden aceptar el sobreimpulso, en cuyo caso se requiere un sistema de lazo cerrado *sobre-amortiguado*, que requerirá una P significativamente menor de la mitad de la configuración de P que causa oscilación.

7.1.5. RTAI

La interfaz de aplicación en tiempo real (RTAI) se utiliza para proporcionar el mejor rendimiento en tiempo real (RT). El kernel parcheado RTAI le permite escribir aplicaciones con restricciones de tiempo estrictas. RTAI da la habilidad tener cosas como la generación de pasos software que requieren gran precisión de sincronización.

7.1.5.1. ACPI

La Interfaz avanzada de configuración y energía (ACPI) tiene muchas funciones diferentes, la mayoría de las cuales interfieren con el rendimiento RT (por ejemplo: administración de energía, apagado de CPU, escala de frecuencia de CPU, etc.) El núcleo LinuxCNC (y probablemente todos los núcleos parcheados con RTAI) tiene ACPI deshabilitado. ACPI también se encarga de apagar el sistema después de que se ha iniciado el apagado, y por eso es posible que deba presionar el botón de encendido para apagar completamente su computadora. El grupo RTAI ha estado mejorando esto en versiones recientes, por lo que su sistema LinuxCNC puede apagarse en sí mismo después de todo.

7.2. Test de Latencia

Esta prueba es la primera que debe realizarse en un PC para ver si es capaz de manejar una máquina CNC.

La latencia es el tiempo que tarda un PC concreto en detener lo que está haciendo y responder a una solicitud externa. Para LinuxCNC, la solicitud es BASE_THREAD, que es el "latido" periódico que sirve como referencia de tiempo para los pulsos de paso. Cuanto menor es la latencia, más rápido puede ejecutar los "latidos", y más rápidos y más suaves seran los pulsos de paso.

La latencia es mucho más importante que la velocidad de la CPU. Un humilde Pentium II que responde siempre a las interrupciones en 10 us pueden dar mejores resultados que un P4 Hyperthreading.

La CPU no es el único factor para determinar la latencia. La placas base, tarjetas de video, puertos USB y un gran número de otros factores puede dañar la latencia. La mejor manera de averiguar con cuanta se esta tratando, es ejecutar la prueba RTAI de latencia.

Generar pulsos de paso en software tiene una gran ventaja; es gratis. Casi todas las PC admiten uno mas puertos paralelo que son capaces de emitir pulsos de pasos generados por el software. Sin embargo, los pulsos de paso software también tiene algunas desventajas:

- velocidad de paso máxima limitada
- jitter (fluctuaciones) en los pulsos generados

- añade carga a la CPU

La mejor forma de descubrir si su PC funcionará bien con LinuxCNC es ejecutar la prueba de latencia HAL. Para ejecutar la prueba, abra una ventana de terminal (En Ubuntu, desde Aplicaciones → Accesorios → Terminal. En Debian, Menu de Aplicaciones → Emulador de Terminal) y ejecute el siguiente comando:

```
latency-test
```

Debería ver algo como esto:

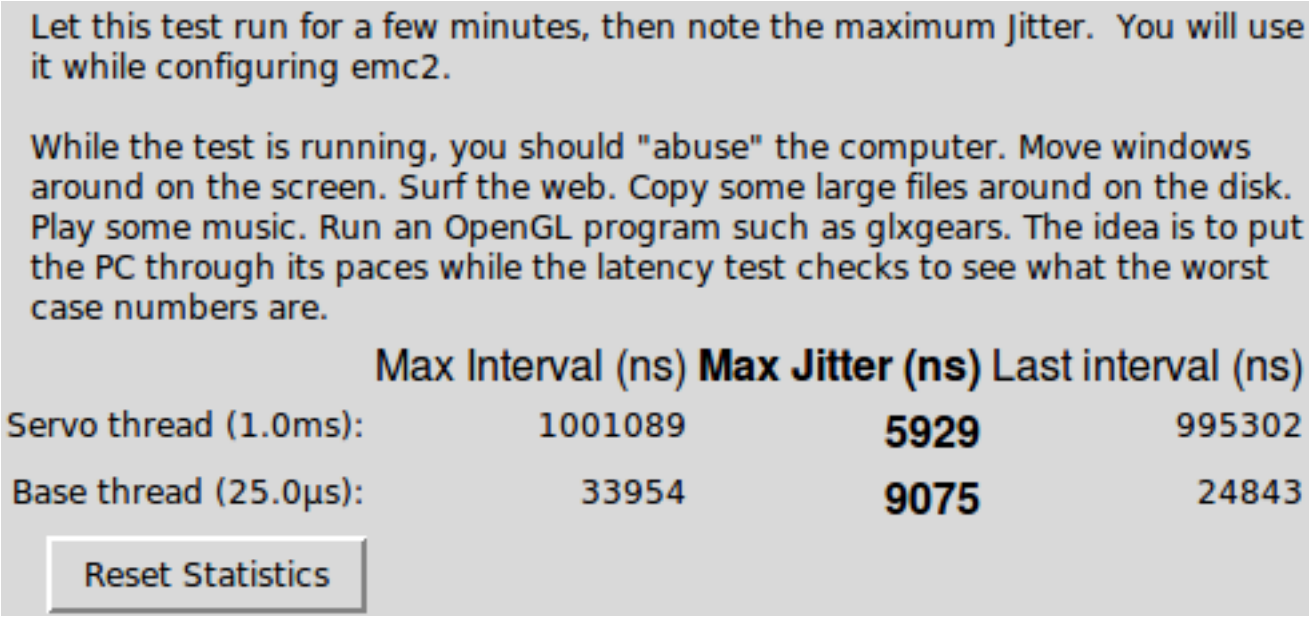


Figura 7.3: Prueba de Latencia HAL

Mientras se ejecuta la prueba, debe *abusar* de la computadora. Mueva las ventanas alrededor de la pantalla. Navege por la web. Copie algunos archivos grandes en el disco. Ponga musica. Ejecute un programa OpenGL como glxgears. La idea es poner al PC en aprietos mientras la prueba de latencia comprueba cuáles son los peores números.

nota
No ejecute LinuxCNC o Stepconf mientras se ejecuta la prueba de latencia.

Los números importantes son los de la columna "max jitter" (máxima fluctuación). En el ejemplo anterior, son 9075 nanosegundos, o 9,075 microsegundos. Anote este número e ingréselo en Stepconf cuando se le solicite.

En el ejemplo, la prueba de latencia solo se ejecutó durante unos segundos. Debe ejecutar la prueba durante al menos varios minutos; a veces el peor de los casos de latencia no ocurre con frecuencia, o solo ocurre cuando se verifica alguna acción particular. Por ejemplo, una placa madre Intel funcionaba bastante bien la mayor parte del tiempo, pero cada 64 segundos tenía una latencia muy mala de 300 us. Afortunadamente eso fue reparable; vea http://wiki.linuxcnc.org/cgi-bin/wiki.pl?FixingSMIIssues

Entonces, ¿qué significan los resultados?. Si su número de Max Jitter es menor de aproximadamente 15-20 microsegundos (15000-20000 nanosegundos), la computadora debería dar buenos resultados con pasos por software. Si la latencia máxima es de entre 30-50 microsegundos, todavía puede obtener buenos resultados, pero su tasa máxima de pasos podría ser un poco decepcionante, especialmente si usa microstepping o tiene tornillos de paso de paso fino. Si los números son 100 us o más (100,000 nanosegundos), entonces la PC no es un buen candidato para pasos por software. Números de más de 1 milisegundo (1,000,000 de nanosegundos) significan la PC no es un buen candidato para LinuxCNC, independientemente de si usa pasos por software o no.

Tenga en cuenta que si obtiene números demasiado altos, puede haber formas de mejorarlos. Otra PC tuvo muy mala latencia (varios milisegundos) cuando usaba el video incorporado. Una tarjeta de video usada de 5$ resolvió el problema.

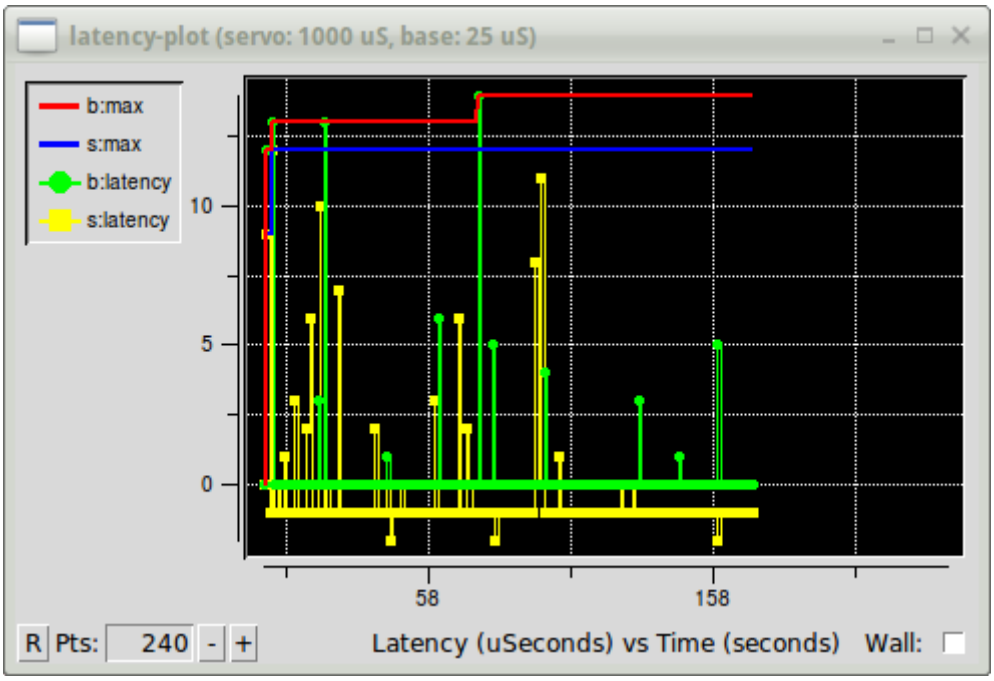
nota
LinuxCNC no requiere hardware de última generación.

Para obtener más información sobre la sintonización de pasos, consulte el capítulo Sintonización de steppers.

Hay herramientas adicionales disponibles desde la línea de comandos para examinar la latencia cuando LinuxCNC no está corriendo.

latency-plot graba un registro de los hilos base y servo. Puede ser útil para ver picos de latencia cuando se inician o se usan otras aplicaciones. Uso:

```
latency-plot --help

Uso:
      latency-plot --help | -?
      latency-plot --hal [Opciones]

Opciones:
      --base nS  (intervalo de hilo base, valor predeterminado: 25000)
      --servo nS (intervalo de hilo servo, por defecto: 1000000)
      --time mS  (intervalo de informe, valor predeterminado: 1000)
      --relative (tiempo relativo de reloj (predeterminado))
      --actual   (hora actual del reloj)
```



latency-histogram muestra un histograma de latencia (jitter) para hilos base y servo.

```
Uso:
   latency-histogram --help | -?
o
   latency-histogram [Opciones]

Opciones:
  --base nS        (intervalo de hilo base, predeterminado: 25000, min: 5000)
  --servo nS       (intervalo de hilo servo, predeterminado: 1000000, min: 25000)
  --bbinsize nS    (tamaño del contenedor base, valor predeterminado: 100)
  --sbinsize nS    (tamaño del contenedor servo, predeterminado: 100)
```

```
  --bbins n        (contenedores base, por defecto: 200)
  --sbins n        (servo bins, por defecto: 200)
  --logscale 0|1   (escala logaritmica del eje y, valor predeterminado: 1)
  - text      note (nota adicional, valor predeterminado: "")
  --show           (muestra el recuento de los contenedores no mostrados)
  --nobase         (hilo servo solamente)
  --verbose        (progreso y depuración)
  --nox            (sin gui, display transcurrido, min, max, sdev para cada hilo)

Notas:
  Linuxcnc y Hal no deberían estar ejecutándose; deténgalos con halrun -U.
  Una gran cantidad de contenedores y/o binsizes pequeños ralentizarán las actualizaciones.
  Para un solo hilo, especifique --nobase (y opciones para el hilo servo).
  Las latencias medidas fuera del rango +/- bin se informan
  con barras finales especiales. Use --show para mostrar el conteo de
  bins fuera del grrafico [pos | neg]
```

image ::../config/images/latency-histogram.png [alt = "latency-histogram muestra un histograma de latencia (jitter) para hilos base y servo"]

7.3. Afinación de Steppers

7.3.1. Obtener el máximo rendimiento del Software de Stepping

Generar pulsos de paso en el software tiene una gran ventaja: es gratis. Casi todas las PC tienen un puerto paralelo que es capaz de generar pulsos de paso por el software. Sin embargo, los pulsos de paso de software también tienen algunas desventajas:

- velocidad de paso máxima limitada
- inestabilidad en los pulsos generados
- carga la CPU

Este capítulo tiene algunos pasos que pueden ayudarlo a obtener los mejores resultados a partir de pasos generados por software.

7.3.1.1. Ejecutar una prueba de latencia

Ejecute la prueba de latencia como se describe en el capítulo Test de Latencia.

Mientras se ejecuta la prueba, debe *abusar* de la computadora. Mueva ventanas alrededor de la pantalla. Navegue por la web. Copie algunos archivos grandes en el disco. Ponga musica. Ejecute un programa OpenGL como Glxgears. La idea es poner a la PC a un fuerte ritmo mientras el test de latencia verifica los resultados para ver cuáles son los peores números.

El último número en la columna etiquetada *Max Jitter* es el más importante. Reservelo; lo necesitarás más tarde. Contiene la peor latencia medida durante toda la ejecución de la prueba. En el ejemplo anterior, eso es 10636 nanosegundos, o 10.6 microsegundos, que es excelente. Sin embargo, el ejemplo solo se ejecutó durante unos segundos (imprime una línea cada segundo). Debe ejecutar la prueba durante al menos varios minutos; a veces, la peor latencia no ocurre muy a menudo, o solo sucede cuando se hace alguna acción particular. Cierta placa base Intel funcionó bastante bien la mayor parte del tiempo, pero cada 64 segundos tenía una muy mala latencia de 300 us. Afortunadamente, eso es reparable, ver FixingDapperSMIIssues en la wiki que se encuentra en wiki.linuxcnc.org.

Entonces, ¿qué significan los resultados? Si su número *ovl max* es menor que aproximadamente 15-20 microsegundos (15000-20000 nanosegundos), la computadora debería da buenos resultados con pasos por software. Si la latencia máxima esta entre 30-50 microsegundos, todavía puede obtener buenos resultados, pero su la velocidad máxima de paso puede ser un poco decepcionante, especialmente si utiliza microstepping o tiene tornillos de paso muy finos. Si los números son 100 us o más (100,000 nanosegundos), entonces ese PC no es buen candidato para pasos por software. Números de más de 1 milisegundo (1,000,000 nanosegundos) significa que la PC no es un buen candidato para EMC, independientemente de que use pasos por software o no.

Tenga en cuenta que si obtiene números altos, puede haber formas de mejorarlos. Por ejemplo, una PC tenía una latencia muy mala (varios milisegundos) usando el video integrado. Pero una tarjeta de video Matrox usada de $5 resolvió el problema problema: EMC no requiere hardware de última generación.

7.3.1.2. Descubra lo que esperan sus drivers

Las diferentes marcas de unidades paso a paso tienen diferentes requisitos de temporización en sus pasos y entradas de dirección. Necesita la hoja de datos que tiene las especificaciones de su unidad.

Del manual de Gecko G202:

```
Frecuencia de pasos: 0 a 200 kHz
Step Pulse "0" Time: 0.5 us min (Paso en el flanco descendente)
Step Pulse "1" Time: 4.5 us min
Configuración de dirección: 1 us min (20 us min tiempo de retencion despues del ↩
   flanco de paso)
```

Del manual de Gecko G203V:

```
Frecuencia de paso: 0 a 333 kHz
Step Pulse "0" Time: 2.0 us min (paso en flanco ascendente)
Step Pulse "1" Time: 1.0 us min

Configuración de dirección:
    200 ns (0.2 us) antes del flanco ascendente del pulso del paso
    200 ns (0.2 us) se mantienen después del paso del pulso flanco ascendente
```

De la hoja de datos de Xylotex:

```
Tiempo de configuración de DIR mínimo antes del flanco ascendente de STEP Pulse ↩
   200 ns Mínimo
Tiempo de retención DIR después del flanco ascendente del impulso STEP 200 ns
Pulso STEP mínimo tiempo alto 2.0 us
Minimum STEP pulso bajo tiempo 1.0 us
El paso sucede en el flanco ascendente
```

Una vez que encuentre los números, anótelos también; los necesita en el próximo paso.

7.3.1.3. Eligir su BASE_PERIOD

BASE_PERIOD es el "latido" de su computadora EMC. En cada período, el generador de pasos de software decide si es el momento de otro impulso de pasos. Un período más corto le permitirá generar más pulsos por segundo, dentro de los limites. Pero si se queda demasiado corto, su computadora gastará mucho tiempo generando pulsos de paso que todo lo demás se ralentizará, o tal vez incluso se bloqueara. Los requisitos de latencia y del driver afectan el período más corto que se puede usar, como veremos en un minuto.

Veamos primero el ejemplo de Gecko. El G202 puede manejar pulsos de pasos que estan en bajo 0.5 us y y en alto 4.5 us. Necesita el pin de dirección estable durante 1us antes del flanco descendente y permanecer estable 20us después del flanco descendente. El requisito de tiempo más largo es el tiempo de espera de 20 us. Un enfoque simple sería establecer el período en 20 us. Eso significa que todos los cambios en las líneas STEP y DIR están separados por 20 us. Todo bien; de acuedo?

¡Incorrecto! Si hubiera latencia CERO, entonces todos los flancos estarían separados por 20 us, y todo estaría bien. Pero todas las computadoras tienen alguna latencia Latencia significa retraso. Si la computadora tiene 11 us de latencia, eso significa que a veces el software corre 11 us más tarde de lo que se suponía. Si una ejecución del software se hace 11 us tarde, y la siguiente se hace en su momento, el retraso de la primera a la segunda es solo 9 us. Si el primero generó un impulso de paso, y el segundo cambia el bit de dirección, acaba de violar el requisito de tiempo de espera de 20 us del G202. Eso significa que su unidad podría haber dado un paso en la dirección incorrecta, y su pieza será de tamaño incorrecto.

La parte realmente desagradable de este problema es que puede ser muy esporadico. Las peores latencias pueden ocurrir solo unas pocas veces por minuto, y las probabilidades de mala latencia justo cuando el motor está cambiando la dirección es baja. Así que se obtienes errores muy extraños que arruinan una pieza de vez en cuando y parecen imposibles de solucionar.

La forma más sencilla de evitar este problema es elegir un BASE_PERIOD que sea la suma de los requisitos de tiempo más largos de su driver, y el el peor caso de latencia de su computadora. Si está ejecutando un Gecko con 20 us como requisito de

tiempo de mantenimimiento, y su prueba de latencia dice que tiene una latencia máxima de 11 us, si configura BASE_PERIOD a 20+11=31 us (31000 nano-segundos en el archivo ini), tiene la garantía de cumplir los requisitos de tiempo del driver.

Pero queda un compromiso. Hacer un pulso de paso requiere al menos dos períodos. Uno para comenzar el pulso y otro para finalizarlo. Como el período es 31 us, se necesita 2x31 = 62 us para crear un pulso de paso. Eso significa que la velocidad máxima de paso es de solo 16129 pasos por segundo. No muy buena. Pero no se rindas todavía, todavía tenemos algunos ajustes que hacer en la próxima sección.

Para Xylotex, los tiempos de activacion y mantenimiento son muy cortos, 200 ns cada uno (0.2 us). El tiempo más largo es el tiempo 2 us. Si tienes 11 us latencia, entonces puede establecer BASE_PERIOD como 11+2=13 us. Conseguir deshacerse del largo tiempo de espera de 20 us realmente ayuda! Con un período de 13 us, un paso completo tarda 2x13=26 us, y la velocidad máxima de paso es 38461 pasos por segundo!

Pero no puede celebrar nada todavía. Tenga en cuenta que 13 us es un período muy corto. Si intenta correr el generador de pasos cada 13 us, podría suceder no le quedará tiempo suficiente para ejecutar cualquier otra cosa, y su computadora podria bloquearse. Si su objetivo son períodos de menos de 25 us, debe comenzar con 25 us o más, ejecute EMC y vea cómo responden las cosas. Si todo esta bien, puedes disminuir gradualmente el período. Si el puntero del mouse comienza a ser lento, o cualquier otra cosa en el PC se ralentiza, su el período es un poco corto. Regrese al valor anterior que deje ala computadora funcionar sin problemas.

En este caso, supongamos que se comenzó en 25 us, tratando de llegar a 13 us, pero usted encuentra que el límite esta alrededor de 16 us; un valor menor menor y la computadora deja de responde bien. Entonces usa 16 us. Con un período de 16 us y 11 us latencia, el tiempo de salida más corto será 16-11=5 us. El dirver solo necesita 2 us, así que tiene un margen. El margen es bueno; no se quiere perder pasos por haber usado un tiempo demasiado corto.

¿Cuál es la tasa de paso máxima? Recuerde, dos períodos para dar un paso. Se estableció en 16 us para el período, por lo que un paso consume 32 us. Funcionariamos a unos no malos 31250 pasos por segundo.

7.3.1.4. Uso de steplen, stepspace, dirsetup y/o dirhold

En la última sección, obtuvimos para el driver Xylotex un período de 16 us y una velocidad máxima de 31250 pasos por segundo. Pero el Gecko estaba atascado en 31 us y unos no tan buenos 16129 pasos por segundo. El ejemplo de Xylotex es lo mejor que podemos hacer con el. Pero el Gecko se puede mejorar.

El problema con el G202 es el requisito de tiempo de espera de 20 us. Esto, y la latencia de 11 us es lo que nos obliga a usar un período lento de 31 us. Pero el El generador de pasos software LinuxCNC tiene algunos parámetros que le permiten aumentar diferentes tiempos en uno o varios períodos. Por ejemplo, si steplen es cambiado de 1 a 2, entonces habrá dos períodos entre el comienzo y final del impulso de paso. Del mismo modo, si se cambia dirhold de 1 a 3, habrá al menos tres períodos entre el pulso de paso y un cambio del pin de dirección.

Si podemos usar dirhold para cumplir con el requisito de 20 us "hold time", entonces el siguiente tiempo más largo es el de 4.5 us high time. Agregue la latencia de 11 us a 4.5 us alta hora, y obtienes un período mínimo de 15.5 us. Cuando intentas 15.5 us, usted encuentra que la computadora es lenta, por lo que se acomoda en 16 nosotros. Si dejamos dirhold en 1 (valor predeterminado), entonces el tiempo mínimo entre paso y dirección es el período de 16 us menos latencia de 11 us = 5 us, que no es suficiente Necesitamos otros 15 nosotros. Como el período es 16 us, nosotros necesita un período más. Entonces cambiamos dirhold de 1 a 2. Ahora el mínimo el tiempo desde el final del impulso de paso hasta el pin de dirección cambiante es 5 + 16 = 21 us, y no tenemos que preocuparnos de que Gecko ponga dirección incorrecta debido a la latencia.

Si la computadora tiene una latencia de 11 us, entonces una combinación de 16 us período base, y un valor dirhold de 2 asegura que siempre nos reuniremos los requisitos de tiempo del Gecko. Para caminar normal (sin dirección cambio), el valor aumentado de dirhold no tiene ningún efecto. Hacen falta dos períodos que totalizan 32 us para hacer cada paso, y tenemos el mismo 31.250 paso por segundo que obtuvimos con el Xylotex.

El número de latencia 11 us utilizado en este ejemplo es muy bueno. Si trabajas a través de estos ejemplos con mayor latencia, como 20 o 25 us, la parte superior la velocidad de paso tanto para Xylotex como para Gecko será menor. Pero el se aplican las mismas fórmulas para calcular el BASE_PERIOD óptimo, y para retocando dirhold u otros parámetros del generador de pasos.

7.3.1.5. ¡No hacer suposiciones!

Para un sistema de pasos basado en software, rápido y confiable, no puede simplemente suponer períodos y otros parámetros de configuración. Necesita mediciones en su computadora, y hacer cálculos para asegurarse de que sus drivers obtienen las señales que necesitan.

Para facilitar las matemáticas, se he creado una hoja de cálculo de Open Office http://wiki.linuxcnc.org/uploads/StepTimingCalculator.od Usted ingresa el resultado de la prueba de latencia y los requisitos de tiempo de su driver paso a paso y la hoja de cálculo calcula el BASE_PERIOD óptimo. A continuación, pruebe el período para asegurarse de que no se ralentizará o bloqueará tu computador. Finalmente, ingrese el período real y la hoja de cálculo le dirá la configuración de parámetros de stepgen que se necesitan para cumplir con su requisitos de tiempo de la unidad. También calcula la velocidad máxima de paso que podrá generar

Se han agregado mas cosas a la hoja de cálculo para calcular la velocidad máxima y cálculos eléctricos paso a paso.

7.4. Diagnósticos para steppers

:ini:' *:hal:*' :ngc:"

Muchas veces lo que se obtiene no es lo que espera; pero todo es cuestión de experiencia. Aprender de la experiencia aumenta la comprensión del todo. La mejor manera de diagnosticar problemas es dividir y conquistar. Con esto quiero decir si puede eliminar la mitad de las variables de la ecuación, cada vez encontrará el problema más rápido. En el mundo real esto no siempre es el caso, pero generalmente es una buena manera de comenzar.

7.4.1. Problemas comunes

7.4.1.1. El Stepper se mueve solo un paso

La razón más común en una nueva instalación para que un motor paso a paso haga esto es que se han intercambiado las señales de paso y dirección. Si presiona el jog hacia adelante y hacia atrás, alternativamente, y el motor se mueve un paso cada vez, y en la misma dirección, ahí está la prueba.

7.4.1.2. Los steppers no se mueven en absoluto

Muchos drivers tienen un pin de activación o necesitan una bomba de carga para activar su salida.

7.4.1.3. Distancia no correcta

Si le ordena al eje que se mueva una distancia específica y se mueve a otra distancia, entonces su ajuste de escala es incorrecto.

7.4.2. Mensajes de error

7.4.2.1. Error de seguimiento

El concepto de error de seguimiento es extraño cuando se habla de motores de pasos. Como son sistemas de lazo abierto, no hay retroalimentacion de posición para hacerle saber si realmente está fuera de rango. LinuxCNC calcula si puede mantener el cumplimiento del movimiento solicitado y, si no, entonces da un error de seguimiento. Los erroresde seguimiento generalmente son el resultado de uno de los siguientes problemas en sistemas paso a paso.

- FERROR demasiado pequeño - MIN_FERROR demasiado pequeño - MAX_VELOCITY demasiado rápida - MAX_ACCELERATIC demasiado rápida - BASE_PERIOD demasiado largo - Backlash agregado a un eje

Cualquiera de los anteriores puede hacer que los pulsos en tiempo real no pueda mantenerse conforme a la tasa de pasos solicitada. Esto puede suceder si no ejecutó la prueba de latencia el tiempo suficiente para obtener un buen número para el Asistente Stepconf, o si configura la Velocidad Máxima o la Aceleración Máxima demasiado alta.

Si agregó backlash, debe aumentar STEPGEN_MAXACCEL hasta que duplique MAX_ACCELERATION en la sección AXIS del archivo INI para cada eje al que se agregó. LinuxCNC utiliza una "aceleración adicional" en una inversión para tener en cuenta el backlash. Sin corrección de backlash, la aceleración del generador de pasos será solo un pequeño porcentaje mayor que la aceleración planificada del movimiento.

7.4.2.2. Error RTAPI

Usted puede recibir este error:

RTAPI: ERROR: retraso inesperado en tiempo real en la tarea n

Rtapi genera este error basándose en una indicación de RTAI de que se violó el tiempo limite. Suele ser una indicación de que BASE_PERIOD en la sección [EMCMOT] del archivo ini está configurado demasiado bajo. Debería correr la prueba de latencia durante un período prolongado de tiempo para ver si tiene algun retraso que causarían este problema. Si utilizó el Asistente Stepconf, ejecútelo de nuevo y pruebe el jitter del período base nuevamente, y ajuste Base Period Maximum Jitter en la página Información Básica de la máquina. Usted debe dejar la prueba en funcionamiento durante un período prolongado de tiempo para encontrar si algún hardware causa problemas intermitentes.

LinuxCNC rastrea el número de ciclos de CPU entre invocaciones de hilos en tiempo real. Si algún elemento de su hardware está causando demoras o sus hilos en tiempo real se configuran demasiado rápido, obtendrá este error.

NOTA: Este error solo se muestra una vez por sesión. Si tuviera su BASE_PERIOD demasiado bajo, podría obtener cientos de miles de mensajes de error por segundo si se mostrara más de uno.

7.4.3. Pruebas

7.4.3.1. Temporización de pasos

Si tiene un eje que termina en una ubicación incorrecta durante movimientos múltiples, es probable que no tenga los tiempos de mantenimiento de dirección o de timing de pasos correctos para sus drivers. En cada cambio de dirección puede estar perdiendo un paso o más. Si los motores se saturan, es también es posible que tenga el conjunto MAX_ACCELERATION o MAX_VELOCITY demasiado alto para ese eje.

El siguiente programa probará la configuración correcta del eje Z. Copie el programa en su directorio ~/emc2/nc_files y asígnele un nombre como TestZ.ngc o similar. Ponga a cero su máquina con Z = 0.000 en la parte superior de la mesa. Cargue y ejecute el programa. Hará 200 movimientos de ida y vuelta de 0.5 a 1". Si tiene un problema de configuración, encontrará que la posición final no terminará en las 0.500" que la pantalla muestra. Para probar otro eje, simplemente reemplace la Z con la letra de eje a probar en las líneas G0.

```
~~~~(programa de prueba para ver si el eje Z pierde posición)
~~~~(msg, prueba 1 de configuración del eje Z)
~~~~G20 #1000=100 (bucle 100 veces)
~~~~(este bucle tiene retrasos después de los movimientos)
~~~~(prueba ajustes de velocidad y aceleracion)
    o100 while [#1000]
    G0 Z1.000
    G4 P0.250
    G0 Z0.500
    G4 P0.250
    #1000 = [#1000 - 1]
    o100 endwhile
~~~~(msg, prueba 2 de la configuración del eje Z, S para continuar)
~~~~M1 (para aquí)
~~~~#1000=100 (bucle 100 veces)
~~~~(el siguiente ciclo no tiene demoras después de los movimientos)
~~~~(prueba los tiempos de retención de dirección del controlador y también la aceleración ↩
    máxima)
    o101 while [#1000]
    G0 Z1.000
    G0 Z0.500
    #1000 = [#1000 - 1]
    o101 endwhile
~~~~(msg, Listo ... Z debe estar exactamente .5" sobre la mesa)
~~~~M2
```

Capítulo 8

Configuracion

8.1. Configuración INI

8.1.1. Los Componentes del Archivo INI

Un archivo INI típico sigue un diseño bastante simple, que incluye;

- comentarios
- secciones
- variables

Cada uno de estos elementos está separado en líneas simples. Cada final de línea o el carácter de nueva línea crea un nuevo elemento.

8.1.1.1. Comentarios

Una línea de comentario se inicia con un ";" o una marca "#". Cuando el lector ini ve cualquiera de estas marcas al comienzo de una línea, el resto de la línea es ignorado por el software. Los comentarios se pueden usar para describir qué hará un elemento INI.

```
; Este es mi archivo de configuración de fresadora.
# Configurado el 12 de enero de 2012
```

Los comentarios también se pueden usar para *desactivar* una variable. Esto hace más fácil elegir entre diferentes variables.

```
DISPLAY = axis
# DISPLAY = touchy
```

En esta lista, la variable DISPLAY se establecerá en Axis porque el otro valor está comentado. Si alguien edita descuidadamente una lista como esta y deja dos de las líneas sin comentar, se utilizará la primera encontrada.

Tenga en cuenta que dentro de una variable, los caracteres "#" y ";" no denotan comentarios:

```
INCORRECT = valor       # y un comentario

# Comentario correcto
CORRECT = value
```

8.1.1.2. Secciones

Las partes relacionadas de un archivo ini se separan en secciones. El nombre de una sección se encierra entre paréntesis como este *[THIS_SECTION]* El orden de las secciones no es importante. Las secciones comienzan en el nombre de sección y finalizan en el siguiente nombre de sección.

Las siguientes secciones son utilizadas por LinuxCNC:

- *[EMC]* información general
- *[<< sec:display-section,DISPLAY>>]* configuraciones relacionadas con la interfaz gráfica de usuario
- *[<< sec:sección de filtro,FILTER>>]* configuración de los programas de filtro de entrada
- *[<< sec:rs274ngc-section,RS274NGC>>]* configuración utilizada por el intérprete de código g
- *[<< sec:emcmot-section,EMCMOT>>]* configuraciones utilizadas por el controlador de movimiento en tiempo real
- *[<< sec:task-section,TASK>>]* configuraciones utilizadas por el controlador de tareas
- *[<< sec:hal-section,HAL>>]* especificar archivos .hal
- *[<< sec:halui-section,HALUI >>]* comandos MDI utilizados por HALUI
- *[<< sec:applications-section,APPLICATIONS>>]* otras aplicaciones que iniciará LinuxCNC
- *[<< sec:traj-section,TRAJ>>]* configuraciones adicionales utilizadas por el controlador de movimiento en tiempo real
- *[<< sec:joint-section,JOINT_n>>]* variables de articulaciones individuales
- *[<< sec:axis-section,AXIS_n>>]* variables de eje individuales
- *[<< sec:kins-section,KINS>>]* variables cinemáticas
- *[<< sec:emcio-section,EMCIO>>]* configuración utilizada por el controlador de E/S

8.1.1.3. Variables

Una línea variable se compone de un nombre variable, un signo igual (=) y un valor. Todo, desde el primer carácter de espacio no blanco después del =, hasta el final de la línea se pasa como valor, por lo que puede incrustar espacios en símbolos de cadena si los desea o los necesita. Un nombre de variable a menudo es llamado palabra clave o keyword

Ejemplo de Variable

```
MACHINE = Mi máquina
```

Una línea variable puede extenderse a varias líneas con una barra diagonal inversa (\). Se permite un máximo de MAX_EXTEND_LINES (== 20). No debe haber espacios en blanco que sigan al carácter de barra invertida final.

Los identificadores de sección no pueden extenderse a varias líneas.

Variable con línea extendida. Ejemplo

```
APP = sim_pin \
ini.0.max_acceleration \
ini.1.max_acceleration \
ini.2.max_acceleration \
ini.0.max_velocity \
ini.1.max_velocity \
ini.2.max_velocity
```

Las siguientes secciones detallan cada sección del archivo de configuración, utilizando valores de muestra para las líneas de configuración.

Las variables que utiliza LinuxCNC siempre deben usar nombres de sección y nombres de variables como se muestra. En el siguiente ejemplo, a la variable *MACHINE* se le asigna el valor *Mi máquina*.

8.1.1.4. Secciones y variables personalizadas

La mayoría de las configuraciones de muestra utilizan secciones y variables personalizadas para poner todas las configuraciones en una ubicación para mayor comodidad.

Para agregar una variable personalizada a una sección LinuxCNC existente, simplemente incluya la variable en esa sección.

Ejemplo de variable personalizada

```
[JOINT_0]
TYPE = LINEAR
...
SCALE = 16000
```

Para introducir una sección personalizada con sus propias variables, agregue la sección y variables al archivo INI.

Ejemplo de sección personalizada

```
[PROBE]
Z_FEEDRATE = 50
Z_OFFSET = 12
Z_SAFE_DISTANCE = -10
```

Para usar las variables personalizadas en su archivo HAL, coloque la sección y el nombre de la variable en lugar del valor.

Ejemplo HAL

```
setp offset.1.offset [PROBE]Z_OFFSET
setp stepgen.0.position-scale [JOINT_0]SCALE
```

nota
El valor almacenado en la variable debe coincidir con el tipo especificado por el pin del componente

Para usar las variables personalizadas en el código G, use la sintaxis de variable global `#<_ini[section]variable>`. El siguiente ejemplo muestra una simple rutina touch-off del eje Z para una fresadora o fresadora que utiliza una placa de sonda.

Ejemplo de código G

```
G91
G38.2 Z#<_ini[probe]z_safe_distance> F#<_ini[probe]z_feedrate>
G90
G1 Z#5063
G10 L20 P0 Z#<_ini[probe]z_offset>
```

8.1.1.5. Archivos include

Un archivo INI puede incluir el contenido de otro archivo usando una directiva #INCLUDE.

Formato #INCLUDE

```
#INCLUDE filename
```

El nombre del archivo se puede especificar como:

* un archivo en el mismo directorio que el archivo INI * un archivo relativo al directorio de trabajo * un nombre de archivo absoluto (comienza con un /) * un nombre de archivo relativo al directorio de usuario (comienza con un ~)

Se admiten varias directivas #INCLUDE.

Ejemplos #INCLUDE

```
#INCLUDE joint_0.inc
#INCLUDE ../parallel/joint_1.inc
#INCLUDE below/joint_2.inc
#INCLUDE /home/myusername/myincludes/display.inc
#INCLUDE ~/linuxcnc/myincludes/rs274ngc.inc
```

Las directivas #INCLUDE son compatibles solo con un nivel de expansión; un archivo incluide no puede incluir archivos include adicionales. La extensión de archivo recomendada es .inc. No utilice una extensión de archivo .ini para los archivos incluidos.

8.1.2. Secciones del Archivo INI

8.1.2.1. Sección [EMC]

- *VERSION = 1.1* - el número de versión para la configuración. Cualquier otro valor distinto de 1.1 hará que se ejecute el verificador de configuración e intente actualizar la configuración al nuevo estilo de configuración ejes/articulaciones.
- *MACHINE = Mi máquina* - este es el nombre de la configuracion del controlador de la maquina, que se imprime en la parte superior de la mayoría de las interfaces gráficas. Puede poner lo que sea aquí, siempre que lo haga en una sola línea.
- *DEBUG = 0* - nivel de depuración; 0 significa que no se imprimirán mensajes cuando LinuxCNC esté ejecutandose desde un terminal. Las marcas de depuración generalmente solo son útiles para desarrolladores. Vea src/emc/nml_intf/debugflags.h para otras configuraciones.

8.1.2.2. Sección [DISPLAY]

Los diferentes programas de interfaz de usuario utilizan diferentes opciones, y no todas las opciónes son compatibles con todas las interfaces de usuario. Hay varias interfaces como Axis, Gmoccapy, Touchy, qtvcp's QtDragon y Gscreen. Axis es una interfaz para usar con computadora y monitor normal. Touchy es para usar con pantallas táctiles. Gmoccapy se puede usar en ambos tipos y también ofrece muchas conexiones para controles hardware. Las descripciones de las interfaces se encuentran en la sección Interfaces del Manual de usuario.

- *DISPLAY = xxx* - El nombre de la interfaz de usuario a usar. Opciones válidas son: *axis*, *touchy*, *gmoccapy*, *gscreen*, *tklinuxcnc*, *qtvcp*
- *POSITION_OFFSET = XXX* - el sistema de coordenadas (RELATIVE o MACHINE) a mostrar en el DRO cuando se inicia la interfaz de usuario. El sistema de coordenada RELATIVA refleja los offsets de coordenadas G92 y G5x vigentes en cada momento.
- *POSITION_FEEDBACK = XXX* - el valor de coordenadas (COMMANDED o ACTUAL) a mostrar en el DRO cuando se inicia la interfaz de usuario. En Axis esto se puede cambiar desde el menú "Ver". La posición COMMANDED es la posición solicitada por LinuxCNC. La posición ACTUAL es la posición retroalimentada de los motores si se tiene retroalimentacion como en la mayoría de los servosistemas. Por lo general, se utiliza el valor COMMANDED.
- *DRO_FORMAT_MM = %+08.6f* - ajusta el formato DRO predeterminado en modo métrico. (normalmente 3 lugares decimales y 6 dígitos, rellenado con espacios, a la izquierda). El ejemplo anterior rellenará con ceros, mostrará 6 dígitos decimales y fuerza visualización de un signo + para números positivos. El formateo sigue la práctica de Python. https://docs.python.org/-2/library/string.html#format-specification-mini-language y se generará un error si el formato no puede aceptar un valor de punto flotante.
- *DRO_FORMAT_IN = % 4.1f* - ajusta el formato DRO predeterminado en modo imperial. (normalmente 4 lugares decimales, rellenados con espacios de 6 dígitos a la izquierda) El ejemplo anterior mostrará solo un dígito decimal. El formato sigue la práctica de Python. https://docs.python.org/2/library/string.html#format-specification-mini-language . Se generará un error si el formato no puede aceptar un valor de punto flotante.
- *CONE_BASESIZE = .25* - ajusta el tamaño predeterminado (.5) de la base del cono/herramienta en la pantalla de gráficos

- *MAX_FEED_OVERRIDE = 1.2* - el máximo ajuste de alimentación que el usuario puede seleccionar. 1.2 significa 120 % de la velocidad de alimentación programada.
- *MIN_SPINDLE_OVERRIDE = 0.5* - El mínimo ajuste del husillo que el usuario puede seleccionar. 0.5 significa el 50 % de la velocidad programada del husillo. (Esto se usa para establecer la velocidad mínima del husillo).
- *MIN_SPINDLE_N_OVERRIDE = 0.5*- El ajuste mínimo del husillo N que el usuario puede seleccionar. 0.5 significa el 50 % de la velocidad programada del husillo. (Esto se usa para establecer la velocidad mínima del husillo). En una máquina de múltiples husillos habrá entradas para cada número de husillo. Solo Qtvcp
- *MAX_SPINDLE_OVERRIDE = 1.0* - El ajuste máximo del husillo que el usuario puede seleccionar. 1.0 significa el 100 % de la velocidad programada del husillo.
- *MAX_SPINDLE_N_OVERRIDE = 1.0* - El ajuste máximo que el usuario puede seleccionar. 1.2 significa 120 % de la velocidad de alimentación programada. En una máquina de múltiples husillos habrá entradas para cada número de husillo. Solo Qtvcp
- *DEFAULT_SPINDLE_SPEED = 100* - Las RPM predeterminadas del husillo cuando se inicia en modo manual. Si esta configuración no está presente, el valor predeterminado es 1 RPM para AXIS y 300 RPM para gmoccapy.
- *DEFAULT_SPINDLE_N_SPEED = 100* - Las RPM predeterminadas del husillo en modo manual. En una máquina de múltiples husillos habrá entradas para cada número de husillo. Solo Qtvcp
- *SPINDLE_INCREMENT = 200* - Incremento utilizado al hacer clic en los botones de aumento/disminución. Qtvcp solamente
- *MIN_SPINDLE_N_SPEED = 1000* - las RPM mínimas que se pueden seleccionar manualmente. En una máquina de múltiples husillos habrá entradas para cada número de husillo. Solo Qtvcp
- *MAX_SPINDLE_0_SPEED = 20000* - las RPM máximas que se pueden seleccionar manualmente. En una máquina de múltiples husillos habrá entradas para cada número de husillo. Solo Qtvcp
- *PROGRAM_PREFIX = ~/linuxcnc/nc_files* - La ubicación predeterminada para archivos de código g y ubicación de códigos M definidos por el usuario. Esta ubicación es buscada para el nombre del archivo antes de la ruta de subrutina y la ruta M de usuario si se especifica en la sección [RS274NGC].
- *INTRO_GRAPHIC = emc2.gif* - la imagen que se muestra en la pantalla de inicio.
- *INTRO_TIME = 5* - el tiempo máximo durante el que mostrar la pantalla de inicio, en segundos.
- *CYCLE_TIME = 0.05* - Tiempo de ciclo en segundos que la pantalla se mantiene entre refrescos.

nota
GladeVCP utiliza los siguientes elementos [DISPLAY], consulte la sección incrustando una pestaña del Capítulo GladeVCP.

- *EMBED_TAB_NAME=Demo GladeVCP*
-

nota
Los diferentes programas de interfaz de usuario utilizan diferentes opciones, y no todas las opciones son compatibles con todas las interfaces de usuario. Consulte el documento GUI AXIS para obtener detalles sobre AXIS. Consulte el documento gmoccapy para obtener detalles sobre Gmoccapy.

- *DEFAULT_LINEAR_VELOCITY = .25* - La velocidad predeterminada para los movimientos lineales, en unidades máquina por segundo.
- *MIN_VELOCITY = .01* - el valor más bajo aproximado del control deslizante de jog.
- *MAX_LINEAR_VELOCITY = 1.0* - La velocidad máxima para jog lineal, en unidades de máquina por segundo.

- *MIN_LINEAR_VELOCITY = .01* - el valor más bajo aproximado del control deslizante de jog lineal.
- *DEFAULT_ANGULAR_VELOCITY = .25* - La velocidad predeterminada para jog angular, en unidades máquina por segundo.
- *MIN_ANGULAR_VELOCITY = .01* - el valor más bajo aproximado del control deslizante de jog angular.
- *MAX_ANGULAR_VELOCITY = 1.0* - La velocidad máxima para jog angular, en unidades de máquina por segundo.
- *INCREMENTS = 1 mm, .5 in, ...* - Define los incrementos disponibles para jogs incrementales. Los INCREMENTS se pueden usar para ajustar los valores predeterminados. Los valores pueden ser números decimales (por ejemplo, 0.1000) o números fraccionarios (por ejemplo, 1/16), opcionalmente seguido por una unidad (cm, mm, um, inch (pulgadas), in (pulgadas) o mil (milésimas de pulgada)). Si no se especifica una unidad, se supone la unidad de máquina. Las distancias métricas e imperiales se pueden mezclar: INCREMENTS = 1 inch, 1 mil, 1 cm, 1 mm, 1 um es una entrada válida.
- *GRIDS = 10 mm, 1 in, ...* - Define los valores preestablecidos para las líneas de cuadrícula. El valor se interpreta de la misma manera que *INCREMENTS*.
- *OPEN_FILE = /path/absoluto/a/file.ngc* - el archivo que se mostrará en la gráfica de vista previa cuando se inicie AXIS. Una cadena en blanco "" no cargará ningún archivo al inicio. gmoccapy no usará esta configuración, ya que ofrece una entrada correspondiente en su página de configuración.
- *EDITOR = gedit* - el editor que se usará al seleccionar Archivo> Editar para editar código G desde el menú de AXIS. Esto debe configurarse para que este elemento de menú trabaje. Otra entrada válida es "gnome-terminal -e vim". Esta entrada no se aplica a gmoccapy, ya que gmoccapy tiene un editor integrado.
- *TOOL_EDITOR = tooledit* - el editor que se utilizará al editar la tabla de herramientas (por ejemplo, al seleccionar "Archivo> Editar tabla de herramientas ..." en Axis). Otras entradas validas son "gedit", "gnome-terminal -e vim" y "gvim". Esta entrada no se aplica a gmoccapy, ya que gmoccapy tiene un editor integrado.
- *PYVCP = /filename.xml* - el archivo de descripción del panel PyVCP. Ver el Capítulo PyVCP para más información.
- *PYVCP_POSITION = BOTTOM* - la ubicación del panel PyVCP en la interfaz de usuario AXIS. Si se omite esta variable, el panel pasará por defecto al lado derecho. La unica alternativa valida es BOTTOM. Vea el Capítulo PyVCP para más información.
- *LATHE = 1* - cualquier valor no vacío (incluido "0") hace que Axis utilice el "modo torno" con una vista superior y con Radio y Diámetro en el DRO.
- *BACK_TOOL_LATHE = 1* - cualquier valor no vacío (incluido "0") hace que Axis utilice el "modo torno de herramienta trasera" con el eje X invertido.
- *FOAM = 1* - cualquier valor no vacío (incluido "0") hace que Axis cambie la visualización para el modo cortador de espuma.
- *GEOMETRY = XYZABCUVW* - controla la vista previa y el backplot de movimiento giratorio. Este item consiste en una secuencia de letras de eje, opcionalmente precedidas por un signo "-". Esta secuencia especifica el orden en que se aplica el efecto de cada eje, con un "-" que invierte el sentido de la rotación. La cadena de GEOMETRY adecuada depende de la configuración de la máquina y de la cinemática usada para controlarla. La cadena de ejemplo GEOMETRY = XYZBCUVW es para una máquina de 5 ejes donde la cinemática hace que UVW se muevan en el sistema de coordenadas de la herramienta y XYZ en el sistema de coordenadas del material. El orden de las letras es importante, porque expresa el orden en que se aplican las diferentes transformaciones. Por ejemplo, girar alrededor de C y luego de B es diferente que girar alrededor B y despues de C. GEOMETRY no tiene efecto sin un eje rotativo. Las máquinas de corte de espuma (FOAM = 1) deben especificar "XY;UV" o dejar el valor en blanco aunque este valor se ignore actualmente en el modo de cortador de espuma. UNA versión futura puede definir qué significa ";", pero si lo hace "XY;UV" significará lo mismo que el cortador de espuma actual por defecto.
- *ARCDIVISION = 64* - Establece la calidad de la vista previa de los arcos. Los arcos se previsualizan dividiendolos en una serie de líneas rectas; un semicírculo se divide en **ARCDIVISIÓN** partes. Los valores más grandes dan una vista previa más precisa, pero tardan más tiempo en cargar y dan como resultado una pantalla más lenta. Los valores más pequeños dan una vista previa menos precisa, pero tarda menos tiempo en cargar y puede resultar en una velocidad más rápida del monitor. El valor predeterminado de 64 significa que un círculo de hasta 3 pulgadas se mostrará con precision de 1 mil (.03 %).
- *MDI_HISTORY_FILE =* - El nombre del archivo de historial MDI local. Si no se especifica, Axis guardará el historial MDI en **.axis_mdi_history** en el directorio de usuario. Esto es útil si tiene múltiples configuraciones en una computadora.

- *JOG_AXES* = - el orden en que se asignan las teclas de desplazamiento a las letras del eje. Las flechas izquierda y derecha se asignan a la letra del primer eje, arriba y abajo a la segunda, página arriba/página abajo a la tercera, y corchetes izquierdo y derecho a la cuarta. Si no se especifica, el valor predeterminado se determina a partir de los valores de [TRAJ]COORDINATES, [DISPLAY]LATHE y [DISPLAY]FOAM.
- *JOG_INVERT* = - para cada letra de eje, se invierte la dirección de jog. El valor predeterminado es "X" para tornos y en blanco en el resto.

nota
La configuración de *JOG_AXES* y *JOG_INVERT* se aplican al modo de jog universal por letra de eje de coordenadas y están vigentes mientras se encuentra en modo universal después de un recorrido homing exitoso. Cuando se opera en modo articulacion anterior al homing, las teclas de desplazamiento del teclado se asignan en una secuencia fija: izquierda/derecha: joint0, arriba/abajo: joint1, pg arriba/ pg abajo: joint2, corchete izquierdo/derecho: joint3

- *USER_COMMAND_FILE* = *mycommands.py* - El nombre de un archivo Python opcional, específico de configuración originado por la GUI Axis en lugar del archivo específico del usuario `~/.axisrc`.

nota
El siguiente elemento [DISPLAY] es utilizado únicamente por la interfaz TKLinuxCNC.

- *HELP_FILE* = *tklinucnc.txt* - Ruta al archivo de ayuda.

8.1.2.3. Sección [FILTER]

AXIS y gmoccapy tienen la capacidad de enviar archivos cargados a través de un programa de filtro. Este filtro puede hacer cualquier tarea deseada; algo tan simple como asegurarse el archivo termina con M2, o algo tan complicado como detectar si la entrada es una imagen de profundidad y generar código g para fresar la forma definida. La sección [FILTER] del archivo ini controla cómo trabajan los filtros. Primero, para cada tipo de archivo, escriba una línea PROGRAM_EXTENSION. Luego, especifique el programa a ejecutar para cada tipo de archivo. Este programa recibe el nombre del archivo de entrada como primer argumento, y debe escribir el código RS274NGC en la salida estándar. Esta salida es lo que se mostrará en el área de texto, se previsualizará en el área de pantalla y sera ejecutado por LinuxCNC cuando se ordene "Ejecutar".

- *PROGRAM_EXTENSION* = *.extension Descripción*

Si su postprocesador genera archivos en mayúsculas, es posible que desee agregar la siguiente linea:

- *PROGRAM_EXTENSION* = *.NGC XYZ Post Processor*

Las siguientes líneas agregan soporte para el convertidor de imagen a código G incluido con LinuxCNC.

- *PROGRAM_EXTENSION* = *.png, .gif, .jpg Imagen de profundidad de escala de grises*
  - *png* = *imagen-to-gcode*
  - *gif* = *imagen-to-gcode*
  - *jpg* = *imagen-to-gcode*

Un ejemplo de un convertidor de código G personalizado ubicado en el directorio linuxcnc.

- 'PROGRAM_EXTENSION = .gcode Impresora 3D
  - *gcode* = */home/mill/linuxcnc/convert.py*

NOTA: El archivo de programa asociado con una extensión debe tener la ruta al programa completa o estar ubicado en un directorio que se encuentra en la ruta del sistema.

También es posible especificar un intérprete:

- *PROGRAM_EXTENSION = .py Python Script*
  - *py = python*

De esta manera, cualquier script de Python se puede abrir y su salida es tratada como código g. Un script de ejemplo de este tipo está disponible en nc_files/holecircle.py. Este script crea código g para perforar una serie de agujeros a lo largo de una circunferencia. Muchos más generadores de códigos g están en el sitio Wiki LinuxCNC http://wiki.linuxcnc.org/.

Si se establece la variable de entorno AXIS_PROGRESS_BAR, entonces las líneas escritas a stderr de la forma

- *FILTER_PROGRESS= %d*

establece la barra de progreso de Axis en el porcentaje dado. Esta característica debe ser utilizado por cualquier filtro que se ejecute durante mucho tiempo.

Los filtros de Python deben usar la función de impresión para enviar el resultado a Axis.

Este programa de ejemplo filtra un archivo y agrega un eje W para que coincida con el eje Z. Depende de que haya un espacio entre cada palabra de eje para trabajar.

```
#!/usr/bin/env python

import sys

def main(argv):

  openfile = open(argv[0], 'r')
  file_in = openfile.readlines()
  openfile.close()

  file_out = []
  for line in file_in:
    # print line
    if line.find('Z') != -1:
      words = line.rstrip('\n')
      words = words.split(' ')
      newword = ''
      for i in words:
        if i[0] == 'Z':
          newword = 'W'+ i[1:]
      if len(newword) > 0:
        words.append(newword)
        newline = ' '.join(words)
        file_out.append(newline)
    else:
      file_out.append(line)
  for item in file_out:
    print "%s" % item

if __name__ == "__main__":
main(sys.argv[1:])
```

8.1.2.4. Sección [RS274NGC]

- *PARAMETER_FILE = myfile.var* - El archivo ubicado en el mismo directorio que el archivo ini que contiene los parámetros utilizados por el intérprete (guardado entre ejecuciones).

- *ORIENT_OFFSET = 0* - Un valor float agregado al parámetro R de una operación M19 Orientar Husillo. Se usa para definir una posición cero arbitraria independientemente de la orientación de montaje del codificador.
- *RS274NGC_STARTUP_CODE = G17 G20 G40 G49 G64 P0.001 G80 G90 G92 G94 G97 G98* - Una cadena de códigos NC que inicializa el intérprete. Esto no es un sustituto para especificar códigos g modales en la parte superior de cada archivo ngc, porque los códigos modales de las máquinas difieren, y pueden ser cambiadas por el código g interpretado anteriormente en la sesión.
- *SUBROUTINE_PATH = ncsubroutines:/tmp/testsubs:lathesubs:millsubs* - Especifica una lista separada por dos puntos (:) de hasta 10 directorios a buscar cuando se especifican subrutinas de un solo archivo en gcode. Estos directorios se buscan después de buscar [DISPLAY] PROGRAM_PREFIX (si está especificado) y antes de buscar [WIZARD] WIZARD_ROOT (si se especifica). Las rutas se buscan en el orden que están listados El primer archivo de subrutina coincidente encontrado en la búsqueda se utiliza. Los directorios se especifican en relación con el directorio actual para el archivo ini o como rutas absolutas. La lista debe no contienen espacios en blanco intermedios.
- *CENTER_ARC_RADIUS_TOLERANCE_INCH = n* Predeterminado 0.00005
- *CENTER_ARC_RADIUS_TOLERANCE_MM = n* Predeterminado 0.00127
- *USER_M_PATH = myfuncs:/tmp/mcodes:experimentalmcodes* - Especifica una lista de directorios separados por dos puntos (:) para funciones definidas por el usuario. Los directorios se especifican relativas al directorio actual del archivo ini o como rutas absolutas. La lista no debe contener ningun espacio en blanco.

  Se realiza una búsqueda para cada posible función definida por el usuario, típicamente (M100-M199). El orden de búsqueda es:

  1. [DISPLAY]PROGRAM_PREFIX (si se especifica)
  2. Si no se especifica [DISPLAY]PROGRAM_PREFIX, busca en la ubicación predeterminada: nc_files
  3. Luego busca en cada directorio de la lista [RS274NGC]USER_M_PATH
     El primer ejecutable M1xx encontrado en la búsqueda se usa para cada M1xx.

---

nota
El número máximo de directorios USER_M_PATH se define en tiempo de compilación (predeterminado: *USER_DEFINED_FUNCTION_MAX_DIRS == 5*).

- *INI_VARS = 1* Predeterminado 1 Permite que los programas de código G lean valores del archivo INI usando el formato #<_ini[sección]nombre>. Ver parámetros del código G
- *HAL_PIN_VARS = 1* Predeterminado 1 Permite que los programas de código G lean los valores de los pines HAL usando el formato #<_hal[Elemento Hal]> El acceso a esta variable es de solo lectura. Consulte parámetros de código G para obtener más detalles y una advertencia importante.
- *RETAIN_G43 = 0* Predeterminado 0 Cuando está configurado, puede activar G43 después de cargar la primera herramienta, y luego despreocuparse por eso a través del programa. Cuando usted finalmente descargue la última herramienta, el modo G43 se cancela.
- *OWORD_NARGS = 0* Predeterminado 0 Si esta función está habilitada, una subrutina llamada puede determinar el número de parámetros posicionales reales pasados al inspeccionar el parámetro `#<n_args>`.
- *NO_DOWNCASE_OWORD = 0* Predeterminado 0 Conservar mayúsculas y minúsculas en los nombres O-word dentro de los comentarios si está configurado, permite leer elementos HAL de mayúsculas y minúsculas en comentarios estructurados como *(debug, #<_hal[MixedCaseItem])*..
- *OWORD_WARNONLY = 0* Predeterminado 0 Advertir en lugar de error en caso de errores en las subrutinas O-word.

[NOTE] Las seis opciones anteriores fueron controladas por la máscara de bits *FEATURES* en versiones de LinuxCNC anteriores a 2.8. Esta etiqueta INI ya no trabaja.

nota
[WIZARD]WIZARD_ROOT es una ruta de búsqueda válida pero el asistente no se ha implementado por completo y los resultados de su uso son impredecibles.

- *REMAP=M400 modalgroup=10 argspec=Pq ngc=myprocedure* Vea el capítulo Remap Extender G-Code para más detalles.
- *ON_ABORT_COMMAND = O <on_abort> call* Vea el capítulo Remap- Extender G-Code para más detalles.

8.1.2.5. Sección [EMCMOT]

Esta sección es una sección personalizada y LinuxCNC no la utiliza directamente. Muchas configuraciones utilizan valores de esta sección para cargar el controlador de movimiento. Para obtener más información sobre el controlador de movimiento, consulte la sección Motion.

- *EMCMOT = motmod* - el nombre del controlador de movimiento generalmente se usa aquí.
- *BASE_PERIOD = 50000* - el período de la tarea *Base* en nanosegundos.
- *SERVO_PERIOD = 1000000* - Este es el período de tarea "Servo" en nanosegundos.
- *TRAJ_PERIOD = 100000* - este es el período de la tarea *Planificador de trayectoria* en nanosegundos
- *COMM_TIMEOUT = 1.0* - Número de segundos para esperar a Motion (la parte en tiempo real del controlador de movimiento) para acusar recibo de mensajes desde Task (la parte no en tiempo real del controlador de movimiento).

8.1.2.6. Sección [TASK]

- *TASK = milltask* - Especifica el nombre del ejecutable *task*. El ejecutable *task* hace varias cosas, como comunicarse con las interfaces de usuario a través de NML, comunicarse con el planificador de movimiento en tiempo real sobre memoria compartida no HAL e interpretar gcode. Actualmente solo hay una tarea ejecutable que tiene sentido para el 99.9 % de usuarios, milltask.
- *CYCLE_TIME = 0.010* - El período, en segundos, en el que se ejecutará TASK. Este parámetro afecta el intervalo de sondeo cuando se espera que se complete el movimiento, cuando se ejecuta una instrucción de pausa y al aceptar un comando desde la interfaz de usuario. Por lo general, no es necesario cambiar este número.

8.1.2.7. Sección [HAL]

- *HALFILE = example.hal* - ejecuta el archivo *example.hal* al inicio. Si se especifica *HALFILE* varias veces, los archivos se ejecutan en el orden en que aparecer en el archivo ini. Casi todas las configuraciones tendrán al menos un *HALFILE*, y los sistemas paso a paso suelen tener dos de estos archivos, uno que especifica la configuración paso a paso genérica (*core_stepper.hal*) y uno que especifica los pines de la máquina (*xxx_pinout.hal*). HALFILES se encuentran mediante una búsqueda. Si el archivo nombrado se encuentra en el directorio que contiene el archivo ini, se utiliza. Si el archivo nombrado no se encuentra en este directorio de archivos ini, se realiza una búsqueda utilizando la biblioteca de sistema de halfiles. Un HALFILE también se puede especificar como una ruta absoluta (cuando el nombre comienza con "/"). No se recomiendan rutas absolutas ya que su uso puede limitar la reubicación de configuraciones.
- *HALFILE = texample.tcl [arg1 [arg2] . . .]]* - Ejecuta el archivo tcl *texample.tcl* al inicio con arg1, arg2, etc. como ::argv list. Los archivos con un sufijo .tcl son procesados como se indica arriba, pero usan haltcl para procesado. Vea el capitulo HALTCL para más información.
- *HALFILE = LIB:sys_example.hal* - Ejecuta el archivo de la biblioteca de sistema *sys_example.hal* al inicio. El uso explícito del prefijo LIB: provoca el uso de la biblioteca del sistema HALFILE sin buscar en el directorio de archivos ini.

- *HALFILE = LIB:sys_texample.tcl [arg1 [arg2 . . .]]* - Ejecuta la biblioteca del sistema archivo *sys_texample.tcl* al inicio. El uso explícito de LIB: el prefijo provoca el uso de la biblioteca del sistema HALFILE sin buscando en el directorio de archivos ini.

  Los elementos HALFILE especifican archivos que cargan componentes Hal y generan conexiones de señales entre pines de componentes. Los errores comunes son 1) omisión de la declaración addf necesaria para agregar las funciones de un componente a un hilo, 2) especificadores de señal (net) incompletos. La omisión de las declaraciones addf requeridas es casi siempre es un error. Las señales generalmente incluyen una o más conexiones de entrada y una sola salida (pero ambas no son estrictamente necesarias). Se proporciona un archivo de biblioteca de sistema para verificar estas condiciones y informar a stdout y en una ventana emergente gui:

```
~~~~HALFILE = LIB:halcheck.tcl [nopopup]
```

nota
La línea LIB:halcheck.tcl debería ser el último [HAL]HALFILE. Especifique la opción *nopopup* para suprimir el mensaje emergente y permitir el inicio inmediato. Las conexiones realizadas con un POSTGUI_HALFILE no serán chequeadas.

- *TWOPASS = ON*- utilice el procesamiento de dos pasos para cargar componentes HAL. Con el procesamiento TWOPASS, las líneas [HAL]HALFILE= se procesan en dos pasadas. En el primer pase (pass0), se leen todos los HALFILES y se acumulan múltiples aspectos de los comandos loadrt y loadusr. Estos comandos de carga acumulada se ejecutan al final de pass0. Esta acumulación permite líneas de carga que se especificarán más de una vez para un componente dado (siempre que los nombres names= utilizados sean únicos en cada uso). En el segundo pase (pase1), los HALFILES son releídos y todos los comandos excepto los comandos de carga ejecutados previamente son ejecutados

- *TWOPASS = nodelete verbose* - la función TWOPASS se puede activar con cualquier cadena no nula que incluya las palabras clave verbose y nodelete. la palabra clave verbose provoca la impresión de detalles en la salida estandar. La palabra clave nodelete conserva archivos temporales en /tmp.

  Para obtener más información, consulte el capítulo Hal TWOPASS.

- *HALCMD = command* - Ejecuta *command* como un solo comando HAL. Si se especifica *HALCMD* varias veces, los comandos se ejecutan en el orden en que aparecen en el archivo ini. Las líneas *HALCMD* se ejecutan después de todas las líneas *HALFILE*.

- *SHUTDOWN = shutdown.hal* - Ejecuta el archivo *shutdown.hal* cuando se sale LinuxCNC. Dependiendo de los controladores de hardware utilizados, esto puede permitir configurar salidas a valores definidos cuando LinuxCNC sale normalmente. Sin embargo, ya que no se garantiza que este archivo se ejecutará (por ejemplo, en el caso de un bloqueo de la computadora) no es un reemplazo para una cadena de parada física adecuada u otras protecciones contra fallos de software.

- *POSTGUI_HALFILE = example2.hal* - Ejecuta *example2.hal* después de que la GUI haya creado sus pines HAL. Algunas GUI crean pines hal y admiten el uso de un halfile postgui para usarlos. Las GUI que admiten halfiles postgui incluyen Touchy, Axis, Gscreen y gmoccapy.

Vea la sección pyVCP con Axis para más información.

- *HALUI = halui* - agrega los pines de la interfaz de usuario de HAL. Para más información, ver el capítulo Interfaz de usuario HAL.

8.1.2.8. Sección [HALUI]

- *MDI_COMMAND = G53 G0 X0 Y0 Z0* - Se puede ejecutar un comando MDI utilizando halui.mdi-command-00. Incremente el número para cada comando que se enumera en la sección [HALUI].

8.1.2.9. Sección [APPLICATIONS]

LinuxCNC puede iniciar otras aplicaciones antes de que se inicie la interfaz gráfica de usuario especificada. Las aplicaciones se pueden iniciar después de un retraso especificado para permitir acciones dependientes de la GUI (como crear pines hal específicos de gui).

- *DELAY = valor* - segundos de espera antes de comenzar otras aplicaciones. Puede ser necesario un retraso si una aplicación tiene dependencias en acciones [HAL]POSTGUI_HALFILE o pines Hal creados por gui. (retardo predeterminado = 0).
- *APP = appname [arg1 [arg2 . . .]]* - Aplicación que se iniciará. Esta especificación se puede incluir varias veces. El nombre de la aplicación puede ser dado explícitamente como un nombre de archivo especificado absoluto o tilde (primer carácter es / o ~), un nombre de archivo relativo (los primeros caracteres del nombre de archivo son ./), o como un archivo en el directorio inifile. Si no se encuentra ningún archivo ejecutable usando estos nombres, se utiliza la ruta de búsqueda del usuario para encontrar la aplicación.

Ejemplos:

- Simular las entradas a los pines hal para la prueba (usando sim_pin, una interfaz simple gráfica de usuario para configurar las entradas a los parámetros, pines no conectados o señales sin escritores):

APP = sim_pin motion.probe-input halui.abort motion.analog-in-00

- Invocar halshow con una lista de observación previamente guardada. Como linuxcnc establece el directorio de trabajo en el directorio para el archivo inifile, puede hacer referencia a los archivos en ese directorio (ejemplo: my.halshow):

```
APP = halshow my.halshow
```

- Alternativamente, se podría especificar un archivo de lista de observación identificado con un nombre de ruta completo:

```
APP = halshow ~/saved_shows/spindle.halshow
```

- Abrir halscope usando una configuración previamente guardada:

```
APP = halscope -i my.halscope
```

8.1.2.10. [TRAJ] Sección

aviso
El nuevo Planificador de trayectoria (TP) está activado de forma predeterminada.
Si no tiene configuraciones TP en su sección [TRAJ] - LinuxCNC por defecto hace:
ARC_BLEND_ENABLE = 1
ARC_BLEND_FALLBACK_ENABLE = 0
ARC_BLEND_OPTIMIZATION_DEPTH = 50
ARC_BLEND_GAP_CYCLES = 4
ARC_BLEND_RAMP_FREQ = 100

La sección [TRAJ] contiene parámetros generales para el módulo de planificación de trayectoria en *motion*.

- *ARC_BLEND_ENABLE = 1* - Activa el nuevo TP. Si se establece en 0, TP utiliza mezcla parabólica (1 segmento adelantado). Valor predeterminado 1.
- *ARC_BLEND_FALLBACK_ENABLE = 0* - Recurrir opcionalmente a mezclas parabólicas si la velocidad estimada es más rápida. Sin embargo, esta estimación es aproximada y parece que deshabilitarlo proporciona un mejor rendimiento. Valor predeterminado 0.

- *ARC_BLEND_OPTIMIZATION_DEPTH = 50* - profundidad de anticipacion en cantidad de segmentos.

  Para ampliar esto un poco, puede elegir este valor de forma algo arbitraria. Aquí hay una fórmula para estimar cuánta *profundidad* necesita para un determinado config:

  # n = v_max / (2.0 * a_max * t_c) # dónde: # n = profundidad de optimización # v_max = velocidad máxima del eje (UU/seg) # a_max = aceleración máxima del eje (UU/seg) # t_c = período servo (segundos)

  Por tanto, una máquina con una velocidad máxima del eje de 10 IPS, una aceleración máxima de 100 IPS^2, y un período servo de 0.001 seg necesitaría:

  10 / (2.0 * 100 * 0.001) = 50 segmentos para alcanzar siempre la velocidad máxima a lo largo del eje más rápido.

  En la práctica, este número no es tan importante que se sintonice, ya que la anticipacion rara vez necesita toda la profundidad a menos que tenga muchos segmentos muy cortos. Si durante la prueba nota ralentizaciones extrañas y no puede averiguar de dónde vienen, primero intente aumentar esta profundidad usando la fórmula anterior.

  Si aún ve ralentizaciones extrañas, puede deberse a que tiene segmentos cortos en el programa. Si este es el caso, intente agregar una pequeño tolerancia para la detección Naive CAM. Una buena regla general es esta:

  # min_length ~= v_req * t_c # dónde: # v_req = velocidad deseada en UU/seg # t_c = servo período (segundos)

  Si desea recorrer un camino a 1 IPS = 60 IPM, y su período servo es de 0.001 segundos, entonces cualquier segmento más corto que min_length ralentizará la trayectoria. Si configura la tolerancia Naive CAM a aproximadamente esta longitud mínima, los segmentos demasiado cortos se combinarán para eliminar este embotellamiento. Por supuesto, establecer la tolerancia demasiado alta significa una gran desviaciones, por lo que debe jugar un poco para encontrar un buen valor. como consejo, comience en 1/2 de la longitud mínima, luego continúe ajuastando según sea necesario.

- *ARC_BLEND_GAP_CYCLES = 4* Qué tan corto debe ser el segmento anterior antes de que el planificador de trayectorias lo *consuma*.

  A menudo, una combinación de arco circular dejará segmentos de línea cortos entre mezclas. Como la geometría tiene que ser circular, no podemos mezclar toda una línea si la siguiente es un poco más corta. Puesto que el planificador de trayectoria tiene que tocar cada segmento al menos una vez, significa que segmentos muy pequeños ralentizarán las cosas significativamente. Mi solución a esta manera de "consumir" el segmento corto haciéndolo parte del arco de mezcla. Ya que la línea + mezcla es un segmento, no tenemos que reducir la velocidad para alcanzar el segmento muy corto. Probablemente, no necesitará tocar esta configuración.

- *ARC_BLEND_RAMP_FREQ = 20* - Esta es una frecuencia de *corte* para usar velocidades en rampa.

  Velocidad en rampa en este caso solo significa aceleración constante sobre el segmento entero. Esto es menos óptimo que un perfil de velocidad trapezoidal, ya que la aceleración no está maximizada. Sin embargo, si el segmento es lo suficientemente corto, no hay suficiente tiempo para acelerar mucho antes de alcanzar el siguiente segmento. Recordemos los segmentos de línea corta de los anteriores ejemplo. Como son líneas, no hay aceleración en las curvas, así que somos libres de acelerar hasta la velocidad solicitada. Sin embargo, si esta línea está entre dos arcos, entonces tendrá que desacelerar rápidamente nuevamente para estar dentro de la velocidad máxima del siguiente segmento. Esto significa tener un pico de aceleración y luego un pico de desaceleración, causando un gran tirón, para muy poco aumento de rendimiento. Esta configuración es una forma de elimine este tirón para segmentos cortos.

  Básicamente, si un segmento se completa en menos tiempo que 1 / ARC_BLEND_RAMP_FREQ, no nos molestamos con un perfil de velocidad trapezoidal en ese segmento, y usariamos aceleración constante. (Ajustar ARC_BLEND_RAMP_FREQ = 1000 es equivalente a usar siempre aceleración trapezoidal, si el servo loop es de 1kHz).

  Puede caracterizar la pérdida de rendimiento en el peor de los casos comparando la velocidad que alcanza un perfil trapezoidal frente a la rampa:

  # v_ripple = a_max / (4.0 * f) # dónde: # v_ripple = velocidad promedio "pérdida" debido a la rampa # a_max = aceleración máxima del eje # f = frecuencia de corte del INI

  Para la máquina mencionada, la ondulación para una frecuencia de corte de 20Hz es 100 / (4 * 20) = 1.25 IPS. Esto parece alto, pero tenga en cuenta que es solo una estimación del peor de los casos. En realidad, el perfil trapezoidal está limitado por otros factores, como la aceleración normal o velocidad solicitada, por lo que la pérdida de rendimiento real debería ser mucho menor. Aumentar la frecuencia de corte puede dar más rendimiento, pero hace que el movimiento sea más duro debido a discontinuidades de la aceleración. Un valor en el rango de 20Hz a 200Hz debería ser razonable para comenzar.

  Finalmente, ninguna cantidad de ajustes acelerará una trayectoria con muchas esquinas pequeñas y estrechas, ya que está limitado por la aceleración en las esquinas.

- *SPINDLES = 3* - El número de husillos a soportar. Es imperativo que este número coincida con el parámetro "num_spindles" pasado al módulo motion.
- *COORDINATES = X Y Z* - los nombres de los ejes que se controlan. Solo son válidos X, Y, Z, A, B, C, U, V, W . Solo ejes nombrados en *COORDINATES* son aceptados en el código g. Está permitido escribir un nombre de eje dos veces (p. ej., X Y Y Z para una máquina de pórtico). Para las *cinemáticas trivkins* comunes, los números de articulación se asignan en secuencia de acuerdo con el parámetro trivkins *coordinates=*. Por tanto, para trivkins *coordinates = xz*, la articulación 0 corresponde a X y la articulación 1 corresponde a Z. Consulte la página de manual de cinemática (*$ man kins*) para obtener información sobre trivkins y otros módulos de cinemática.
- *LINEAR_UNITS = <unidades>* - Especifica las *unidades máquina* para ejes lineales. Las opciones posibles son mm o pulgadas. Esto no afecta las unidades lineales en el código NC (las G20 y G21 palabras hacen esto).
- *ANGULAR_UNITS = <unidades>* - Especifica las *unidades máquina* para ejes de rotación. Las opciones posibles son *deg*, *degree* (360 por círculo), *rad*, *radian* (2pi por círculo), *grad* o *gon* (400 por círculo). Esto no afecta las unidades angulares del código NC. En RS274NGC,las palabras A-, B- y C- siempre se expresan en grados.
- *DEFAULT_LINEAR_VELOCITY = 0.0167* - La tasa inicial para jogs de ejes lineales, en unidades máquina por segundo. El valor que se muestra en *Axis* es igual a unidades máquina por minuto.
- *DEFAULT_LINEAR_ACCELERATION = 2.0* - en máquinas con cinemática no trivial, la aceleración utilizada para jog "teleop" (espacio cartesiano), en *unidades máquina* por segundo al cuadrado.
- *MAX_LINEAR_VELOCITY = 5.0* - La velocidad máxima para cualquier eje o movimiento coordinado, en *unidades máquina* por segundo. El valor mostrado es igual a 300 unidades por minuto.
- *MAX_LINEAR_ACCELERATION = 20.0* - La aceleración máxima para cualquier eje o movimiento coordinado, en *unidades máquina* por segundo cuadrado.
- *POSITION_FILE = position.txt* - si se establece en un valor no vacío, las posiciones articulares se almacenan entre ejecuciones en este archivo. Esto permite que la máquina comience con el mismo coordenadas que tenía en el apagado. Esto supone que no hubo movimiento de la máquina mientras está apagada. Si no se establece, las posiciones no se almacenan y comenzará en 0 cada vez que se inicie LinuxCNC. Esto puede ayudar en pequeñas máquinas sin interruptores home. Si usa la interfaz de resolver de Mesa, este archivo se puede usar para emular codificadores absolutos y eliminar la necesidad de home (sin pérdida de precisión). Ver la página de manual de hostmot2 para más detalles.
- *NO_FORCE_HOMING = 1* - el comportamiento predeterminado es que LinuxCNC fuerce al usuario a iniciar la máquina antes de ejecutar cualquier programa o comando MDI. Normalmente, solo se permite jog antes de homing. Para configuraciones usando cinemática de identidad, establecer NO_FORCE_HOMING = 1 permite al usuario hacer movimientos MDI y ejecuta programas sin homing previo de la máquina. Interfaces que usen cinemática de identidad sin capacidad de búsqueda de home necesitarán tener esta opción establecida en 1.
- *HOME = 0 0 0 0 0 0 0 0 0* - Se necesita una posición de inicio mundial para los módulos de cinemática que calculan las coordenadas mundiales usando kinematicsForward() al cambiar de modo articular a teleop. Hasta nueve valores de coordenadas (X Y Z A B C U V W) pueden especificarse; los elementos no utilizados pueden omitirse. Este valor es solo utilizado para máquinas con cinemática no trivial. En máquinas con cinemática trivial (fresadoras, tornos, varios tipos de pórtico) este valor se ignora. Nota: la configuración sim de hexapod requiere un valor distinto de cero para la coordenada Z.

aviso
LinuxCNC no conocerá sus límites de articulaciones cuando use *NO_FORCE_HOMING = 1*.

8.1.2.11. Sección [KINS]

- *JOINTS = 3* - especifica el número de articulaciones (motores) en el sistema. Por ejemplo, una máquina trivkins XYZ con un solo motor para cada eje tiene 3 articulaciones. Una máquina de pórtico con un motor en cada uno de los dos ejes, y dos motores en el tercer eje, tiene 4 articulaciones. (Esta variable de configuración puede ser utilizada por una interfaz gráfica de usuario para establecer el número de articulaciones (num_joints) especificado en el módulo de movimiento (motmod)). La interfaz gráfica de usuario Axis, pncconf y stepconf usan este elemento.

- *KINEMATICS = trivkins* - especifica un módulo de cinemática para el módulo motion. Las Guis puede usar esta variable para especificar la línea de carga en archivos hal para el módulo motmod. Para obtener más información sobre los módulos de cinemática, consulte la página de manual: *$ man kins*

8.1.2.12. Sección [AXIS_<letter>]

El <letter> especifica uno de: X Y Z A B C U V W

- *MAX_VELOCITY = 1.2* - Velocidad máxima para este eje en unidades máquina por segundo.
- *MAX_ACCELERATION = 20.0* - Aceleración máxima para este eje en unidades máquina por segundo cuadrado
- *MIN_LIMIT = -1000* - El límite mínimo (límite soft) para el movimiento del eje, en unidades máquina. Cuando se excede este límite, el controlador aborta el movimiento del eje.
- *MAX_LIMIT = 1000* - El límite máximo (límite suave) para el movimiento del eje, en unidades de máquina. Cuando se excede este límite, el controlador aborta el movimiento del eje.
- *WRAPPED_ROTARY = 1* - Cuando se establece en 1 para una articulación ANGULAR, la articulación se moverá 0-359.999 grados. Los números positivos moverán la articulación en una dirección positiva y los números negativos moverán la articulación en la dirección negativa.
- *LOCKING_INDEXER_JOINT = 4* - este valor selecciona una articulación para usar un indexador de bloqueo para el eje<letter> especificado. En este ejemplo, la articulación es 4, que correspondería al eje B para un sistema XYZAB con cinemática trivkins (identidad). Cuando se establece, un movimiento G0 para este eje iniciará un desbloqueo con el pin de desbloqueo joint.4.unlock y luego espera el pin joint.4.is-unlocked. Luego mueve la articulación a velocidad rápida para esa articulación. Después del movimiento, joint.4.unlock será falso y el movimiento esperará a que joint.4.is-unlocked se vuelva falso. No se permite mover otras articulaciones al mover un articulación rotativa de bloqueo. Para crear los pines de desbloqueo, use el parámetro motmod:

unlock_joints_mask=jointmask

Los bits jointmask son: (LSB) 0: articulación0, 1: articulación1, 2: articulación2, . . .

Ejemplo: loadrt motmod . . . unlock_joints_mask = 0x38 crea pines de desbloqueo para articulaciones 3,4,5

- *OFFSET_AV_RATIO = 0.1* - si no es cero, este elemento permite el uso de pines Hal de entrada para compensaciones de eje externas:

axis.<letter>.eoffset-enable axis.<letter>.eoffset-count axis.<letter>.eoffset-scale

Consulte el capítulo: Offsets Externos de Ejes para información de su uso.

8.1.2.13. Sección [JOINT_<num>]

<num> especifica el número de articulación 0 . . . (num_joints-1) El valor de *num_joints* lo establece [KINS]JOINTS =

Las secciones [JOINT_0], [JOINT_1], etc. contienen parámetros generales para los componentes individuales en el módulo de control de articulaciones. Los nombres en la sección comienzan a numerarse en 0 y llegan hasta el número de articulaciones especificado en la entrada [KINS]JOINTS menos 1.

Típicamente (para sistemas que usan *cinemática trivkins*, hay correspondencia 1:1 entre una articulación y un eje):

- JOINT_0 = X
- JOINT_1 = S
- JOINT_2 = Z
- JOINT_3 = A

- JOINT_4 = B
- JOINT_5 = C
- JOINT_6 = U
- JOINT_7 = V
- JOINT_8 = W

Otros módulos de cinemática con cinemática de identidad están disponibles para admitir configuraciones con conjuntos parciales de ejes. Por ejemplo, usando trivkins con coordenadas = XZ, las relaciones de ejes comunes son:

- JOINT_0 = X
- JOINT_1 = Z

Para obtener más información sobre los módulos cinemáticos, consulte la página de manual: *$ man kins*

- *TYPE = LINEAR* - El tipo de articulación, ya sea LINEAR o ANGULAR.
- *UNITS = INCH* - Si se especifica, esta configuración, se anula la configuración relacionada [TRAJ]UNITS. (por ejemplo, [TRAJ] LINEAR_UNITS si el TYPE de esta articulación es LINEAR, [TRAJ]ANGULAR_UNITS si el TYPE de esta articulación es ANGULAR)
- *MAX_VELOCITY = 1.2* - Velocidad máxima para esta articulación en unidades máquina por segundo.
- *MAX_ACCELERATION = 20.0* - Aceleración máxima para esta articulación en unidades máquina por segundo cuadrado
- *BACKLASH = 0.0000* - Backlash en unidades de máquina. El valor de Backlash se puede utilizar para compensar pequeñas deficiencias en el hardware utilizado para conducir una articulacion. Si se agrega Backlash a una articulación y está utilizando paso a paso, STEPGEN_MAXACCEL debe aumentarse de 1,5 a 2 veces del valor de MAX_ACCELERATION para la articulación. La compensación de Backlash excesiva puede causar sacudidas en el eje a medida que cambia de dirección. Si se especifica un COMP_FILE para un eje, BACKLASH no se utiliza.

- *COMP_FILE = file.extension* - El archivo de compensación consiste en un mapa de información de posición para la articulación. Los valores del archivo de compensación están en unidades máquina. Cada conjunto de valores está en una línea separada por un espacio. El primer valor es el valor nominal (la posición ordenada). El segundo y tercer valor dependerá de la configuración de COMP_FILE_TYPE. Los puntos entre valores nominales están interpolados entre los dos nominales. Los archivos de compensación deben comenzar con el mínimo nominal y estar en orden ascendente hasta el mayor valor de los nominales. Los nombres de archivo distinguen entre mayúsculas y minúsculas y pueden contener letras y/o números. Actualmente, el límite dentro de LinuxCNC es de 256 tripletas por eje.

  Si se especifica COMP_FILE para un eje, BACKLASH no se utiliza. Se debe especificar UN *COMP_FILE_TYPE* para cada *COMP_FILE*.

- *COMP_FILE_TYPE = 0 o 1* - especifica el tipo de archivo de compensación. El primer valor es la posición nominal (ordenada) para ambos tipos.

  - *Tipo 0:* El segundo valor especifica la posición real a medida que se mueve el eje en la dirección positiva (valor creciente) y el tercer valor especifica la posición real a medida que el eje se mueve en la dirección negativa (valor decreciente).

    Ejemplo Tipo 0

    ```
    -1.000 -1.005 -0.995
    0.000 0.002 -0.003
    1.000 1.003 0.998
    ```

  - *Tipo 1:* El segundo valor especifica el desplazamiento positivo del nominal mientras se va en la dirección positiva. El tercer valor especifica el negativo compensado del nominal mientras se va en una dirección negativa.

    Ejemplo de tipo 1

```
-1.000 0.005 -0.005
0.000 0.002 -0.003
1.000 0.003 -0.004
```

- *MIN_LIMIT = -1000* - El límite mínimo para el movimiento del eje, en unidades máquina. Cuando se alcanza este límite, el controlador aborta el movimiento del eje. El eje debe tener home antes de que MIN_LIMIT esté en vigor. Para un eje rotativo con rotación ilimitada que no tiene MIN_LIMIT para ese eje en [JOINT_n], entonces se usa el valor -1e99.
- *MAX_LIMIT = 1000* - El límite máximo para el movimiento del eje, en unidades máquina. Cuando se alcanza este límite, el controlador aborta el movimiento del eje. El eje debe tener home antes de que MAX_LIMIT esté en vigor. Para un eje rotativo con rotación ilimitada que no tiene MAX_LIMIT para ese eje en [JOINT_n], se usa el valor 1e99.
- *MIN_FERROR = 0.010* - Este es el valor en unidades máquina que el eje puede desviarse de la posición ordenada a muy bajas velocidades. Si MIN_FERROR es más pequeño que FERROR, los dos producen una rampa de puntos de disparo de error. Podría pensar en esto como un gráfico donde una dimensión es velocidad y el otro el error de seguimiento permitido. A medida que la velocidad aumenta, la cantidad de error de seguimiento también aumenta hacia el valor FERROR.
- *FERROR = 1.0* - FERROR es el error de seguimiento máximo permitido, en unidades máquina. Si la diferencia entre la posición ordenada y la detectada excede esta cantidad, el controlador deshabilita los cálculos servo, establece todas las salidas a 0.0, y desactiva los amplificadores. Si MIN_FERROR está presente en el archivo .ini, se utilizan los siguientes errores proporcionales a la velocidad. Aquí el error de seguimiento máximo permitido es proporcional a la velocidad, con FERROR aplicando a la tasa rápida establecida por [TRAJ]MAX_VELOCITY, y proporcionalmente errores de seguimiento más pequeños para velocidades más lentas. El error de seguimiento máximo permitido siempre será mayor que MIN_FERROR. Esto evita pequeños errores de seguimiento para ejes estacionarios al abortar inadvertidamente el movimiento. Pequeños errores de seguimiento siempre estarán presentes debido a la vibración, etc.
- *LOCKING_INDEXER = 1* - Indica que la articulación se utiliza como indexador con bloqueo.

Homing Estos parámetros están relacionados con Homing; para una mejor explicación lea el Capítulo Configuración Homing.

- *HOME = 0.0* - La posición a la que irá la articulación al finalizar la secuencia homing.
- *HOME_OFFSET = 0.0* - La posición articular del interruptor home o pulso índice, en unidades máquina. Cuando se encuentra el punto home durante el proceso homing, esta es la posición asignada a ese punto. Al compartir interruptores home y de límite y usar una secuencia home que deje el interruptor home/límite en el estado activado, el offset home puede ser utilizado para definir la posición del interruptor home para que sea diferente de 0 si se desea que la posición home sea 0.
- *HOME_SEARCH_VEL = 0.0* - Velocidad de homing inicial en unidades de máquina por segundo. El signo indica la dirección de recorrido. Un valor de cero significa asumir que la ubicación actual es la posición de inicio de la máquina. Si su máquina no tiene interruptores de inicio querrá dejar este valor en cero.
- *HOME_LATCH_VEL = 0.0* - Velocidad de homing en unidades máquina por segundo a la posicion de enclavamiento del interruptor home. El signo indica la dirección del recorrido.
- *HOME_FINAL_VEL = 0.0* - Velocidad en unidades de máquina por segundo desde la posición de enclavamiento a la posición home. Si se deja en 0 o no se incluye en la articulación, se usa la velocidad rápida. Debe ser un número positivo.
- *HOME_USE_INDEX = NO* - Si el codificador utilizado para esta articulación tiene un pulso índice, y la electronica tiene provisión para esta señal, puede configurarla en YES. Cuando es YES, se afectará el tipo de patrón de inicio utilizado. Actualmente no puede indexar con steppers a menos que esté usando stepgen en modo de velocidad y PID.
- *HOME_INDEX_NO_ENCODER_RESET = NO* - Use YES si el codificador utilizado para esta articulación no restablece su contador cuando se detecta un pulso índice después de la activacion del pin hal index_enable. Aplicable solo para HOME_USE_INDEX = YES.
- *HOME_IGNORE_LIMITS = NO* - Cuando usa el interruptor de límite tambien como interruptor home, esto debe establecerse en YES. Cuando se establece en YES, el interruptor de límite para esta articulación se ignora durante homing. Debe configurar su homing para que al final del movimiento a home el interruptor home/límite no esté en el estado activado; recibiria un error de interruptor de límite después del homing.

- *HOME_IS_SHARED = <n>* - Si la entrada home es compartida por más de una articulacion, haga <n> igual a 1 para evitar que se inicie homing si uno de los conmutadores compartidos está ya está cerrado. Establezca <n> en 0 para permitir el homing si un interruptor está cerrado.
- *HOME_ABSOLUTE_ENCODER = 0 | 1 | 2 '- Usado para indicar que la articulación usa un codificador absoluto. A una petición de homing, el valor de la articulacion actual se establece en el valor 'HOME_OFFSET*. Si la configuración *HOME_ABSOLUTE_ENCODER* es 1, la máquina hace el habitual movimiento final al valor *HOME*. Si la configuración *HOME_ABSOLUTE_ENCODER* es 2, no se realiza ningún movimiento final.
- *HOME_SEQUENCE = <n>* - Se utiliza para definir la secuencia "Home Todo". <n> debe comenzar en 0 o 1 o -1. Se pueden especificar secuencias adicionales con números crecientes de 1 en 1 (en valor absoluto). No se permite omitir los números de secuencia. Si se omite una HOME_SEQUENCE, la articulación no será homeada por la función "Home Todo". Se puede homear más de una articulación al mismo tiempo especificando el mismo número de secuencia para más de una articulación. Se utiliza un número de secuencia negativa para diferir el movimiento final para todas las articulaciones que tienen ese número de secuencia (negativo o positivo). Para obtener información adicional, consulte: SECUENCIA HOME
- *VOLATILE_HOME = 0* - Cuando se habilita (se establece en 1), esta articulación no se homeara si la alimentación de la máquina está apagada o si E-Stop está encendido. Esto es útil si su máquina tiene interruptores Home y no tiene retroalimentación de posición, como en máquina paso y dirección.

Servo Estos parámetros son relevantes para las articulaciones controladas por servos.

> **aviso**
> Las siguientes son entradas de archivos INI personalizadas que puede encontrar en un archivo INI de muestra o un archivo generado por asistente. Estos no son utilizados por el software LinuxCNC. Solo están ahí para poner todas las configuraciones en un solo lugar. Para más información sobre entradas de archivo INI personalizadas ver la subsección Secciones y Variables personalizadas .

Los siguientes elementos pueden ser utilizados por un componente PID y se supone que la salida es voltios.

- *DEADBAND = 0.000015* - qué tan cerca es "suficientemente cerca" como para considerar el motor en posición, en unidades máquina. Esto a menudo se establece en una distancia equivalente a 1, 1.5, 2, o 3 recuentos de codificador, pero no hay reglas estrictas. Las configuraciones más grandes permiten menos *hunting (caza)* de servos a expensas de una menor precisión. Las configuraciones más estrictas (más pequeñas) intentan una mayor precisión a expensas de más *hunting*. ¿Es realmente más preciso si también es más incierto? Como regla general, es bueno evitar si puede, o al menos limitar, el *hunting* de servos.

Tenga cuidado al ir por debajo de 1 recuento de codificador, ya que puede crear una condición donde no hay lugar donde su servo esté satisfecho. Esto puede ir más allá de *hunting* (lento) a *nervous* (rápido), e incluso *squealing* (estrepitoso), que es fácil de confundir con la oscilación causada por un ajuste incorrecto. Es mejor perder un conteo o dos al principio, hasta que se haya pasado por una *afinación bruta* al menos.

Ejemplo de cálculo de unidades máquina por pulso de codificador para usar al decidir el valor de DEADBAND:

// ///////// latexmath: [\ frac {X \, pulgadas} {1 \, encoder \, count} = \ frac {1 \, revolution} {1000 \, encoder \, lines} \ times \ frac {1 \, codificador \, línea} {4 \, cuadratura \, conteos} \ veces \ frac {0.200 \, pulgadas} {1 \, revolución} = \ frac {0.200 \, pulgadas} {4000 \, encoder \, recuentos} = \ frac {0.000050 \, pulgadas} {1 \, encoder \, count}] // /////////

imagen::images/encoder-count-math.png[align="center"]

- *BIAS = 0.000* - Esto es utilizado por hm2-servo y algunos otros. BIAS es una cantidad constante que se agrega a la salida. En la mayoría de los casos, debe dejarse en cero. Sin embargo, a veces puede ser útil para compensar servoamplificadores, o para equilibrar el peso de un objeto que se mueve verticalmente. BIAS se desactiva cuando el bucle PID está desactivado, al igual que todos los demás componentes de la salida.
- *P = 50* - La ganancia proporcional para el servo. Este valor multiplica el error entre la posición ordenada y la real en unidades máquina, lo que resulta en una contribución a la tensión calculada para el amplificador del motor. Las unidades en la ganancia P son voltios por unidad máquina, por ejemplo, $\frac{volts}{unit}$

- *I = 0* - La ganancia integral para el servo. El valor multiplica el error acumulativo entre la posición ordenada y la real en unidades máquina, lo que resulta en una contribución a la tensión calculada para el amplificador de motor. Las unidades en la ganancia I son voltios por unidad máquina por segundo, por ejemplo, $\frac{volts}{unit\ second}$

- *D = 0* - La ganancia derivada para el servo. El valor multiplica la diferencia entre los errores actuales y anteriores, lo que resulta en una contribución a la tensión calculada para el amplificador del motor. las unidades en la ganancia D son voltios por unidad de máquina por segundo, por ejemplo, $\frac{volts}{unit\ second}$

- *FF0 = 0* - ganancia de avance de orden 0. Este numero es multiplicado por el posición ordenada, lo que resulta en una contribución a la tensión calculada para el amplificador del motor. Las unidades en la ganancia FF0 son voltios por unidad máquina, por ejemplo, $\frac{volts}{unit}$

- *FF1 = 0* - ganancia de avance de 1er orden. Este numero es multiplicado por el cambio en la posición ordenada por segundo, lo que resulta en una contribución al voltaje calculado para el amplificador del motor. Las unidades en FF1 son voltios por unidad máquina por segundo, por ejemplo, $\frac{volts}{unit\ second}$

- *FF2 = 0* - ganancia de avance de segundo orden. Este numero es multiplicado por el cambio en la posición ordenada por segundo por segundo, lo que resulta en un contribución a la tensión calculada para el amplificador del motor. Las unidades en la ganancia FF2 son voltios por unidad máquina por segundo por segundo, por ejemplo, $\frac{volts}{unit\ second^2}$

- *OUTPUT_SCALE = 1.000* -

- *OUTPUT_OFFSET = 0.000* - estos dos valores son los factores de escala y offset para la salida a los amplificadores del motor. El segundo valor (offset) se resta de la salida calculada (en voltios), y se divide por el primer valor (escala), antes de ser escrito en los convertidores D/A. Las unidades de los valores de escala están en voltios verdaderos por voltios de salida DAC. Las unidades del valor de offset está en voltios. Estos se pueden usar para linealizar un DAC. Específicamente, al escribir salidas, LinuxCNC primero convierte la salida deseada en unidades cuasi-SI a valores de actuador sin procesar, por ejemplo, voltios para un amplificador DAC. Esta escala se parece a: $raw = \frac{output - offset}{scale}$

El valor de la escala se puede obtener analíticamente haciendo un análisis de unidades, es decir, las unidades son [unidades SI de salida]/[unidades de actuador]. Por ejemplo, en una máquina con un amplificador de modo de velocidad tal que 1 voltio da como resultado una velocidad de 250 mm/seg.

$$amplifier[volts] = (output[\frac{mm}{sec}] - offset[\frac{mm}{sec}])/250\ \frac{mm}{secvolt}$$

Tenga en cuenta que las unidades del offset están en unidades máquina, por ejemplo, mm/seg, y se restan previamente de las lecturas del sensor. El valor para este offset se obtiene al encontrar el valor de su salida que produce 0.0 para la salida del actuador. Si el DAC está linealizado, este offset es normalmente 0.0.

La escala y el offset también se pueden usar para linealizar el DAC, resultando en valores que reflejan los efectos combinados de la ganancia del amplificador, no linealidad del DAC, unidades DAC, etc.

Para hacer esto, siga este procedimiento.

1. Cree una tabla de calibración para la salida, alimentando el DAC con el voltaje deseado y midiendo el resultado.

2. Haga un ajuste lineal de mínimos cuadrados para obtener los coeficientes a, b tales como $measured = a * raw + b$

3. Tenga en cuenta que queremos una salida en bruto de modo que nuestro resultado medido sea idéntico a la salida ordenada. Esto significa
   a. $command = a * raw + b$
   b. $raw = (command - b)/a$
4. Como resultado, los coeficientes a y b del ajuste lineal pueden ser utilizado como la escala y el offset para el controlador directamente.

La siguiente tabla es un ejemplo de mediciones de voltaje.

Mediciones de voltaje de salida

Raw	Medido
-10	-9.93
-9	-8.83
0	-0.03
1	0.96
9	9.87
10	10.87

- *MAX_OUTPUT = 10* - El valor máximo para la salida de la compensación PID que se escribe en el amplificador del motor, en voltios. El valor calculado de salida está sujeto a este límite. El límite se aplica antes de escalado a unidades de salida en bruto. El valor se aplica simétricamente tanto al lado positivo como al negativo.
- *INPUT_SCALE = 20000* - en configuraciones de muestra
- *ENCODER_SCALE = 20000* - en configuraciones construidas con PNCconf Especifica el número de pulsos que corresponde a un movimiento de una unidad máquina como se establece en la sección [TRAJ]. Para una articulación lineal, una unidad máquina será igual a la configuración de LINEAR_UNITS. Para una articulación angular, una unidad es igual a la configuración en ANGULAR_UNITS. Un segundo número, si se especifica, se ignora. Por ejemplo, en un codificador de 2000 cuentas por revolucion, y una transmision de 10 revoluciones por pulgada y unidades de pulgada, tenemos:

$$input\,scale = 2000\frac{counts}{rev} * 10\frac{rev}{inch} = 20000\frac{counts}{inch}$$

Stepper Estos parámetros son relevantes para las articulaciones controladas por steppers.

aviso
Las siguientes son entradas de archivos INI personalizadas que puede encontrar en un archivo INI de muestra o un archivo generado por el asistente. Estos no son utilizados por el software LinuxCNC. Solo están ahí para poner todas las configuraciones en un solo lugar. Para más información sobre entradas de archivo INI personalizadas ver la subsección Secciones y Variables personalizadas .

Los siguientes elementos pueden ser utilizados por un componente stepgen.

- *SCALE = 4000* - en configuraciones de muestra
- *STEP_SCALE = 4000* - en configuraciones construidas con PNCconf Especifica el número de pulsos que corresponde a un movimiento de una unidad máquina como se establece en la sección [TRAJ]. Para sistemas paso a paso, esto es el número de pulsos de paso emitidos por unidad máquina. Para una articulación lineal una unidad de máquina será igual a la configuración de LINEAR_UNITS. Para una articulacion angular es igual a la configuración en ANGULAR_UNITS. Para servo sistemas, este es el número de pulsos de retroalimentación por unidad máquina. Un segundo número, si se especifica, se ignora.

Por ejemplo, en un motor paso a paso de 1.8 grados con semipasos, y transmision de 10 revoluciones por pulgada, y deseado unidades máquina en pulgada, tendriamos:

$$input\,scale = \frac{2\,steps}{1.8\,degrees} * 360\,\frac{degree}{rev} * 10\,\frac{rev}{inch} = 4000\,\frac{steps}{inch}$$

- *ENCODER_SCALE = 20000* (Opcionalmente utilizado en configuraciones construidas con PNCconf) - Especifica el número de pulsos que corresponde a un movimiento de una unidad máquina como se establece en la sección [TRAJ]. Para una articulación lineal, una unidad máquina será igual a la configuración de LINEAR_UNITS. Para una articulación angular, una unidad es igual a la configuración en ANGULAR_UNITS. Un segundo número, si se especifica, se ignora. Por ejemplo, en un conteo de 2000 por revolucion de codificador, transmision de 10 revoluciones por pulgada, y unidades en pulgada, tener:

$$input\,scale = 2000\,\frac{counts}{rev} * 10\,\frac{rev}{inch} = 20000\,\frac{counts}{inch}$$

- *STEPGEN_MAXACCEL = 21.0* - Límite de aceleración para el generador de pasos. Esto debería ser entre 1 % y 10 % más grande que MAX_ACCELERATION de la articulación. Este valor mejora la afinación del "bucle de posición" de stepgen. Si ha agregado compensación backlash a una articulación, entonces esto debería ser 1.5 a 2 veces mayor que MAX_ACCELERATION.
- *STEPGEN_MAXVEL = 1.4* - los archivos de configuración antiguos tienen también un límite de velocidad para el generador de pasos. Si se especifica, también debería ser entre 1 % y 10 % más grande que MAX_VELOCITY de la articulación. Pruebas posteriores han demostrado que el uso de STEPGEN_MAXVEL no mejora el ajuste del bucle de posición de stepgen.

8.1.2.14. Sección [EMCIO]

- *EMCIO = io* - Nombre del programa controlador IO
- *CYCLE_TIME = 0.100* - El período, en segundos, en el que se ejecutará EMCIO. Haciendolo 0.0 o un número negativo le dirá a EMCIO que no duerma en absoluto. Generalmente no es necesario cambiar este número.
- *TOOL_TABLE = tool.tbl* - El archivo que contiene información sobre herramientas, descrito en el manual de usuario.
- *TOOL_CHANGE_POSITION = 0 0 2* - Especifica la ubicación XYZ a la que moverse al realizar un cambio de herramienta si se utilizan tres dígitos. Especifica la ubicación XYZABC cuando se usan 6 dígitos. Especifica la ubicación XYZABCUVW cuando se utilizan 9 dígitos. Los cambios de herramienta se pueden combinar. Por ejemplo, si combina la pinola con la posición de cambio, puede mover primero la Z y luego la X e Y.
- *TOOL_CHANGE_WITH_SPINDLE_ON = 1* - El husillo se dejará encendido durante el cambio de herramienta cuando el valor sea 1. Útil para tornos o máquinas donde el material está en el husillo, no la herramienta.
- *TOOL_CHANGE_QUILL_UP = 1* - El eje Z se moverá a cero máquina antes del cambio de herramienta cuando el valor es 1. Esto es lo mismo que emitir un G0 G53 Z0.
- *TOOL_CHANGE_AT_G30 = 1* - La máquina se mueve al punto de referencia definido por los parámetros 5181-5186 para G30 si el valor es 1. Para obtener más información, consulte el sección de parámetros y <<gcode:g30-g30.1.
- *RANDOM_TOOLCHANGER = 1* - Esto es para máquinas que no pueden volver a colocar la herramienta en la ranura de la que vino. Por ejemplo, máquinas que intercambian la herramienta en la ranura activa con la herramienta en el husillo.

= Configuracion de Homing

8.1.3. Descripción general

Homing establece el origen cero de las coordenadas máquina G53.
Los límites soft se definen en relación con el origen de la máquina.
Los límites soft automáticamente desaceleran y detienen los ejes antes de que toquen los interruptores de límites.
Una máquina configurada y funcionando correctamente no se moverá más allá de los límites soft (software) y
tendrá el origen de la máquina configurado tan repetible como lo permita el mecanismo switch/index de home.
Linuxcnc se puede hacer home a ojo (marcas de alineación), con interruptores, con interruptores e índice de codificador, o

utilizando codificadores absolutos.
El recorrido de homing parece bastante simple: simplemente mueve cada articulación a una ubicación conocida, y establece las variables internas de LinuxCNC en consecuencia.
Sin embargo, diferentes máquinas tienen diferentes requisitos, y el recorrido de homing es realmente bastante complicado.

nota
Si bien es posible usar linuxcnc sin interruptores, procedimientos de homing o interruptores de límite, la seguridad adicional de los límites soft no es suficiente.

8.1.4. Prerrequisitos

Homing se basa en algunos supuestos fundamentales de la máquina.

- Las direcciones negativa y positiva se basan en Movimientos de la herramienta que puede ser diferentes del movimiento real de la máquina. Es decir, en una fresadora, la mesa se mueve en lugar de la herramienta.
- Todo está referenciado desde el origen G53, cero de la máquina. El origen puede estar en cualquier lugar (incluso fuera de donde puede moverse)
- El origen cero de la máquina G53 está típicamente dentro del área de límites soft, pero no necesariamente.
- El offset del interruptor home establece dónde está el origen, pero incluso esto se referencia desde el origen.
- Los límites soft negativos son lo máximo que puede mover en la dirección negativa después de homing.
  (pero pueden no ser negativos en sentido absoluto)
- Los límites soft positivos son lo máximo que puede mover en la dirección positiva después de homing.
  (pero podrían no ser positivos en sentido absoluto, aunque es habitual establecerlo como un número positivo)
- Los límites software están dentro del área de interruptores de límite.
- (Si se utiliza homing basado en conmutadores), los conmutadores de homing utilizan los interruptores de límite (interruptor de home/límite compartido),
  o cuando se usa un interruptor home separado, están dentro del área de interruptores de límite.
- Si usa un interruptor home independiente, es posible comenzar a buscar en el lado equivocado del interruptor,
  lo que combinado con la opción HOME_IGNORE_LIMITS, provocará un bloqueo grave.
  Puede evitar esto haciendo que el interruptor home cambie su estado cuando el disparador está en un lado en particular hasta que vuelva a pasar el punto nuevamente.
  Dicho de otra manera, el estado del interruptor home debe representar la posición del disparador con respecto al interruptor (es decir, antes o después del interruptor),
  y debe permanecer así incluso si el disparador pasa por el interruptor en la misma dirección.

8.1.5. Ejemplo de Diseño de interruptor home separado

Este ejemplo muestra los interruptores de límite mínimo y máximo con un interruptor home separado.

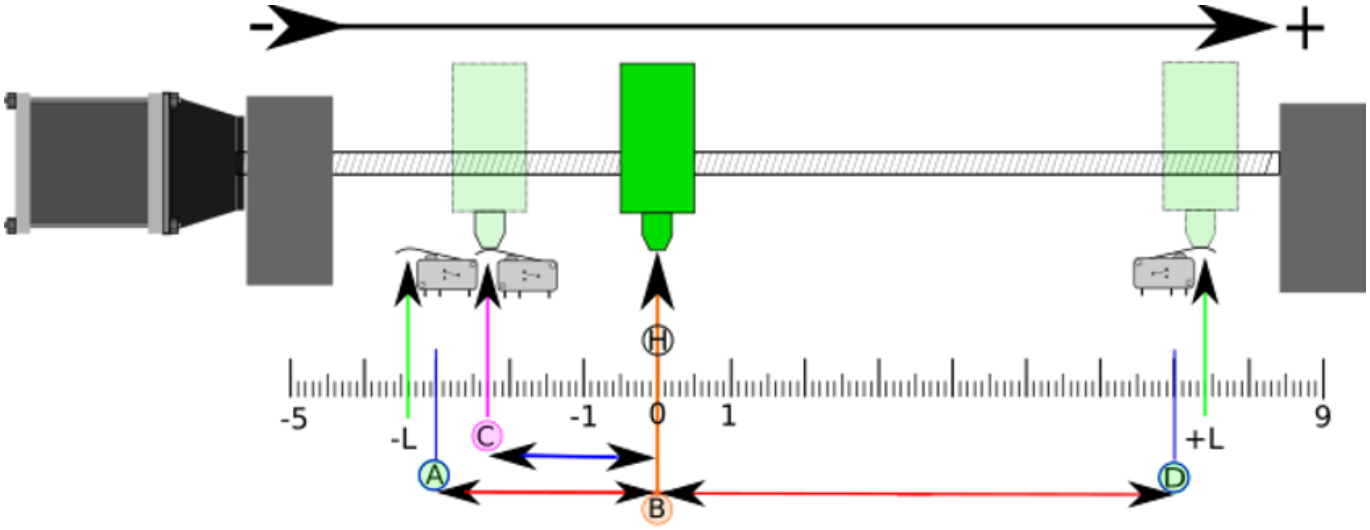


Figura 8.1: Diseño de demostrativo interruptor separado

- A es el límite soft negativo
- B es la coordenada Origen de máquina G53
- C es el punto de disparo del interruptor home
- D es el límite soft positivo
- H es la posición home final (HOME) = 0 unidades
- -L y +L son los puntos de disparo de los interruptores de límite
- A<->B son los límites soft negativos (MIN_LIMITS) = -3 unidades
- B<->C es el home_offset (HOME_OFFSET) = -2.3 unidades
- B<->D son los límites soft positivos (MAX_LIMITS) = 7 unidades
- A<->D es el recorrido total = 10 unidades
- La distancia entre los interruptores de límite y los límites soft (-L<->A y D<-+L) se amplía en este ejemplo
- Tenga en cuenta que hay una distancia entre los interruptores de límite y el contacto físico real para la inercia después de que el amplificador esté desactivado.

nota
Homing establece el sistema de coordenadas G53, mientras que el origen de la máquina (punto cero) puede estar en cualquier lugar.
Establecer el punto cero en el límite soft negativo hace que todas las coordenadas G53 sean positivas,
lo cual es probablemente más fácil de recordar. Haga esto configurando MIN_LIMIT = 0 y asegúrese de que MAX_LIMIT sea positivo.

8.1.6. Ejemplo de Diseño de Límite/Home compartido

Este ejemplo muestra un interruptor de límite máximo y un interruptor combinado de límite mínimo/home.

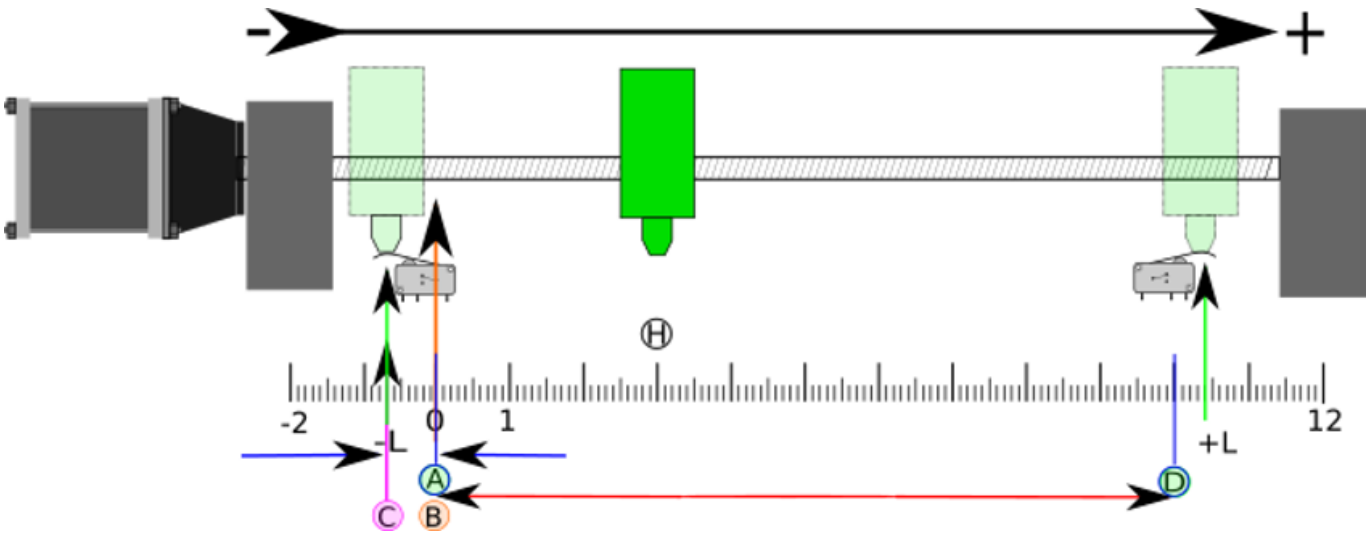


Figura 8.2: Diseño demostrativo de conmutador compartido

- A es el límite soft negativo
- B es la coordenada G53 Origen de máquina
- C es el punto de disparo del interruptor home compartido con (-L) límite mínimo
- D es el límite soft positivo
- H es la posición final (HOME) = 3 unidades
- -L y +L son los puntos de disparo del interruptor de límite
- A<->B son los límites soft negativos (MIN_LIMITS) = 0 unidades
- B<->C es el home_offset (HOME_OFFSET) = -0.7 unidades
- B<->D son los límites soft positivos (MAX_LIMITS) 10 unidades
- A<->D es el recorrido total = 10 unidades
- La distancia entre los interruptores de límite y los límites soft (-L<->A y D<->+L) se amplía en este ejemplo
- Tenga en cuenta que hay una distancia entre los interruptores de límite y el contacto físico real para la inercia después de que el amplificador esté desactivado.

8.1.7. Secuencia Homing

Hay cuatro posibles secuencias homing definidas por el signo de HOME_SEARCH_VEL y HOME_LATCH_VEL, junto con los parámetros de configuración asociados como se muestra en la siguiente tabla. Existen dos condiciones básicas, HOME_SEARCH_VEL y HOME_LATCH_VEL son el mismo signo o son signos opuestos. Para una descripción detallada de lo que hace cada parámetro de configuración, vea la sección siguiente.

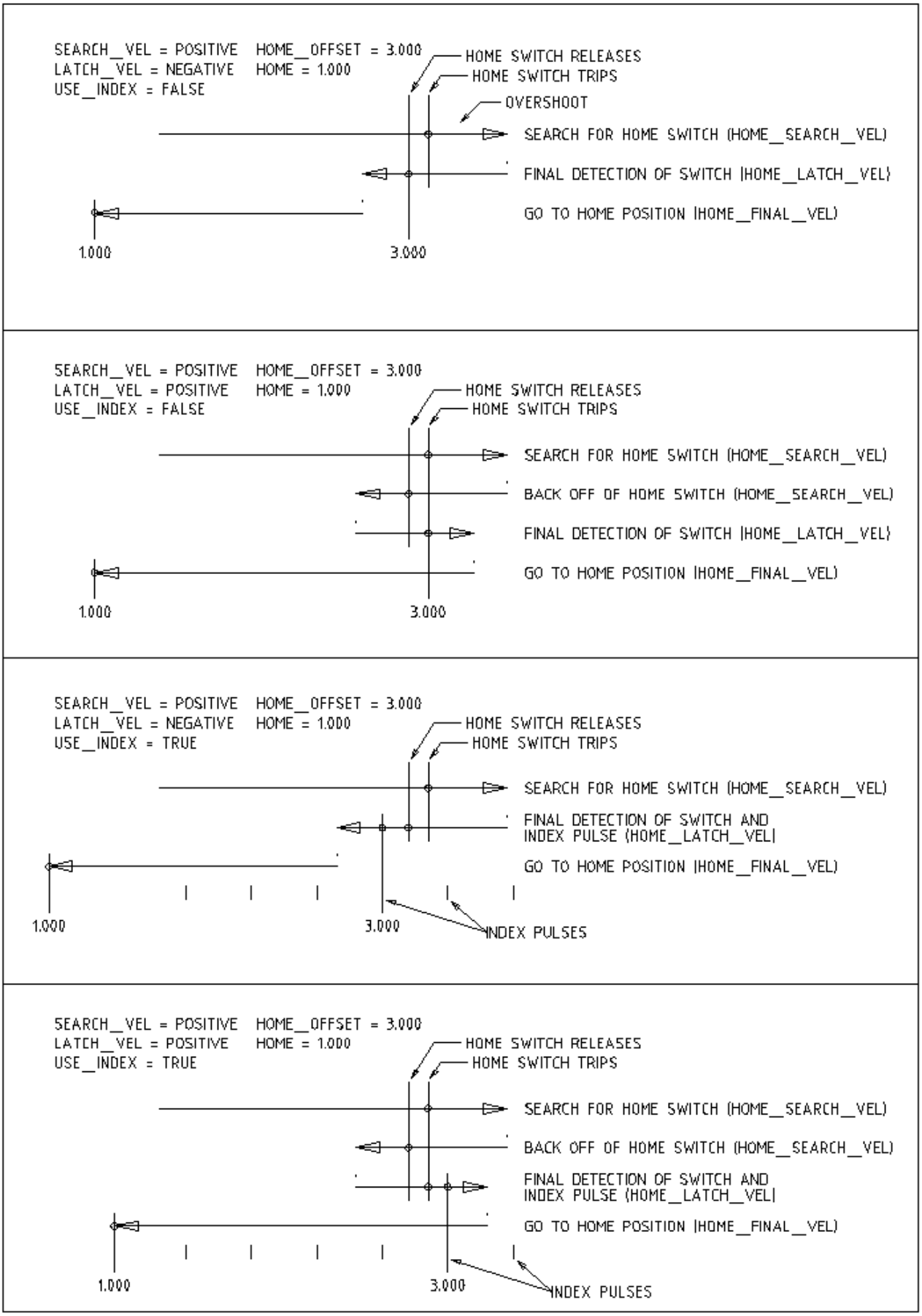


Figura 8.3: Secuencia Homing

8.1.8. Configuración

Lo siguiente determina exactamente cómo se comporta la secuencia homing. Se definen en una sección [JOINT_n] del inifile.

Tipo de Homing	HOME_SEARCH_VEL	HOME_LATCH_VEL	HOME_USE_INDEX
Immediato	0	0	NO
Solo Index	0	no cero	YES
Solo Switch	no cero	no cero	NO
Switch e Index	no cero	no cero	YES

nota
Cualquier otra combinación puede provocar un error.

8.1.8.1. HOME_SEARCH_VEL

Esta variable tiene unidades de unidades máquina por segundo.

El valor por defecto es cero. Un valor de cero hace que LinuxCNC asuma que no hay interruptor home; se omite la etapa de búsqueda de home.

Si HOME_SEARCH_VEL no es cero, entonces LinuxCNC supone que hay un interruptor home. Comienza comprobando si el interruptor home ya está activado. Si lo está, retrocede desde el interruptor a HOME_SEARCH_VEL. La dirección del retroceso es opuesta al signo de HOME_SEARCH_VEL. Despues busca el interruptor home moviéndose en la dirección especificada por el signo de HOME_SEARCH_VEL, a una velocidad determinada por su valor absoluto. Cuando se detecta el interruptor home, la articulación se detendrá tan rápido como le sea posible, pero siempre habrá un exceso. La cantidad de exceso depende de la velocidad. Si es demasiado alto, la articulación podría sobrepasar lo suficiente para alcanzar un interruptor de límite o chocar contra el final de carrera. Por otro lado, si HOME_SEARCH_VEL es demasiado bajo, el recorrido homing puede tomar un largo tiempo.

8.1.8.2. HOME_LATCH_VEL

Esta variable tiene unidades de unidades máquina por segundo.

Especifica la velocidad y dirección que utiliza LinuxCNC cuando realiza su determinación final precisa del interruptor home (si está presente) e ubicación del pulso índice (si está presente). Por lo general, será más lento que la velocidad de búsqueda para maximizar la precisión. Si HOME_SEARCH_VEL y HOME_LATCH_VEL tienen el mismo signo, entonces la fase de enclavamiento se realiza mientras se mueve en la misma dirección que la fase de búsqueda (en ese caso, LinuxCNC primero retrocede desde el interruptor, antes de moverse hacia él nuevamente a la velocidad de enclavamiento). Si HOME_SEARCH_VEL y HOME_LATCH_VEL tienen signos opuestos, la fase de enclavamiento se realiza mientras se mueve en la dirección opuesta a la fase de búsqueda. Eso significa que LinuxCNC enclavara el primer pulso después de que salga del interruptor. Si HOME_SEARCH_VEL es cero (lo que significa que no hay un interruptor home), y este parámetro no es cero, LinuxCNC pasa a buscar el pulso de índice. Si HOME_SEARCH_VEL no es cero y este parámetro es cero, es un error y la operación de búsqueda fallará. El valor predeterminado es cero.

8.1.8.3. HOME_FINAL_VEL

Esta variable tiene unidades de unidades máquina por segundo.

Especifica la velocidad que utiliza LinuxCNC cuando realiza su movimiento desde HOME_OFFSET a la posición HOME. Si falta HOME_FINAL_VEL en el archivo ini, se usa la velocidad máxima para hacer este movimiento. El valor debe ser un número positivo.

8.1.8.4. HOME_IGNORE_LIMITS

Puede contener los valores YES / NO. El valor predeterminado para este parámetro es NO. Este indicador determina si LinuxCNC ignorará la entrada del interruptor de límite para esta articulación mientras hace homing. Esta configuración no ignorará las entradas de límite para otras articulaciones. Si no tiene un interruptor home separado, configúrelo en YES y conecte la señal del interruptor de límite a la entrada del interruptor home en HAL. LinuxCNC ignorará la entrada del interruptor de límite para esta articulación durante el recorrido de homing. Para usar solo una entrada para todo el recorrido de homing y límites, tendrá que bloquear las señales de límite de las articulaciones que no estan haciendo homing en HAL y hacer home en una articulación cada vez.

8.1.8.5. HOME_USE_INDEX

Especifica si hay o no un pulso de índice. Si la bandera es verdadera (HOME_USE_INDEX = YES), LinuxCNC se enclavara en el borde ascendente del pulso índice. Si es falso, LinuxCNC enclavara en el borde ascendente o descendente de el interruptor home (dependiendo de los signos de HOME_SEARCH_VEL y HOME_LATCH_VEL). El valor predeterminado es NO.

nota
HOME_USE_INDEX requiere conexiones en su archivo hal para joint.n.index-enable desde encoder.n.index-enable.

8.1.8.6. HOME_INDEX_NO_ENCODER_RESET

El valor predeterminado es NO. Utilice YES si el codificador utilizado para esta articulación no restablecer su contador cuando se detecta un pulso de índice después de la aserción del pin hal de la articulación index_enable. Aplicable solo si HOME_USE_INDEX = YES.

8.1.8.7. HOME_OFFSET

Esto define la ubicación del punto cero de origen del sistema de coordenadas G53 de la máquina.
Es la distancia (offset), en unidades articulares, desde el origen de la máquina hasta el punto de disparo del interruptor home o pulso índice.
Después de detectar el punto de disparo del interruptor/pulso de índice, LinuxCNC establece la posición de la coordenada a HOME_OFFSET, definiendo así el origen, desde el cual el soft limita las referencias. El valor por defecto es cero.

NOTA: La ubicación del interruptor home, como lo indica la variable HOME_OFFSET, puede estar dentro o fuera de los límites soft. Se compartirán con o dentro de los finales de carrera fisicos.

8.1.8.8. HOME

La posición a la que irá la articulación al finalizar la secuencia homing. Después de detectar el interruptor home o el interruptor y el pulso index (según la configuración) y establecer la coordenada de ese punto en HOME_OFFSET, LinuxCNC se traslada a HOME como el paso final del proceso de búsqueda. El valor por defecto es cero. Tenga en cuenta que incluso si este parámetro es igual que HOME_OFFSET, la articulación sobrepasará ligeramente la posición enclavada mientras para. Por lo tanto, siempre habrá un pequeño movimiento en este momento (a menos que HOME_SEARCH_VEL sea cero, y se omitió toda la etapa de búsqueda/enclavamiento). Este movimiento final se realizará a la velocidad máxima de la articulación a menos que HOME_FINAL_VEL tenga valor.

nota
La distinción entre *HOME_OFFSET* y *HOME* es que *HOME_OFFSET* establece primero la ubicación de origen y la escala en la máquina aplicando el valor *HOME_OFFSET* a la ubicación donde se encontró home, y luego *HOME* dice dónde debe moverse la articulación a esa escala.

8.1.8.9. HOME_IS_SHARED

Si no hay una entrada de interruptor home separado para esta articulación, sino un numero de interruptores momentáneos conectados al mismo pin, establezca este valor en 1 para evitar que el homing se inicie si uno de los conmutadores compartidos está ya está cerrado. Establezca este valor en 0 para permitir el recorrido homing incluso si el interruptor ya está cerrado.

8.1.8.10. HOME_ABSOLUTE_ENCODER

Usado con codificadores absolutos. Cuando se hace una solicitud homing de la articulación, la posición actual se establece en el valor *[JOINT_n]HOME_OFFSET*.

El movimiento final a la posición *[JOINT_n]HOME* es opcional según a la configuración *HOME_ABSOLUTE_ENCODER*:

```
HOME_ABSOLUTE_ENCODER = 0 (predeterminado) la articulación no usa un codificador absoluto
HOME_ABSOLUTE_ENCODER = 1 Codificador absoluto, movimiento final a [JOINT_n]HOME
HOME_ABSOLUTE_ENCODER = 2 Codificador absoluto, NO movimiento final a [JOINT_n]HOME
```

nota
Una configuración HOME_IS_SHARED se ignora.

nota
Una solicitud para volver a colocar la articulación se ignora.

8.1.8.11. HOME_SEQUENCE

Se usa para definir una secuencia homing múltiple **HOME ALL** y aplicar un orden de referencia (p. ej., Z puede no estar homeado si X aún no lo está). Una articulación puede ser homeada después de todas las articulaciones con un valor más bajo (en valor absoluto) de HOME_SEQUENCE ya han sido homeadas y están en HOME_OFFSET. Si dos articulaciones tienen la misma HOME_SEQUENCE, pueden ser homeadas al mismo tiempo.

nota
Si HOME_SEQUENCE no se especifica, la articulacion no será homeada por la secuencia **HOME ALL** (pero si individualmente con comandos homing de la articulación).

El número inicial de HOME_SEQUENCE puede ser 0, 1 (o -1). El valor absoluto de los números de secuencia debe incrementarse en uno; saltar números de secuencia no está permitido. Si un número de secuencia se omite, **HOME ALL** se detendrá al finalizar el ultimo número de secuencia válido.

Los valores de HOME_SEQUENCE **negativos** indican que las articulaciones en la secuencia debería **sincronizar el movimiento final** a [JOINT_n]HOME esperando hasta que todas las articulaciones en la secuencia estén listas. Si alguna articulación tiene un HOME_SEQUENCE **negativo**, todas las articulaciones con el mismo valor absoluto (positivo o negativo) del valor del elemento HOME_SEQUENCE se sincronizarán en el movimiento final.

Un HOME_SEQUENCE **negativo** también se aplica a los comandos para iniciar una sola articulación. Si el valor de HOME_SEQUENCE es **negativo**, todas las articulaciones que tienen el mismo valor absoluto de esa HOME_SEQUENCE serán **homeadas juntas con un movimiento final sincronizado**. Si el valor de HOME_SEQUENCE es cero o positivo, un comando para homear la articulación solo afectará a la articulación especificada.

El jog en modo articulación de las articulaciones que tienen una HOME_SEQUENCE negativa no está permitido. En aplicaciones de pórtico comunes, tal jog puede conducir a la desalineación. Tenga en cuenta que el jog convencional en las coordenadas mundiales siempre están disponibles una vez que la máquina tenga sus home.

Ejemplos para un sistema de 3 articulaciones Dos secuencias (0,1), sin sincronización

```
[JOINT_0]HOME_SEQUENCE = 0
[JOINT_1]HOME_SEQUENCE = 1
[JOINT_2]HOME_SEQUENCE = 1
```

Dos secuencias, articulaciones 1 y 2 sincronizadas

```
[JOINT_0]HOME_SEQUENCE = 0
[JOINT_1]HOME_SEQUENCE = -1
[JOINT_2]HOME_SEQUENCE = -1
```

Con valores mixtos positivos y negativos, las articulaciones 1 y 2 sincronizadas

```
[JOINT_0]HOME_SEQUENCE = 0
[JOINT_1]HOME_SEQUENCE = -1
[JOINT_2]HOME_SEQUENCE = 1
```

Una secuencia, sin sincronización

```
[JOINT_0]HOME_SEQUENCE = 0
[JOINT_1]HOME_SEQUENCE = 0
[JOINT_2]HOME_SEQUENCE = 0
```

Una secuencia, todas las articulaciones sincronizadas

```
[JOINT_0]HOME_SEQUENCE = -1
[JOINT_1]HOME_SEQUENCE = -1
[JOINT_2]HOME_SEQUENCE = -1
```

8.1.8.12. VOLATILE_HOME

Si esta configuración es verdadera, esta articulacion queda sin home cada vez que la máquina pasa al estado OFF. Esto es apropiado para cualquier articulación que no mantenga la posición cuando el accionamiento de la articulación está apagado. Algunas unidades paso a paso, especialmente las unidades de microstep, pueden necesitar esto.

8.1.8.13. LOCKING_INDEXER

Si esta junta es un indexador rotativo con bloqueo, se desbloqueará antes de homing y se bloqueara después.

8.1.8.14. Homing inmediato

Si una articulación no tiene interruptores home o no tiene una posición home lógica, como una articulacion rotativa, y desea que esa articulacion haga home en la posición actual cuando se presiona el botón "Home All" en la GUI Axis, se necesitan las siguientes entradas .ini para esa articulación.

1. HOME_SEARCH_VEL = 0
2. HOME_LATCH_VEL = 0
3. HOME_USE_INDEX = NO
4. HOME igual a HOME_OFFSET
5. HOME_SEQUENCE = 0 (u otro número de secuencia válido)

nota
Los valores predeterminados para HOME_SEARCH_VEL, HOME_LATCH_VEL, HOME_USE_INDEX, HOME y HOME_OFFSET son **cero**, por lo que pueden ser omitidos cuando se solicita la búsqueda inmediata. Un numero HOME_SEQUENCE válido generalmente se debe incluir, ya que omitir una HOME_SEQUENCE elimina la articulación del comportamiento **HOME ALL** como se indicó anteriormente.

8.1.8.15. Inhibición de home

Se proporciona un pin hal (motion.homing-inhibit) para no permitir iniciación de homing para "Home All" y para articulación individual.

Algunos sistemas aprovechan las disposiciones para sincronizar movimientos homing conjuntos finales controlados por elementos negativos [JOINT_N]HOME_SEQUENCE = del archivo ini. Por defecto, las disposiciones de sincronización no permiten jog de **articulación** antes del recorrido homing con el fin de evitar movimientos de **articulación** que podrían desalinear la máquina (pórtico, por ejemplo).

El integrador del sistema puede permitir jog **articular** antes de homing con lógica HAL que cambia los elementos [JOINT_N]HOME_SE
Esta lógica también debe activar el pin **motion.homing-inhibit** para garantizar que el recorrido homing no se inicia inadvertidamente cuando el jog **articular** está habilitado.

Ejemplo: articulaciones sincronizadas 0,1 utilizando una secuencia negativa (-1) para búsqueda sincronizada con un interruptor (allow_jjog) que selecciona una secuencia positiva (1) para jog individual **articular** antes de recorrido homing (código hal parcial):

```
loadrt mux2           names=home_sequence_mux
loadrt conv_float_s32 names=home_sequence_s32
setp home_sequence_mux.in0 -1
setp home_sequence_mux.in1  1
addf home_sequence_mux servo-thread
addf home_sequence_s32 servo-thread
...
net home_seq_float <= home_sequence_mux.out
net home_seq_float => home_sequence_s32.in
net home_seq_s32   <= home_sequence_s32.out
net home_seq_s32   => ini.0.home_sequence
net home_seq_s32   => ini.1.home_sequence
...
# allow_jjog: pin creado por un panel virtual o conmutador hardware
net hsequence_select <= allow_jjog
net hsequence_select => home_sequence_mux.sel
net hsequence_select => motion.homing-inhibit
```

nota
Los pines inihal (como ini.N.home_sequence) no están disponibles hasta que milltask comienza, por lo que la ejecución de los comandos hal anteriores debe ser diferida utilizando un halfile postgui o [APLICACIÓN] APLICACIÓN = script retrasado.

nota
Sincronización en tiempo real del jog articular para múltiples articulaciones requiere conexiones hal adicionales para pines del generador de pulso manual (MPG) (joint.N.enable, joint.N.scale, joint.N.counts).

Un ejemplo de configuración de simulación (gantry_jjog.ini) que demuestra el jog conjunto cuando se utilizan secuencias home negativas se encuentra en el directorio: configs/sim/axis/gantry/.

= Configuración de torno

8.1.9. Plano predeterminado

Cuando el intérprete de LinuxCNC se escribió por primera vez, fue diseñado para fresadoras. Es por eso que el plano predeterminado es XY (G17). Un torno normal solo usa el plano XZ (G18). Para cambiar el plano predeterminado, coloque la siguiente línea en el archivo .ini en la sección RS274NGC.

```
RS274NGC_STARTUP_CODE = G18
```

Lo anterior se puede sobrescribir en un programa de código g, por lo que siempre debe configurar cosas importantes en el preámbulo del archivo de código g.

8.1.10. Configuración INI

Las siguientes configuraciones .ini son necesarias para el modo torno en Axis además de o reemplazando a la configuración normal en el archivo .ini. Estas configuraciones históricas usan cinemática de identidad (trivkins) y *tres* articulaciones (0,1,2) correspondientes a coordenadas x, y, z. En estas configuraciones históricas se requiere la articulacion 1 para el eje Y no utilizado. Las configuraciones de torno simuladas pueden usar estos ajustes históricos. Gmoccapy también usa las configuraciones mencionadas, pero ofrece configuraciones adicionales. Verifique la sección gmoccapy para más detalles.

```
[DISPLAY]
DISPLAY = axis
LATHE = 1
...

[KINS]
KINEMATICS = trivkins
JOINTS = 3

[TRAJ]
COORDINATES = X Z
...

[JOINT_0]
...
[JOINT_2]
...
[AXIS_X]
...
[AXIS_Z]
...
```

Con la incorporación de joint_axes, se puede hacer una configuración más simple con solo las dos articulaciones requeridas especificando trivkins con el parámetro *coordinates*=:

```
[DISPLAY]
DISPLAY = axis
LATHE = 1
...

[KINS]
KINEMATICS = trivkins coordinates=xz
JOINTS = 2

[TRAJ]
COORDINATES = X Z
...

[JOINT_0]
...
[JOINT_1]
```

```
...
[AXIS_X]
...
[AXIS_Z]
...
```

8.2. Archivos HALTCL

El lenguaje halcmd sobresale en la especificación de componentes y conexiones, pero no ofrece capacidades computacionales. Como resultado, los archivos ini están limitados en la claridad y la brevedad que es posible con lenguajes de nivel superior.

La instalación haltcl proporciona un medio para usar scripts tcl y sus características para cálculos, bucles, bifurcaciones, procedimientos, etc. en archivos ini. Para usar estas funcionalidades, se usa el lenguaje tcl y la extensión .tcl para halfiles.

La extensión .tcl es entendida por el script principal (linuxcnc) que procesa archivos ini. Los archivos haltcl se identifican en la sección HAL de los archivos ini. (al igual que los archivos .hal).

Ejemplo

```
[HAL]
HALFILE = archivo_convencional.hal
HALFILE = archivo_tcl.tcl
```

Con el cuidado adecuado, los archivos .hal y .tcl se pueden mezclar.

8.2.1. Compatibilidad

El lenguaje halcmd utilizado en los archivos .hal tiene una sintaxis simple que en realidad es un subconjunto del lenguaje de scripting tcl de propósito general, más poderoso.

8.2.2. Comandos Haltcl

Los archivos haltcl utilizan el lenguaje de scripting tcl aumentado con comandos específicos de la capa de abstracción de hardware (HAL) de LinuxCNC. Los comandos especificos de hal son:

```
addf, alias,
delf, delsig,
getp, gets
ptype,
stype,
help,
linkpp, linkps, linksp, list, loadrt, loadusr, lock,
net, newsig,
save, setp, sets, show, source, start, status, stop,
unalias, unlinkp, unload, unloadrt, unloadusr, unlock,
waitusr
```

Hay dos casos especiales para los comandos *gets* y *list* debido a conflictos con los comandos internos de tcl. Para haltcl, estos comandos deben ir precedidos de la palabra clave *hal*.

```
halcmd   haltcl
------   ------
gets     hal gets
list     hal list
```

8.2.3. Variables Ini Haltcl

Las variables ini son accesibles tanto por halcmd como por haltcl pero con sintaxis diferente

Los archivos ini de LinuxCNC utilizan los especificadores de SECTION e ITEM para identificar elementos de configuración.

```
[SECTION_A]
ITEM1 = value_1
ITEM2 = value_2
...
[SECTION_B]
...
```

Los valores de los archivos ini son accesibles por sustituciones de texto en archivos .hal utilizando la forma.

```
[SECTION]ITEM
```

Los mismos valores de archivo ini son accesibles en archivos .tcl usando la forma de una variable global de matriz tcl.

```
$::SECTION(ITEM)
```

Por ejemplo, un elemento de archivo ini como:

```
[JOINT_0]
MAX_VELOCITY = 4
```

se expresa como [JOINT_0]MAX_VELOCITY en archivos .hal para halcmd
y como $::JOINT_0(MAX_VELOCITY) en archivos .tcl para haltcl

Debido a que los inifiles pueden repetir el mismo ITEM en la misma SECCIÓN varias veces, $::SECTION(ITEM) es en realidad una lista Tcl de cada valor individual.

Cuando hay un solo valor y es un valor simple (todos los valores que son solo letras y números sin espacios en blanco están en este grupo), entonces es posible tratar $::SECTION(ITEM) como si no fuera una lista.

Cuando el valor puede contener caracteres especiales - comillas, corchetes, espacios en blanco incrustados y otros caracteres que tiene un significado especial en Tcl - es necesario distinguir entre la lista de valores y el valor inicial (y posiblemente unico) en la lista.

En Tcl, esto se escribe `[lindex $::SECTION(ITEM) 0]`.

Por ejemplo: dados los siguientes valores ini

```
[HOSTMOT2]
DRIVER=hm2_eth
IPADDR="10.10.10.10"
BOARD=7i92
CONFIG="num_encoders=0 num_pwmgens=0 num_stepgens=6"
```

Y este comando loadrt:

```
loadrt $::HOSTMOT2(DRIVER) board_ip=$::HOSTMOT2(IPADDR) config=$::HOSTMOT2(CONFIG)
```

este sera el comando real que se ejecuta:

```
loadrt hm2_eth board_ip={"10.10.10.10"} config={"num_encoders=0 num_pwmgens=0 num_stepgens ↩
    =6"}
```

Esto falla porque loadrt no reconoce las llaves.

Por tanto, para obtener los valores tal como se ingresaron en el archivo ini, vuelva a escribir la línea loadrt así:

```
loadrt $::HOSTMOT2(DRIVER) board_ip=[lindex $::HOSTMOT2(IPADDR) 0] config=[lindex $:: ↩
    HOSTMOT2(CONFIG) 0]
```

8.2.4. Convertir archivos .hal a archivos .tcl

Los archivos .hal existentes se pueden convertir en archivos .tcl mediante la edición manual para adaptar las diferencias mencionadas anteriormente. El proceso puede ser automatizado con scripts, que los convierten utilizando estas sustituciones:

```
[SECTION]ITEM ---> $::SECTION(ITEM)
gets          ---> hal gets
list          ---> hal list
```

8.2.5. Notas para Haltcl

En haltcl, el argumento de valor para los comandos *sets* y *setp* se trata implícitamente como una expresión en el lenguaje tcl.

Ejemplo

```
# establecer ganancia para convertir grados/segundos a unidades/min para el radio JOINT_0
setp scale.0.gain 6.28/360.0*$::JOINT_0(radius)*60.0
```

No se permite espacios en blanco en la expresión simple; use comillas para ello:

```
setp scale.0.gain "6.28 / 360.0 * $::JOINT_0(radius) * 60.0"
```

En otros contextos, como *loadrt*, debe usar explícitamente el comando tcl expr, *([expr {}])* para expresiones computacionales.

Ejemplo

```
loadrt motion base_period=[expr {500000000/$::TRAJ(MAX_PULSE_RATE)}]
```

8.2.6. Ejemplos de Haltcl

Consideremos el asunto *stepgen headroom*. El software stepgen funciona mejor con una restricción de aceleración que sea "un poco más alta" que la utilizada por el planificador de movimiento. Por lo tanto, cuando se utilizan archivos halcmd, forzamos a los archivos ini a tener un valor calculado manualmente.

```
[JOINT_0]
MAXACCEL = 10.0
STEPGEN_MAXACCEL = 10.5
```

Con haltcl, puede usar los comandos tcl para hacer el cálculo y eliminar el elemento inifile STEPGEN_MAXACCEL por completo.

```
setp stepgen.0.maxaccel $::JOINT_0(MAXACCEL)*1.05
```

Otra característica haltcl es bucle y prueba. Por ejemplo, las configuraciones de muchos simuladores usan los archivos hal "core_sim.hal" o "core_sim9.hal". Estos difieren debido a la necesidad de conectar más o menos ejes. En el siguiente haltcl, el código funcionaría para cualquier combinación de ejes en una máquina con cinematica trivkins.

```
# Crear señales de posición, velocidad y aceleración para cada eje.
set ddt 0
for {set jnum 0} {$jnum < $::KINS(JOINTS)} {incr jnum} {
  # 'list pin' devuelve una lista vacía si el pin no existe
  if {[hal list pin joint.${jnum}.motor-pos-cmd] == {}} {
    continue
  }
  net ${jnum}pos joint.${jnum}.motor-pos-cmd => joint.$axno.motor-pos-fb \
                                             => ddt.$ddt.in
  net ${axis}vel <= ddt.$ddt.out
  incr ddt
  net ${axis}vel => ddt.$ddt.in
```

```
  net ${axis}acc <= ddt.$ddt.out
  incr ddt
}
puts [show sig *vel]
puts [show sig *acc]
```

8.2.7. Haltcl Interactivo

El comando halrun reconoce archivos haltcl. Con la opción -T, haltcl puede ejecutarse de forma interactiva como intérprete de tcl. Esta capacidad es útil para pruebas y para aplicaciones hal independientes.

Ejemplo

```
$ halrun -T haltclfile.tcl
```

8.2.8. Ejemplos Haltcl de la distribución de (sim)

El directorio configs /sim/axis/simtcl incluye un archivo ini que usa un archivo .tcl para demostrar una configuración haltcl junto con el uso de procesamiento twopass. El ejemplo muestra el uso de procedimientos tcl, bucles, uso de comentarios, y salida al terminal.

8.3. Remap Extendiendo el código G

8.3.1. Introducción: Extensión del intérprete RS274NGC mediante remapeado de códigos

8.3.1.1. Definición: Remapeado de Códigos

Por *Remapeado de Códigos* nos referimos a uno de los siguientes casos:

1. Definir la semántica de un código M o G nuevo, es decir, actualmente sin asignar.
2. Redefinir la semántica de un conjunto actualmente limitado de códigos existentes.

8.3.1.2. ¿Por qué querría extender el intérprete RS274NGC?

El conjunto de códigos (M, G, T, S, F) entendido actualmente por el intérprete RS274NGC es fijo y no puede extenderse por opciones de configuración.

En particular, algunos de estos códigos implementan una secuencia fija de pasos para ser ejecutados. Mientras que algunos de estos, como M6, pueden ser moderadamente configurados activando o saltando algunos de estos pasos a través de opciones del archivo .ini, en general el comportamiento es bastante rígido. Si usted está conforme con esta situación, entonces puede ignorar esta sección del manual.

En muchos casos, esto significa que el soporte para configuraciones o máquinas *mas especiales* son engorrosas o imposibles, o requiere recurrir a cambios a nivel del lenguaje *C/C*+\+. Esto último es impopular por buenas razones: el cambio de las características internas requiere un análisis profundo, comprensión de los aspectos internos del intérprete, y además trae su propio conjunto de problemas de soporte. Si bien es posible que ciertos parches podrían encontrar acomodo en la distribución principal de LinuxCNC, el resultado de este enfoque es una mezcolanza de soluciones de casos especiales.

Un buen ejemplo de esta deficiencia es el soporte de cambio de herramienta en LinuxCNC: mientras que los cambiadores de herramientas random están bien soportados, es casi imposible definir razonablemente una configuración para una máquina de cambio de herramienta manual con, por ejemplo, un desplazamiento automático a un interruptor de longitud de herramienta que sea visitado después del cambio, y que los offsets se establezcan en consecuencia. Además, aun existiendo un parche para un cambiador de herramientas rack muy específico, no se ha encontrado la forma de regresar al código base principal.

Sin embargo, muchas de estas cosas pueden solucionarse usando procedimientos O-word en lugar de un código incorporado; siempre que el código incorporado sea insuficiente, llame al procedimiento O-word en su lugar. Si bien es posible, es engorroso; requiere edición de código fuente de programas NGC, reemplazando todas las llamadas al código deficiente por una llamada a un procedimiento O-words.

En su forma más simple, un código reasignado no es mucho más que una llamada espontánea a un procedimiento O-word. Esto sucede en el transfondo. El procedimiento es visible en el nivel de configuración, pero no en el nivel de programa NGC.

En general, el comportamiento de un código reasignado se puede definir de las siguientes maneras:

- usted define una subrutina O-word que implementa el comportamiento deseado.
- alternativamente, puede emplear una función Python que amplíe el comportamiento del intérprete.

Como unir las cosas Los códigos M y G, y las llamadas de subrutinas O-words tienen una sintaxis bastante diferente.

Los procedimientos O-word, por ejemplo, toman parámetros posicionales con una sintaxis específica tal como:

```
o<test> call [1.234] [4.65]
```

mientras que los códigos M o G normalmente toman parámetros de *palabra*, requerida u opcional. Por ejemplo, G76 (roscado) requiere las palabras P, Z, I, J y K, y opcionalmente toma las palabras R, Q, H, E y L.

Así que no es suficiente decir *siempre que encuentre el código X, por favor llamar al procedimiento Y*; se necesita al menos alguna comprobación y conversión de parámetros. Esto requiere un cierto *código de union* entre el nuevo código y su correspondiente procedimiento NGC que se debera ejecutar antes de pasar el control al procedimiento NGC.

Este código de union es imposible de escribir como un procedimiento O-word ya que el lenguaje RS274NGC carece de capacidades introspectivas y acceso a las estructuras internas de datos del intérprete para lograr el efecto requerido. De nuevo, hacer el código de union en *C/C*+\+ sería una solución inflexible y por lo tanto insatisfactoria.

Como Encaja Python Embebido Para hacer solucionable una situación compleja y que una situación simple sea facil, el problema del codigo de union se trata de la siguiente manera:

- para situaciones simples, un procedimiento de union incorporado (`argspec`) cubre la mayoría de los requisitos comunes de paso de parámetros.
- para el remapeado de T, M6, M61, S, F hay un algo de codigo de union Python estándar que debería cubrir la mayoría de las situaciones, ver union estándar
- para situaciones más complejas, puede escribir su propio codigo de union Python para implementar un nuevo comportamiento

Las funciones Python embebidas en el intérprete comenzaron como un codigo de union, pero resultaron muy útiles mas alla de eso. Los usuarios familiarizados con Python probablemente encontrará más fácil escribir códigos reasignados que unir procedimientos O-word, etc, en Python puro, sin recurrir en absoluto al algo engorroso lenguaje RS274NGC.

Unas Palabras sobre Python Embebido Muchas personas están familiarizadas con *extender* el intérprete de Python mediante módulos *C/C++*, y esto se usa mucho en LinuxCNC para acceder a Task e interioridades de HAL y del intérprete mediante scripts de Python. *Extender Python* básicamente significa que su script de Python se ejecuta *de la forma estandar*, y puede acceder a código que no es Python importando y usando módulos de extensión escritos en *C/C*+\+. Ejemplos de esto son los módulos de LinuxCNC `hal`,` gcode` y `emc`.

Python Embebido es un poco diferente y menos conocido; el programa principal está escrito en *C/C++* y puede usar Python como una subrutina. Este es un poderoso mecanismo de extensión y la base para las *extensiones de scripts* encontradas en muchos programas conocidos. El código Python Embebido puede acceder a las variables *C/C*+\+ y funciona a través de un método de extensión de módulo similar.

8.3.2. Comenzando

La definición de un código implica los siguientes pasos:

- Elegir un código - usar un código no asignado, o redefinir un código existente
- Decidir cómo se manejan los parámetros.
- Decidir si se manipulan los resultados y cómo.
- Decidir sobre la secuencia de ejecución.

8.3.2.1. Escogiendo un código

Tenga en cuenta que actualmente solo se pueden redefinir algunos códigos existentes, mientras que hay muchos códigos *libres* que pueden estar disponibles para remapeado. Al desarrollar un código existente redefinido, podría ser una buena idea comenzar con un código G o M sin asignar, de modo que se pueda emplear tanto un comportamiento existente como uno nuevo. Cuando haya terminado, redefina el código existente para utilizar su configuración de remapeado.

- el conjunto actual de códigos M no utilizados, disponibles para definición de usuario, se puede encontrar aquí,
- Se enumeran los códigos G no asignados aquí.
- Los códigos existentes que pueden ser reasignados están listados aquí.

8.3.2.2. Manejo de parámetros

Asumamos que el nuevo código será definido por un procedimiento NGC y necesita algunos parámetros, unos necesarios y otros opcionales. Tenemos las siguientes opciones para alimentar al procedimiento con sus valores:

1. extraer palabras del bloque actual y pasarlas al procedimiento como parámetros (como `X22.34` o `P47`)
2. refiriéndose a las variables del archivo ini
3. refiriéndose a variables globales (como `#2200 =47.11` o `#<_global_param> =315.2`

El primer método se prefiere para parámetros de naturaleza dinámica, como posiciones. Es necesario definir qué palabras en el bloque actual tienen algun significado para su nuevo código, y especificar cómo se pasan al procedimiento NGC. Una forma fácil es usar la declaración argspec. Un prologo personalizado podría proporcionar mejores mensajes de error.

Para referirse a la información de configuración de su máquina, es más útil usar las variables de archivo ini; por ejemplo, una posición fija como la posición del sensor de longitud de la herramienta. La ventaja de este método es que los parámetros son fijos en su configuración, independientemente del archivo NGC en ejecucion.

Siempre es posible hacer referencia a variables globales, pero son fáciles de pasar por alto.

Tenga en cuenta que hay una cantidad limitada de palabras que pueden usarse como parámetros, por lo que podría tener que recurrir al segundo y tercer método si se necesitan muchos parámetros.

8.3.2.3. Manejo de resultados

Su nuevo código podría tener éxito o fallar, por ejemplo, si se pasa una combinación de parámetros inválida. O puede elegir "ejecutar" el procedimiento y descartar los resultados, en cuyo caso no hay mucho trabajo por hacer.

Los manejadores de epilogo ayudan en el procesamiento de los resultados de los procedimientos de remapeado; consulte la sección de referencia.

8.3.2.4. Secuenciación de ejecución

Las palabras de código G ejecutables se clasifican en grupos modales, que también definen su comportamiento relativo de ejecución.

Si un bloque de código G contiene varias palabras ejecutables en una línea, estas palabras se ejecutan en un orden de ejecución, no en el orden en que aparecen en bloque.

Cuando define un nuevo código ejecutable, el intérprete todavía no saber dónde encaja su código en este esquema. Por lo tanto, debe elegir un grupo modal apropiado para su código.

8.3.2.5. Un ejemplo mínimo de código remapeado

Para darle una idea de cómo encajan las piezas, exploremos una definición de código bastante minimalista pero completa. Elegimos un código M no asignado y agregamos la siguiente opción al archivo ini:

```
[RS274NGC]
REMAP=M400  modalgroup=10 argspec=Pq ngc=myprocedure
```

Esto significa, en pocas palabras:

- El código `M400` toma un parámetro requerido `P` y otro opcional `Q`. Otras palabras en el bloque actual son ignoradas con respecto al código `M400`. Si la palabra `P` no está presente, la ejecución falla con un error.
- cuando se encuentra un código `M400`, se ejecuta `myprocedure.ngc` junto con otros códigos M del grupo modal 10 según el orden de ejecución.
- el valor de `P` y `Q` están disponibles en el procedimiento como parámetros nombrados locales. Pueden denominarse `#<P>` y `#<Q>`. El procedimiento puede probar si la palabra `Q` estaba presente con la función incorporada EXISTS.

Se espera que el archivo `myprocedure.ngc` exista en el directorio `[DISPLAY]NC_FILES` o en `[RS274NGC]SUBROUTINE_PATH`.

Una discusión detallada de los parámetros de REMAP se encuentra en la sección de referencia que sigue.

8.3.3. Configuración del remapeo

8.3.3.1. La sentencia REMAP

Para reasignar un código, defínalo usando la opción `REMAP` en la sección `RS274NG` de su archivo ini. Use una línea `REMAP` por cada código reasignado.

La sintaxis de `REMAP` es:

REMAP=<*code*> <*options*>
: donde <*code*> puede ser `T`,` M6`, `M61`,` S` o `F` (códigos existentes) o cualquiera de los códigos M o códigos G sin asignar.

Es un error omitir el parámetro <*code*>.

Las opciones de la instrucción REMAP están separadas por espacios en blanco. Las opciones son pares de palabra clave-valor y actualmente son:

modalgroup=<*modal group*>

Códigos G
: el único grupo modal actualmente soportado es 1, que también es el valor predeterminado si no se da ningún grupo. Grupo 1 significa *ejecutar junto con otros códigos G*.

Códigos M

Los grupos modales soportados actualmente son: 5,6,7,8,9,10. Si no se da ningún grupo modal, el valor predeterminado es 10 (*ejecutar después de todas las otras palabras en el bloque*).

```
T, S, F; Para estos el grupo modal es fijo y cualquier
opción `modalgroup=` se ignora.
```

argspec=<*argspec*>

Ver descripción de opciones de parámetros argspec. Opcional.

ngc=<*nombre_báse_ngc*>

Nombre base de un nombre de archivo de subrutina O-word. No especifique la extensión .ngc. Se busca en los directorios especificados en el directorio dado en `[DISPLAY]PROGRAM_PREFIX`, y luego en `[RS274NGC]SUBROUTINE_PATH`. Es mutuamente excluyente con `python=`. Es un error omitir tanto `ngc=` como `python=`.

python=<*nombre de la función de Python*>

En lugar de llamar a un procedimiento ngc O-word, llame a una función Python. Se espera que la función se defina en el módulo `module_basename.oword`. Mutuamente excluyente con `ngc=`.

prolog=<*nombre de la función de Python*>

Antes de ejecutar un procedimiento ngc, llame a esta función Python. Se espera que la función se defina en el módulo `module_basename.remap`. Opcional.

epilog=<*nombre de la función de Python*>

Después de ejecutar un procedimiento ngc, llame a esta función Python. Se espera que la función se defina en el módulo `module_basename.remap`. Opcional.

Las opciones `python`,` prolog` y `epilog` requieren que el plugin de intérprete Python sea configurado, y las funciones apropiadas de Python se definirán allí para que puedan ser referidas con estas opciones.

La sintaxis para definir un nuevo código y redefinir un código existente es idéntica.

8.3.3.2. Combinaciones útiles de opciones de REMAP

Tenga en cuenta que si bien son posibles muchas combinaciones de opciones argspec, no todas ellas tienen sentido. Las siguientes combinaciones son expresiones útiles:

argspec=<*words*> ngc=<*procname*> modalgroup=<*group*>

Forma recomendada de llamar a un procedimiento NGC con conversión estándar de parámetro argspec. Se utiliza si argspec es suficientemente bueno para nuestro proposito. Tenga en cuenta que no es suficientemente bueno para volver a asignar los códigos de cambio de herramientas Tx y M6/M61.

prolog=<*pythonprolog*> ngc=<*procname*> epilog=<*pythonepilog*> modalgroup=<*group*>

Llama a una función de prologo de Python para realizar cualquier paso preliminar, luego llama al procedimiento NGC. Cuando ha terminado, llama a la función de epilogo de Python para hacer cualquier limpieza o trabajo de extracción de resultados que no pueda ser manejado en código G. Es la forma más flexible de volver a asignar un código a un procedimiento NGC, ya que casi todas las variables, y algunas funciones, internas del intérprete se pueden acceder desde los manipuladores de prologo y epilogo. Pero tambien es la forma mas propensa a errores propios.

python=<*pythonfunction*> modalgroup=<*group*>

Llama directamente a una función de Python sin ninguna conversión de argumentos. La forma más poderosa de reasignar un código e ir directamente a python. Use esto si no necesita un procedimiento NGC, o NGC se usa accidentalmente.

argspec=<*words*> python=<*pythonfunction*> modalgroup=<*group*>

Convierte las palabras argspec y las pása a una función Python como argumento diccionario de palabras clave. Úselo para no tener que investigar las palabras pasadas en el bloque por usted mismo.

Tenga en cuenta que si todo lo que quiere lograr es llamar a algún código Python desde código G, hay una forma algo más fácil de llamar a funciones de Python como procedimientos O-word.

8.3.3.3. El parámetro argspec

La especificación del argumento (palabra clave `argspec`) describe las palabras requeridas y opcionales a pasar a un procedimiento ngc, así como las condiciones previas opcionales para que ese código se ejecute.

Un argspec consta de 0 o más caracteres de la clase `[@A-KMNP-Za-kmnp-z^>]` . Puede estar vacío (como `argspec=`).

Un argumento argspec vacío, o ningún argumento argspec en absoluto, implica que el código remapeado no recibe ningún parámetro del bloque. Se ignora cualquier parámetro extra presente.

Tenga en cuenta que las reglas RS274NGC se aplican todavía; por ejemplo, puede usar palabras de eje (por ejemplo, X, Y, Z) solo en el contexto de un código G.

ABCDEFGHIJKMPQRSTUVWXYZ
: Define un parámetro de palabra requerido; una letra mayúscula especifica que la palabra correspondiente **debe** estar presente en el bloque actual. El valor de la palabra será pasado como un parámetro con nombre local con un nombre correspondiente. Si el caracter `@` esta presente en argspec, se pasará como parámetro posicional; ver más abajo.

abcdefghijkmpqrstuvwxyz
: Define un parámetro de palabra opcional: una letra minúscula especifica que la palabra correspondiente **puede** estar presente en el bloque actual. Si la palabra está presente, el valor de la palabra será pasado como un parámetro con nombre local. Si el caracter `@` esta presente en argspec, se pasará como parámetro posicional; ver más abajo.

@
: El `@` (signo -at-) le dice a argspec que pase palabras como parámetros posicionales, en el orden definido después de la opción `@`. Tenga en cuenta que cuando se utiliza el paso de parámetros posicionales, un procedimiento no puede determinar si una palabra estaba presente o no; vea un ejemplo a continuación.

---

sugerencia
esto ayuda a empaquetar los procedimientos existentes de NGC como códigos remapeados. Los procedimientos existentes esperan parámetros posicionales. Con la opción `@`, puede evitar reescribirlos para referirse a parámetros con nombre locales.

---

^
: El carácter ^ (caret) especifica que la velocidad actual del husillo debe ser mayor que cero (husillo en marcha), de lo contrario, el código falla con un mensaje de error apropiado.

>
: El carácter > (mayor que) especifica que la velocidad de alimentacion actual debe ser mayor que cero, de lo contrario el código falla con un mensaje de error apropiado.

n
: El carácter `n` especifica que se pase el número de línea actual al parámetro nombrado local `n`.

De forma predeterminada, los parámetros se pasan con nombre local a un procedimiento NGC. Estos parámetros locales aparecen como *ya establecidos* cuando el procedimiento comienza a ejecutarse, lo que es diferente de la semántica existente (las variables locales comienzan con el valor 0.0 y debe ser asignado un valor explícitamente).

Los parámetros de palabra opcionales se pueden probar para detectar su presencia mediante `EXISTS(#<word>)`.

Ejemplo para el paso de parámetros con nombre a procedimientos NGC Supongamos que el código se define como

```
REMAP=M400 modalgroup=10 argspec=Pq ngc=m400
```

y `m400.ngc` es como sigue:

```
o<m400>sub
(Se requiere P ya que está en mayúsculas en argspec)
(debug, palabra P=#<P>)
(q es opcional ya que está en minúscula en argspec. Use de la siguiente manera: )
```

```
o100 if [EXISTS[#<q>]]
    (debug, palabra asignada Q=#<q>)
o100 endif
o<m400> endsub
M2
```

- ejecutando `M400` fallará con el mensaje `M400 definido por el usuario:falta:P`
- la ejecución de `M400 P123` mostrará` palabra P=123.000000`
- la ejecución de `M400 P123 Q456` mostrará` palabra P=123.000000` y `palabra asignada Q=456.000000`

Ejemplo para pasar parámetros posicionales a procedimientos NGC Supongamos que el código se define como

`REMAP=M410 modalgroup=10 argspec=@PQr ngc=m410`

y `m410.ngc` es como sigue:

```
o<m410> sub
(debug, [1]=#1 [2]=#2 [3]=#3)
o<m410> endsub
M2
```

- la ejecución de `M410 P10` mostrará` m410.ngc: [1]=10.000000 [2]=0.000000`
- la ejecución de `M410 P10 Q20` mostrará` m410.ngc: [1]=10.000000 [2]=20.000000`

NB: se pierde la capacidad de distinguir más de una palabra de parámetro opcional, y no se puede saber si un parámetro opcional estaba presente pero tenía el valor 0, o no estaba presente en absoluto.

Ejemplo simple para pasar un parámetro con nombre a una función de Python Es posible definir nuevos códigos *sin* procedimiento NGC. Esto es un primer ejemplo simple; uno más complejo se puede encontrar en la siguiente sección.

Supongamos que el código se define como

`REMAP=G88.6 modalgroup=1 argspec=XYZp python=g886`

Esto le indica al intérprete que ejecute la función Python `g886` en el modulo `module_basename.remap`, que podría ser así:

```
from interpreter import INTERP_OK
from emccanon import MESSAGE

def g886(self, **words):
    for key in words:
        MESSAGE("word '%s' = %f" % (key, words[key]))
    if words.has_key('p'):
        MESSAGE("la palabra P estaba presente")
    MESSAGE("comentario en esta línea: '%s'" % (self.blocks[self.remap_level].comment))
    return INTERP_OK
```

Pruebe esto con: g88.6 x1 y2 z3 g88.6 x1 y2 z3 p33 (un comentario aquí)

Notará la introducción gradual al entorno de Python incrustado. - vea esto para más detalles. Tenga en cuenta que con las funciones de remapeado Python, no tiene sentido tener funciones de prologo o epilogo ya que está ejecutando una función Python en primer lugar.

Ejemplo avanzado: códigos remapeados en Python puro Los módulos `interpreter` y` emccanon` exponen la mayoría de interioridades del intérprete y algunos de Canon; muchas cosas que hasta ahora requerían codificación en *C/C*+\+ ahora se puede hacer en Python.

El siguiente ejemplo se basa en el script `nc_files/involute.py` - pero enlatado como un código G con algunos parámetros de extracción y comprobación. Esto también demuestra la llamada al intérprete de forma recursiva (consulte `self.execute()`).

Suponiendo una definición como esta (NB: esto no usa argspec):

```
REMAP=G88.1 modalgroup=1 py=involute
```

La función `involute` en` python/remap.py` que aparece a continuación hace toda la extracción de palabras directamente del bloque actual. Tenga en cuenta que los errores del intérprete pueden ser traducidos a excepciones de Python. Recuerde que esto es *readahead time* - los errores de tiempo de ejecución no pueden ser atrapados de esta manera.

```
import sys
import traceback
from math import sin,cos

from interpreter import *
from emccanon import MESSAGE
from util import lineno, call_pydevd
# genera InterpreterException si fallan execute() o read()
throw_exceptions = 1

def involute(self, **words):
    """ función de remapeado con acceso directo a las funciones internas del intérprete """

    if self.debugmask & 0x20000000: call_pydevd() # USER2 debug flag

    if equal(self.feed_rate,0.0):
        return "se requiere alimentacion > 0"

    if equal(self.speed,0.0):
        return "se requiere velocidad de husillo > 0"

    plunge = 0.1 #si se dio la palabra Z, descender - con alimentación reducida

    # inspeccionar bloque de control para palabras relevantes
    c = self.blocks[self.remap_level]
    x0 = c.x_number if c.x_flag else 0
    y0 = c.y_number if c.y_flag else 0
    a  = c.p_number if c.p_flag else 10
    old_z = self.current_z

    if self.debugmask & 0x10000000:
        print "x0=%f y0=%f a=%f old_z=%f" % (x0,y0,a,old_z)

    try:
        #self.execute("G3456")  # generaría una excepción InterpreterException
        self.execute("G21",lineno())
        self.execute("G64 P0.001",lineno())
        self.execute("G0 X%f Y%f" % (x0,y0),lineno())

        if c.z_flag:
            feed = self.feed_rate
            self.execute("F%f G1 Z%f" % (feed * plunge, c.z_number),lineno())
            self.execute("F%f" % (feed),lineno())

        for i in range(100):
            t = i/10.
            x = x0 + a * (cos(t) + t * sin(t))
            y = y0 + a * (sin(t) - t * cos(t))
            self.execute("G1 X%f Y%f" % (x,y),lineno())

        if c.z_flag: # retrae a la altura inicial
            self.execute("G0 Z%f" % (old_z),lineno())

    except InterpreterException,e:
        msg = "%d: '%s' - %s" % (e.line_number,e.line_text, e.error_message)
```

```
  return msg

    return INTERP_OK
```

Los ejemplos descritos hasta ahora se pueden encontrar en *configs/sim/axis/remap/getting-started* con configuraciones completas.

8.3.4. Actualización de una configuración existente para remapeado

Los requisitos mínimos para usar las declaraciones `REMAP` son las siguientes:

- el plug Python debe activarse especificando `[PYTHON]TOPLEVEL=<path-to-toplevel-script>` en el archivo ini.
- el script de nivel superior debe importar el módulo `remap`, que puede estar inicialmente vacío, pero la importación debe estar en su lugar.
- El intérprete de Python necesita encontrar el módulo remap.py, por lo que la ruta al directorio donde residen los módulos de Python debe estar añadida con `[PYTHON]PATH_APPEND=<path-to-your-local-Python-directory>`
- Recomendado: importe los manejadores `stdglue` en el módulo `remap`. En este caso, Python también necesita encontrar `stdglue.py` - simplemente lo copiamos desde la distribución para que pueda realizar cambios locales como sea necesario. Dependiendo de su instalación, la ruta a `stdglue.py` podría variar.

Asumiendo que sus configuraciones residen bajo `/home/user/xxx` y el archivo ini es `/home/user/xxx/xxx.ini`, ejecute los siguientes comandos.

```
$ cd /home/user/xxx
$ mkdir python
$ cd python
$ cp /usr/share/linuxcnc/ncfiles/remap_lib/python-stdglue/stdglue.py .
$ echo 'from stdglue import *' >remap.py
$ echo 'import remap' >toplevel.py
```

Ahora edite `/home/user/xxx/xxx.ini` y agregue lo siguiente:

```
[PYTHON]
TOPLEVEL=/home/user/xxx/python/toplevel.py
PATH_APPEND=/home/user/xxx/python
```

Ahora verifique que LinuxCNC no presenta ningún mensaje de error; desde un ventana de terminal ejecutar:

```
$ cd /home/user/xxx
$ linuxcnc xxx.ini
```

8.3.5. Remapeo de códigos relacionados con el cambio de herramienta: T, M6, M61

8.3.5.1. Descripción general

Si no está familiarizado con las partes internas de LinuxCNC, primero lea la sección Cómo trabaja el cambio de herramienta actualmente (pesado, pero necesario).

Tenga en cuenta que al volver a asignar un código existente, deshabilitamos completamente la funcionalidad incorporada de estos códigos del intérprete.

Así que nuestro código reasignado tendrá que hacer algo más que generar algunos comandos para mover la máquina como nos gustaria; también tendrá que replicar los pasos de esta secuencia que son necesarios para mantener al intérprete y a Task sin problemas.

Sin embargo, esto **no** afecta el procesamiento de comandos relacionados con el cambio de herramienta en Task e iocontrol. Esto significa que cuando ejecutemos el paso 6b esto aún causará que iocontrol haga sus cosas.

Decisiones, decisiones:

- ¿Queremos usar un procedimiento O-word o hacerlo todo en código Python?
- ¿Es la secuencia HAL de iocontrol (preparación de herramienta/herramienta preparada y pines de cambio de herramienta/herramienta cambiada) suficientemente buenos o necesitamos un tipo diferente de interacción HAL para nuestro cambiador de herramientas (por ejemplo: más pines HAL involucrados con una secuencia de interacción diferente)?

Dependiendo de la respuesta, tenemos cuatro escenarios diferentes:

- Cuando se usa un procedimiento O-word, necesitamos funciones de prologo y epilogo.
- Si usa solo código Python y ningún procedimiento O-word, una función Python es suficiente.
- cuando se utilizan los pines de iocontrol, nuestro procedimiento O-word o el código Python contendrá movimientos en su mayoría.
- cuando necesitamos una interacción más compleja que la ofrecida por iocontrol, necesitamos definir completamente nuestra propia interacción, usando los pines `motion.digital*` y `motion.analog*`, y esencialmente ignorar los pines de iocontrol puenteandolos

NOTA: Si odias los procedimientos O-word y te encanta Python, eres libre de hacerlo todo en Python, en cuyo caso solo tendrías una especificacion `python=<function>` en la sentencia REMAP. Pero suponiendo que la mayoría de la gente estaría interesada en utilizar procedimientos O-word porque están más familiarizados con eso, lo haremos asi como el primer ejemplo.

El enfoque general para nuestro primer ejemplo será:

1. Por flexibilidad, nos gustaría hacer todo lo posible con el código G en un procedimiento de palabra O. Eso incluye toda la interacción HAL que normalmente sería manejada por iocontrol, porque preferiríamos hacer cosas inteligentes con movimientos, sondas, pines I/O HAL y demas.
2. intentaremos minimizar el código de Python en la medida necesaria para mantener sin problemas al intérprete, y hacer que task haga realmente algo. Eso entrará en las funciones de Python `prolog` y` epilog`.

8.3.5.2. Entender el rol de iocontrol con códigos de cambio de herramienta remapeados

iocontrol proporciona dos secuencias de interacción HAL que podemos utilizar o no:

- cuando el mensaje NML puesto en cola por un comando canonico SELECT_POCKET() es ejecutado, se desencadena la secuencia HAL "preparar herramienta y esperar que herramienta preparada pase a alto" en iocontrol, además de ajustar los pines XXXX
- cuando el mensaje NML puesto en cola por el comando canonico CHANGE_TOOL() es ejecutado, esto activa la secuencia HAL "cambiar de herramienta y esperar que herramienta cambiada pase a alto" en iocontrol, además de ajustar de los pines XXXX

Lo que debe decidir es si las secuencias HAL de iocontrol existentes son suficientes para manejar su cambiador. Tal vez necesite una secuencia de interacción diferente - por ejemplo, más pines HAL, o tal vez interacción más compleja. Dependiendo de la respuesta, podríamos seguir utilizando las secuencias HAL de iocontrol, o definir las nuestras propias.

Para documentarlo mejor, deshabilitaremos estas secuencias de iocontrol y ejecutaremos las nuestras - el resultado se parecera a la interacción existente, pero ahora tenemos control completo sobre ellas porque se ejecutan en nuestro propio procedimiento O-word.

Para ello, lo que haremos sera usar `motion.digital-*` y `motion.analog-*` y los comandos asociados `M62` ..` M68` para hacer nuestra propia interacción HAL en nuestro procedimiento O-word, y aquellos que efectivamente reemplacen las secuencias *tool-prepare/tool-ready* y *tool-change/tool-changed* de iocontrol. Así que vamos a definir nuestros pines, reemplazando funcionalmente los pines iocontrol existentes, y seguir adelante y hacer un bucle de interacciones iocontrol. Usaremos la siguiente correspondencia en nuestro ejemplo:

Correspondencia de pines iocontrol en los ejemplos.

pin iocontrol.0	pin motion
tool-prepare	digital-out-00
tool-prepared	digital-in-00
tool-change	digital-out-01
tool-changed	digital-in-01
tool-prep-number	analog-out-00
tool-prep-pocket	analog-out-01
tool-number	analog-out-02

Supongamos que desea redefinir el comando M6 y reemplazarlo por un procedimiento O-word pero, aparte de eso, las demas cosas deberían continuar trabajando.

Por tanto, lo que nuestro procedimiento O-word haría es reemplazar los pasos descritos aquí. Mirando estos pasos encontrará que el código NGC puede usarse para la mayoría de ellos, pero no todos. Así que las cosas que NGC no puede manejar se harán en las funciones prolog y epilog de Python.

8.3.5.3. Especificando el reemplazo M6

Para transmitir la idea, simplemente reemplazamos la semántica M6 incorporada con la nuestra propia. Una vez que funcione, puede seguir adelante y colocar cualquier acción que quiera encajar en el procedimiento O-word.

Al revisar los pasos, encontramos:

1. Compruebe si el comando T ya se ejecutó - **ejecutar en el prologo Python**
2. verificar si la compensación del cortador está activa - **ejecutar en el prologo Python**
3. detener el husillo si es necesario - **se puede hacer en NGC**
4. pinola arriba - **se puede hacer en NGC**
5. si se estableció TOOL_CHANGE_AT_G30:
   a. mueva los indexadores A, B y C si corresponde - **se puede hacer en NGC**
   b. generar movimiento rápido a la posición G30 - **se puede hacer en NGC**
6. enviar un comando canonico CHANGE_TOOL a Task - **ejecutar en el epilogo Python**
7. configurar los parámetros números 5400-5413 de acuerdo con la nueva herramienta - **ejecutar en el epilogo Python**
8. enviar una señal a Task para que deje de llamar al intérprete para lectura antes de completar el cambio de herramienta - **ejecutar en epilogo Python**

Así que necesitamos un prologo y un epilogo. Asumamos que, en nuestro archivo ini, el remapeo M6 tiene el siguiente aspecto:

```
REMAP=M6   modalgroup=6  prolog=change_prolog ngc=change epilog=change_epilog
```

Decidimos pasar algunas variables al procedimiento de remapeado que se puede inspeccionar y cambiar allí, o utilizarlas en un mensaje. Esos son: `tool_in_spindle`, `selected_tool` (números de herramientas) y sus respectivas ranuras `current_pocket` y` selected_pocket`. Con ello, el prologo que cubre los pasos 1 y 2 se vería así:

```
def change_prolog(self, **words):
    try:
        if self.selected_pocket < 0:
            return "M6: ninguna herramienta preparada"

        if self.cutter_comp_side:
            return "No se pueden cambiar herramienta con compensación de radio de corte ↩
                activada"
```

```
        self.params["tool_in_spindle"] = self.current_tool
        self.params["selected_tool"] = self.selected_tool
        self.params["current_pocket"] = self.current_pocket
        self.params["selected_pocket"] = self.selected_pocket
        return INTERP_OK
    except Exception, e:
        return "M6/change_prolog: %s" % (e)
```

Encontrará que la mayoría de las funciones de prologo son muy similares: primero probar que todas las condiciones previas para ejecutar el código se cumplen. Luego preparar el entorno - inyectar variables y/o hacer cualquier paso de procesamiento preparatorio que no se pueden hacer fácilmente en código NGC; luego pasar al procedimiento NGC devolviendo INTERP_OK.

Nuestra primera iteración de procedimiento O-word es poco interesante; solo verifica que tengamos los parámetros correctos y señalemos el éxito devolviendo un valor positivo; los pasos 3-5 eventualmente serían cubiertos aquí (ver aquí para las variables referentes a la configuración del archivo ini):

```
O<change> sub
(debug, cambio: current_tool=#<current_tool>)
(debug, cambio: selected_pocket=#<selected_pocket>)
;
; inserte cualquier código g que vea adecuado aquí, por ejemplo:
; G0  #<_ini[setup]tc_x>  #<_ini[setup]tc_y>  #<_ini[setup]tc_z>
;
O<change> endsub [1]
m2
```

Asumiendo el éxito de `change.ngc`, necesitamos limpiar los pasos 6-8:

```
def change_epilog(self, **words):
    try:
        if self.return_value > 0.0:
            # cambio
            self.selected_pocket =  int(self.params["selected_pocket"])
            emccanon.CHANGE_TOOL(self.selected_pocket)
            # causar sync()
            self.tool_change_flag = True
            self.set_tool_parameters()
            return INTERP_OK
        else:
            return "M6 abortado (código de retorno %.1f)" % (self.return_value)

    except Exception, e:
        return "M6/change_epilog: %s" % (e)
```

Este reemplazo M6 es compatible con el código incorporado, excepto los pasos 3-5, que deben completarse con su código NGC.

Una vez más, la mayoría de los epilogos tienen un esquema común: primero, determinar si las cosas salieron bien en el procedimiento de remapeado, luego hacer cualquier accion de confirmación y limpieza que no se pueden hacer en código NGC.

8.3.5.4. Configurando iocontrol con un M6 remapeado

Tenga en cuenta que la secuencia de operaciones ha cambiado: hacemos todo lo requerido en el procedimiento O-word - incluyendo cualquier configuración/lectura de pin HAL para activar un cambiador, y para reconocer un cambio de herramienta - probablemente con pines IO `motion.digital-*` y `motion-analog-*`. Cuando finalmente ejecutamos el comando `CHANGE_TOOL()`, todos los movimientos y las interacciones HAL ya están completos.

Normalmente, solo ahora iocontrol haría su trabajo como se describe aquí. Sin embargo, no necesitamos mover los pines HAL más - todo lo que queda por hacer con iocontrol es aceptar que hemos terminado con preparado y cambiado.

Esto significa que los pines iocontrol correspondientes no tienen ninguna función más. Por lo tanto, configuramos iocontrol para reconocer inmediatamente un cambio, de esta manera:

```
# puenteo de señales de cambio al reasignar M6
net tool-change-loop iocontrol.0.tool-change iocontrol.0.tool-changed
```

Si por alguna razón desea remapear `Tx` (preparar), los pines de iocontrol correspondientes también deben estar puenteados.

8.3.5.5. Escribiendo el cambio y preparando procedimientos O-word

Los prologos y epilogos estándar encontrados en `ncfiles/remap_lib/python-stdglue/stdglue.py` pasan algunos *parámetros expuestos* al procedimiento de remapeado.

Un *parámetro expuesto* es una variable local nombrada visible en un procedimiento de remapeado que corresponde a la variable interna del intérprete que es relevante para el remapeado actual. Los parámetros expuestos se establecen en el prologo respectivo y se inspeccionan en el epilogo. Se puede cambiar en el procedimiento de remapeado y se recogerá el cambio en el epilogo. Los parámetros expuestos para códigos incorporados remapeables son:

- `T` (prepare_prolog): `#<tool>` , `#<pocket>`
- `M6` (change_prolog): `#<tool_in_spindle>`, `#<selected_tool>`, `#<current_pocket>`, `#<selected_pocket>`
- `M61` (settool_prolog): `#<tool>` , `#<pocket>`
- `S` (setspeed_prolog): `#<speed>`
- `F` (setfeed_prolog): `#<feed>`

Si tiene necesidad específica de hacer visibles parámetros adicionales, simplemente agregelos al prologo; prácticamente todas las partes internas del intérprete son visibles para Python.

8.3.5.6. Haciendo cambios mínimos a los códigos incorporados, incluyendo `M6`

Recuerde que, normalmente, el remapeo de un código desactiva completamente todo el procesamiento interno para ese código.

Sin embargo, en algunas situaciones podría ser suficiente agregar algunos códigos alrededor del `M6` existente, como una sonda de longitud de herramienta, pero que conserve el comportamiento de `M6`.

Dado que este podría ser un escenario común, el comportamiento de los códigos reasignados se han puesto a disposición dentro del procedimiento de remapeado. El intérprete detecta que nos estamos refiriendo a un código reasignado dentro del procedimiento que se supone que redefine su comportamiento. En este caso, se utiliza el comportamiento incorporado - este actualmente está habilitado para el conjunto: `M6`,` M61`, `T`,` S`, `F`). Note que de lo contrario, referirse a un código dentro de su propio procedimiento de remapeado sería un error - una `recursión remapping`.

Retorciendo un poco una incorporada se vería así (en el caso de `M6`):

```
REMAP=M6 modalgroup=6 ngc=mychange
```

```
o<mychange> sub
M6 (uso de comportamiento incorporado de M6)
(.. mover al interruptor de longitud de la herramienta, probar y ajustar la longitud de la ↩
    herramienta ..)
o<mychange> endsub
m2
```

PRECAUCIÓN: al redefinir un código incorporado, **no especifique ningún cero encabezando los códigos G o M**; por ejemplo, diga `REMAP=M1 ..`, no `REMAP=M01 ...`.

Vea el directorio `configs/sim/axis/remap/extend-builtins` para una configuración completa que es el punto de partida recomendado para su trabajo propio.

8.3.5.7. Especificando el reemplazo de T (preparar)

Si está a gusto con la implementación por defecto, no necesitaría hacer esto. Pero el remapeado es también una forma de solucionar las deficiencias en la implementación actual, por ejemplo, no bloquear hasta que se establezca el pin "tool-prepared".

Lo que podría hacer, por ejemplo, es: - en una T remapeada, simplemente establezca el equivalente del pin "tool-prepare", pero **no** espere "tool-prepared" aquí - en el M6 remapeado correspondiente, espere a "tool-prepared" al principio del procedimiento O-word.

Nuevamente, los pines de iocontrol tool-prepare/tool-ready no se utilizarían y serian reemplazados por pines `motion.*`, por lo que esos pines deben estar puenteados:

```
# puentear señales preparar al reasignar T
net tool-prep-loop iocontrol.0.tool-prepare iocontrol.0.tool-prepared
```

Aquí está la configuración para una T reasignada:

```
REMAP=T  prolog=prepare_prolog epilog=prepare_epilog ngc=prepare
```

```
def prepare_prolog(self,**words):
    try:
        cblock = self.blocks[self.remap_level]
        if not cblock.t_flag:
            return "T requiere un número de herramienta"

        tool  = cblock.t_number
        if tool:
            (status, pocket) = self.find_tool_pocket(tool)
            if status != INTERP_OK:
                return "T%d: ranura no encontrado" % (tool)
        else:
            pocket = -1 # esto es T0 - descarga de herramienta

        # estas variables serán visibles en la sub oword de ngc
        # como variables locales #<tool> y #<pocket> , y pueden ser
        # modificadas allí - el epilogo recuperará los valores
        # cambiados
        self.params["tool"] = tool
        self.params["pocket"] = pocket

        return INTERP_OK
    except Exception, e:
        return "T%d/prepare_prolog: %s" % (int(words['t']), e)
```

El procedimiento mínimo de preparación de ngc de nuevo se ve así:

```
o<prepare> sub
; Devolviendo un valor positivo:
o<prepare> endsub [1]
m2
```

Y el epilogo:

```
def prepare_epilog(self, **words):
    try:
        if self.return_value > 0:
            self.selected_tool = int(self.params["tool"])
            self.selected_pocket = int(self.params["pocket"])
            emccanon.SELECT_POCKET(self.selected_pocket, self.selected_tool)
            return INTERP_OK
        else:
```

```
            return "T%d: abortado (código de retorno% .1f)" % (int(self.params["tool"]), ↩
                self.return_value)

    except Exception, e:
        return "T%d/prepare_epilog: %s" % (tool,e)
```

prepare_prolog y prepare_epilog son parte del *codigo de union estándar* proporcionado por *nc_files/remap_lib/python-stdglue/stdglue.py*
Este módulo está destinado a cubrir la mayoría de situaciones estándar de remapeado de una manera común.

8.3.5.8. Manejo de errores: tratando con abort

El procedimiento de cambio de herramienta incorporado tiene algunas precauciones para tratar con un aborte de programa (por ejemplo, al presionar Escape in Axis durante un cambio). Su función reasignada no tiene nada de esto, por lo tanto, alguna limpieza explícita podría ser necesaria si un código reasignado es abortado. En particular, un procedimiento de remapeado podría establecer ajustes modales que son indeseables tener activos después de un abort. Por ejemplo, si su procedimiento de remapeado tiene códigos de movimiento (G0, G1, G38 ..) y el remapeado es abortado, entonces el último código modal permanecerá activo. Sin embargo, es muy probable que desee que se cancele cualquier movimiento modal cuando el remapeado es abortado.

La forma de hacerlo es mediante el uso de la característica `[RS274NGC]ON_ABORT_COMMAND`. Esta opción de ini especifica una llamada de procedimiento O-word que es ejecutada si Task, por alguna razón, aborta la ejecución del programa.

```
[RS274NGC]
ON_ABORT_COMMAND=O <on_abort> call
```

El procedimiento on_abort sugerido se vería así (adaptelo a sus necesidades):

```
o<on_abort> sub

G54 (las compensaciones de origen se establecen en el valor predeterminado)
G17 (seleccion del plano XY)
G90 (modo absoluto)
G94 (modo de alimentación: unidades/minuto)
M48 (ajuste de velocidad de avance y husillo)
G40 (compensación de corte desactivada)
M5  (husillo apagado)
G80 (cancelar movimiento modal)
M9  (niebla y refrigerante apagado)

o<on_abort> endsub
m2
```

PRECAUCION: Nunca use un `M2` dentro de una subrutina O-word, incluyendo esta. Esto causará errores difíciles de encontrar. Por ejemplo, usando un `M2` en una subrutina, no terminará la subrutina correctamente y dejará el archivo NGC del subprograma abierto, no el programa principal.

Asegúrese de que `on_abort.ngc` esté en la ruta de búsqueda del intérprete (ubicación recomendada: `SUBROUTINE_PATH` para no desordenar su directorio `NC_FILES` con procedimientos internos). `on_abort` recibe un un solo parámetro que indica la causa de llamada al procedimiento de abortado, que podría ser utilizado para la limpieza condicional.

Las declaraciones en ese procedimiento típicamente aseguran que el post-aborto ha limpiado cualquier estado, y que los pines HAL se restablecieron correctamente. Por ejemplo, vea `configs/sim/axis/remap/rack-toolchange`.

Tenga en cuenta que terminar un código reasignado devolviendo INTERP_ERROR desde el epilogo (ver la sección anterior) también causará llamada al procedimiento `on_abort`.

8.3.5.9. Manejo de errores: error en un procedimiento NGC de remapeado de código

Si determina en su procedimiento de manejo que ocurrio alguna condición de error, no use `M2` para finalizar su manejador - vea mas arriba.

Si se muestra un mensaje de error al operador y es suficientemente aceptable detener el programa actual, use la característica `(abort, <message>)` para terminar el manejador con un mensaje de error. Tenga en cuenta que puede sustituir parámetros HAL numerados, nombrados e ini en el texto como en este ejemplo (vea también `tests/interp/abort-hot-comment/test.ngc`):

```
o100 if [..] (alguna condición de error)
     (abort, ¡Algo va Mal! p42=#42 q=#<q> ini=#<_ini[a]x> pin=#<_hal[component.pin])
o100 endif
```

NOTA: la expansión de variables ini y HAL es opcional y se pueden deshabilitar en el archivo INI

Si se necesita una acción de recuperación más precisa, use lo presentado en el ejemplo anterior:

- defina una función de epilogo, incluso si es solo para señalar una condición de error.
- pasar un valor negativo desde el manejador para señalar el error
- inspeccionar el valor de retorno en la función de epilogo.
- tomar cualquier acción de recuperación necesaria
- devolver la cadena de mensaje de error desde el manejador, que establecerá el mensaje de error del intérprete y aborta el programa (casi como `abort, message=`)

Este mensaje de error se mostrará en la interfaz de usuario, y devolviendo INTERP_ERROR provocará que este error se maneje como cualquier otro error de tiempo de ejecución.

Tenga en cuenta que tanto `(abort, msg)` como devolver INTERP_ERROR desde un epilogo hará que también se llame a cualquier manejador ON_ABORT si está definido (ver apartado anterior).

8.3.6. Reasignando otros códigos existentes: S, M0, M1, M60

8.3.6.1. Selección automática de marcha reasignando S (ajuste de la velocidad del husillo)

Un uso potencial para un código S reasignado sería una *Selección automática de marcha* dependiendo de la velocidad. En el procedimiento de remapeado, se probaría la velocidad deseada alcanzable dada la configuración actual de engranajes, y cambiaria de marcha adecuadamente si no es así.

8.3.6.2. Ajustando el comportamiento de M0, M1, M60

Un caso de uso para el remapeado de M0/M1 sería personalizar el comportamiento del código existente. Por ejemplo, podría ser deseable desactivar el husillo, la niebla y la inundación durante una pausa del programa M0 o M1, y configurar el reencendido cuando se reanude el programa.

Para un ejemplo completo haciendo eso, vea *configs/sim/axis/remap/extend-builtins/*, que adapta M1 como se muestra arriba.

8.3.7. Creando nuevos ciclos de codigo G

Un ciclo de código G, como se usa aquí, debe comportarse de la siguiente manera:

- En la primera invocación, se recogen las palabras asociadas y se ejecuta el ciclo de codigo G
- Si en las líneas subsiguientes simplemente continúan las palabras de parámetro aplicables a este código, pero no un nuevo código G, el código G anterior se vuelve a ejecutar con los parámetros cambiados en consecuencia.

Un ejemplo: Supongamos que tiene un `G84.3` definido como ciclo de código G reasignado con el siguiente segmento ini (ver aquí para una descripción detallada de `cycle_prolog` y `cycle_epilog`):

```
[RS274NGC]
# Un ciclo con un procedimiento oword: G84.3 <X- Y- Z- Q- P->
REMAP=G84.3 argspec=xyzabcuvwpr prolog=cycle_prolog ngc=g843 epilog=cycle_epilog modalgroup ↩
    =1
```

Ejecutando las siguientes lineas:

```
g17
(1) g84.3 x1 y2 z3 r1
(2) x3 y4 p2
(3) x6 y7 z5
(4) G80
```

provoca lo siguiente (*R* es sticky y *Z* es sticky porque el plano es *XY*):

1. `g843.ngc` se llama con las palabras x = 1, y = 2, z = 3, r = 1
2. `g843.ngc` se llama con las palabras x = 3, y = 4, z = 3, p = 2, r = 1
3. `g843.ngc` se llama con las palabras x = 6, y = 7, z = 3, r = 1
4. El ciclo `G84.3` se cancela.

Además de crear nuevos ciclos, esto proporciona un método fácil para reempaquetar códigos G existentes que no se comportan como ciclos. Por ejemplo, el código de roscado rígido `G33.1` no se comporta como un ciclo. Con tal envoltorio, se puede crear fácilmente un nuevo código que use `G33.1` pero se comporte como un ciclo.

Vea *configs/sim/axis/remap/cycle* para un ejemplo completo de esta característica. Contiene dos ciclos, uno con un procedimiento NGC como el anterior, y un ejemplo de ciclo usando solo Python.

8.3.8. Configurando Python Embebido

El complemento de Python sirve tanto al intérprete como a task, si es configurado así, y por lo tanto tiene su propia sección `PYTHON` en el archivo ini.

8.3.8.1. Plugin Python : configuración de archivos ini

`[PYTHON]`

TOPLEVEL=*<nombre de archivo>*
: nombre de archivo de la secuencia de comandos de Python inicial para ejecutar en la puesta en marcha. Este script es responsable de configurar la estructura del nombre del paquete, ver más abajo.

PATH_PREPEND=*<directorio>*
: añade delante este directorio a `PYTHON_PATH`. Repetible.

PATH_APPEND=*<directorio>*
: agrega detras este directorio a `PYTHON_PATH`. Repetible.

LOG_LEVEL=*<integer>*
: Nivel de registro de las acciones relacionadas con el plugin. Aumente esto si sospecha problemas. Puede ser muy detallado.

RELOAD_ON_CHANGE=*[0 | 1]*
: vuelve a cargar la secuencia de comandos *TOPLEVEL* si se cambió el archivo. Práctico para la depuración, pero actualmente incurre en una sobrecarga de tiempo de ejecución. Apaguelo para configuraciones de producción.

PYTHON_TASK=*[0 | 1]*
: Inicia el complemento de tareas de Python. Experimental. Ver xxx.

8.3.8.2. Ejecutando sentencias de Python desde el intérprete

Para la ejecución ad hoc de comandos, ha sido añadido el *comentario caliente* de Python. La salida de Python por defecto va a la salida estándar, por lo que necesita comenzar LinuxCNC desde una ventana de terminal para ver los resultados. Ejemplo (por ejemplo, en la ventana MDI):

```
;py,print 2*3
```

Tenga en cuenta que la instancia del intérprete está disponible aquí como `self`, por lo que también podría correr:

```
;py,print self.tool_table[0].toolno
```

La estructura `emcStatus` también es accesible:

```
;py,from emctask import *
;py,print emcstat.io.aux.estop
```

8.3.9. Programación de Python Embebido en el intérprete RS274NGC

8.3.9.1. El espacio de nombres del plugin Python

Se espera que el espacio de nombres se distribuya de la siguiente manera:

`oword`
: Cualquier codigo llamable en este módulo es candidato para procedimientos Python O-word Tenga en cuenta que el módulo de Python `oword` se testea **antes** que un procedimiento NGC con el mismo nombre - en efecto, nombres en `oword` ocultarán los archivos NGC del mismo nombre base

`remap`
: Cualquier codigo llamable Python referenciado en un argspec `prolog`, `epilog` u opción `python`, se espera que se encuentre aquí.

`namedparams`
: Las funciones de Python en este módulo amplían o redefinen el espacio de nombres de parámetros nombrados predefinidos, ver agregar parámetros predefinidos.

`task`
: Aquí se esperan codigos llamables relacionados con task.

8.3.9.2. El intérprete visto desde Python

El intérprete es una clase existente C++ (*Interp*) definida en *src/emc/rs274ngc*. Conceptualmente, todos las llamadas a Python `oword.<function>` y `remap.<function>` son métodos de esta clase Interp, aunque no hay una definición explícita de Python de esta clase (es una instancia de envoltorio *Boost.Python*) y, por lo tanto, recibe el primer parámetro *self* que se puede utilizar para acceder a elementos internos.

8.3.9.3. Las funciones del intérprete `__init__` y `__delete__`

Si el módulo `TOPLEVEL` define una función `__init__`, será llamada una vez que el intérprete está totalmente configurado (archivo ini leído, y estado sincronizado con el modelo mundial).

Si el módulo `TOPLEVEL` define una función `__delete__`, será llamada una vez antes que el intérprete se apague y después de que los parámetros persistentes se han guardado en `PARAMETER_FILE`.

Nota_ en este momento, el manejador `__delete__` no funciona para instancias de intérprete creadas importando el módulo `gcode`. Si necesita una funcionalidad equivalente (lo cual es bastante improbable), por favor considere el módulo Python `atexit`.

```
# esto sería definido en el módulo TOPLEVEL

def __init__(self):
    # agregar cualquier inicialización unica aquí
    if self.task:
  # esta es la instancia milltask de interp
  pass
    else:
  # esta es una instancia de interp no-milltask
        pass

def __delete__(self):
    # agregar cualquier acción de limpieza/salvado de estado aquí
    if self.task: # como arriba
  pass
    else:
        pass
```

Esta función se puede utilizar para inicializar cualquier atributo del lado de Python que puede ser necesario más adelante, por ejemplo, en funciones remap u o-word, y guardar o restaurar el estado más allá de lo que proporciona `PARAMETER_FILE`.

Si hay acciones de configuración o limpieza que van a ocurrir solo en la instancia milltask del intérprete (a diferencia de la instancia de intérprete que se encuentra en el módulo Python `gcode` y sirve propósitos de visualización de vista previa/progreso pero nada más), esto puede ser probado por evaluar self.task.

Un ejemplo de uso de `__init__` y `__delete__` se puede encontrar en *configs/sim/axis/remap/cycle/python/toplevel.py* inicializando los atributos necesario para manejar los ciclos en *ncfiles/remap_lib/python-stdglue/stdglue.py* (e importado a *configs/sim/axis/remap/cycle/python/remap.py*).

8.3.9.4. Convenciones de llamada: NGC a Python

El código Python se llama desde NGC en las siguientes situaciones:

- durante la ejecución normal del programa:
  - cuando se ejecuta una llamada O-word como `O<proc> call` y el nombre `oword.proc` está definido y es llamable
  - cuando se ejecuta un comentario como `;py,<Python statement>`
- durante la ejecución de un código reasignado: cualquier manejador `prolog =`, `python =` y `epilog =`.

Llamar a subrutinas Python O-word

Argumentos:

`self`
: la instancia del intérprete

`*args`
: La lista de parámetros posicionales reales. Ya que el numero de los parámetros reales pueden variar, es mejor usar este estilo de declaración:

```
# esto sería definido en el módulo oword
def mysub(self, *args):
    print "número de parámetros pasados:", len(args)
    for a in args:
  print a
```

Devolver los valores de las subrutinas Python de O-word Al igual que los procedimientos NGC pueden devolver valores, también lo hacen las subrutinas O-word Python. Se espera que sean uno de los siguientes:

- no devuelve ningún valor (no hay una declaración `return` o el valor `None`)
- un valor float o int
- una cadena, esto significa *esto es un mensaje de error, abortar el programa*. Funciona como `(abort, msg)`.

Cualquier otro tipo de valor de retorno generará una excepción de Python.

En un entorno NGC de llamada, los siguientes parámetros nombrados predefinidos están disponibles:

#<_value>
: Valor devuelto por el último procedimiento llamado. Inicializado a 0.0 en el inicio. Expuesto en Interp como `self.return_value` (float).

#<_value_returned>
: indica el último procedimiento llamado `devuelto` o `endsub` con un valor explícito. 1.0 si es cierto. Establecido a 0.0 en cada *call*. Expuesto en Interp como `self.value_returned` (int).

Vea también `tests/interp/value-return` para un ejemplo.

Convenciones de llamada para las subrutinas prolog= y epilog= Los argumentos son:

self
: la instancia del intérprete

words
: parámetro diccionario de palabras clave. Si estaba presente un argspec, se recogen del bloque actual en consecuencia y se pasan al diccionario por conveniencia (las palabras también podrían ser recuperadas directamente del bloque llamante, pero esto requiere más conocimientos internos del intérprete). Si no se pasó argspec, o solo se especificaron valores opcionales y ninguno de estos estaban presentes en el bloque llamante, este diccionario estará vacío. Los nombres de las palabras se convierten a minúsculas.

Ejemplo de llamada:

```
def minimal_prolog(self, **words): # in remap module
    print len(words),"palabras pasadas"
    for w in words:
        print "%s: %s" % (w, words[w])
    if words['p'] < 78: # NB: podría provocar una excepción si p fuera opcional
       retornando "fallando miserablemente"
    return INTERP_OK
```

Valores de retorno:

INTERP_OK
: devolver esto en éxito. Se necesita importar esto desde "interpreter".

"un mensaje de texto"
: devolver una cadena desde un manejador significa 'esto es un mensaje de error, abortar el programa '. Funciona como `(abortar, msg)`.

.

Convenciones de llamada para las subrutinas python= Los argumentos son:

self
: la instancia del intérprete

words
: parámetro diccionario de palabras clave. el mismo diccionario kwargs que prolog y epilog (ver arriba).

Ejemplo mínimo de la función `python=`:

```
def useless(self,  **words): # en el módulo de remapeado
    return INTERP_OK
```

Valores de retorno:

INTERP_OK
: devolver esto en éxito

"mensaje de texto"
: devolver una cadena desde un manejador significa 'esto es un mensaje de error, abortar el programa '. Funciona como `(abort, msg)`.

Si el manejador necesita ejecutar una operación *queuebuster*(cambio de herramienta, sonda, lectura del pin HAL) se supone que se suspende la ejecución con la siguiente declaración:

yield INTERP_EXECUTE_FINISH
: Esto señala a task para que detenga la lectura adelantada, ejecuta todas las operaciones en cola, ejecutar la operación *queue-buster*, sincroniza el estado del intérprete con el estado de la máquina, y luego señaliza al intérprete que continue. En este punto la función es reanudada en la declaración siguiente a la declaración `yield ...`

Tratando con queue-buster: Sonda, Cambio de Herramienta y espera de un pin HAL Los destructores de colas interrumpen un procedimiento en el punto en que se llama a tal operación, por lo tanto el procedimiento debe ser reiniciado después de synch() del intérprete. Cuando esto sucede, el procedimiento necesita saber si se reinicia, y dónde continuar. El método generador de Python se utiliza para tratar el reinicio del procedimiento.

Esto demuestra la continuación de la llamada con un solo punto de reinicio:

```
def read_pin(self,*args):
    # espere 5 segundos para que la entrada digital 00 sea alta
    emccanon.WAIT(0,1,2,5.0)
    # cede el control después de ejecutar el destructor de colas:
    yield INTERP_EXECUTE_FINISH
    # La ejecución de post-sync() se reanuda aquí:
    pin_status = emccanon.GET_EXTERNAL_DIGITAL_INPUT(0,0);
    print "pin status=",pin_status
```

ADVERTENCIA: La característica *yield* es frágil. Las siguientes restricciones se aplica al uso de *yield INTERP_EXECUTE_FINISH*:

- El código Python que ejecuta un *yield INTERP_EXECUTE_FINISH* debe ser parte de un procedimiento de remapeado. yield no funciona en un procedimiento Python o-word.
- Una subrutina de remapeado de Python que contiene la declaración de *yield INTERP_EXECUTE_FINISH* puede no devolver un valor, como ocurre con las declaraciones yield de Python normales.
- El código que sigue a un yield no puede llamar al intérprete de forma recursiva, como con self.execute("<comando_mdi>"). Esta es una restricción de la arquitectura del intérprete y no se puede reparar sin un rediseño importante.

8.3.9.5. Convenciones de llamada: Python a NGC

El código NGC se ejecuta desde Python cuando:

- se ejecuta el método `self.execute(<código NGC>[,<número_de_línea>])`
- durante la ejecución de un código reasignado, si está definida una función `prolog=`, el procedimiento NGC dado en `ngc=` se ejecuta inmediatamente.

El manejador prolog no llama al manejador, sino que prepara el entorno de llamada, por ejemplo, mediante la configuración de parámetros locales predefinidos.

Insertando parámetros en un prolog, y recuperándolos en un epilog Conceptualmente un prolog y un epilog se ejecutan al mismo nivel de llamada que un procedimiento O-word, es decir, después de que se establece la llamada de subrutina y antes de que la subrutina finalice o regrese.

Esto significa que cualquier variable local creada en un prolog será una variable local en un procedimiento O-word, y cualquier variable local creada en el procedimiento O-word todavía es accesible cuando se ejecuta el epilog.

La matriz `self.params` maneja la lectura y configuración de parámetros numerados y nombrados. Si un parámetro con nombre comienza con _ (guión bajo), se asume que es un parámetro global; si no, es local al procedimiento llamante. Además, los parámetros numerados en el rango 1..30 se tratan como variables locales; sus valores originales son restaurados en los return/endsub de procedimientos O-word.

Aquí hay un ejemplo de código reasignado que demuestra la inserción y extracción de parámetros en/desde un procedimiento O-word:

```
REMAP=m300 prolog=insert_param ngc=testparam epilog=retrieve_param modalgroup=10
```

```
def insert_param (self, **words): # en el módulo remapeado
    print "insert_param call level=",self.call_level
    self.params["myname"] = 123
    self.params[1] = 345
    self.params[2] = 678
    return INTERP_OK

def retrieve_param(self, **words):
    print "retrieve_param call level=",self.call_level
    print "#1=", self.params[1]
    print "#2=", self.params[2]
    try:
        print "result=", self.params["result"]
    except Exception,e:
  return "testparam olvidó asignar #<result>"
    return INTERP_OK
```

```
o<testparam> sub
(debug, call_level=#<_call_level> myname=#<myname>)
; Intente descomentar la siguiente línea y corra otra vez.
#<result> = [#<myname> * 3]
#1 = [#1 * 5]
#2 = [#2 * 3]
o<testparam> endsub
m2
```

`self.params()` devuelve una lista de todos los nombres de variables actualmente definidos. Como `myname` es local, desaparece después de que finaliza el epilog.

Llamar al intérprete desde Python Puede llamar de forma recursiva al intérprete desde el código de Python de la siguiente manera:

```
self.execute(<código NGC>[,<número de línea>])
```

Ejemplos:

```
  self.execute("G1 X%f Y%f" % (x,y))
  self.execute("O <myprocedure> call", currentline)
```

Es posible que desee probar si el valor de retorno es menor que `INTERP_MIN_ERROR`. Si está usando muchas instrucciones execute(), es probablemente sea más fácil atrapar InterpreterException como se muestra a continuación.

PRECAUCIÓN: el método de inserción/recuperación de parámetros descrito en la sección anterior no trabaja en este caso. Es lo suficientemente bueno para comandos ejecutar simples NGC o una llamada de procedimiento e introspección avanzada en el procedimiento, y el paso de los parámetros locales con nombre no es necesario. La caracteristica de llamada recursiva es frágil.

Excepción del intérprete durante execute() Si `interpreter.throw_exceptions` es distinto de cero (valor predeterminado 1), y self.execute() devuelve un error, se genera la excepción `InterpreterException`. InterpreterException tiene los siguientes atributos:

`line_number`
: donde ocurrió el error

`line_text`
: la sentencia NGC causando el error

`error_message`
: mensaje de error del intérprete

Los errores pueden ser atrapados de la siguiente manera:

```
import interpreter
interpreter.throw_exceptions = 1
   ...
   try:
        self.execute("G3456")  #  raise InterpreterException

   except InterpreterException,e:
        msg = "%d: '%s' - %s" % (e.line_number,e.line_text, e.error_message)
        return msg  # reemplazar el mensaje de error incorporado
```

Canon La capa canonica está prácticamente compuesta de funciones libres. Ejemplo:

```
import emccanon
def example(self,*args):
    ....
    emccanon.STRAIGHT_TRAVERSE(line,x0,y0,z0,0,0,0,0,0,0)
    emccanon.STRAIGHT_FEED(line,x1,y1,z1,0,0,0,0,0,0)
    ...
    return INTERP_OK
```

Las funciones canonicas reales se declaran en `src/emc/nml_intf/canon.hh` y se implementan en `src/emc/task/emccanon.cc`. La implementación de las funciones Python se pueden encontrar en `src/emc/rs274ncg/canonmodule.cc`.

8.3.9.6. Módulos Integrados

Los siguientes módulos están integrados:

`interpreter`
: Expone la clase Interp. Ver `src/emc/rs274ngc/interpmodule.cc`, y el test de regresión `tests/remap/introspect`.

`emccanon`
: expone la mayoría de las llamadas de `src/emc/task/emccanon.cc`.

`emctask`
: expone la instancia de la clase `emcStatus`. Consulte `src/emc/task/taskmodule.cc`. No presente cuando se usa el módulo `gcode` usado para interfaces de usuario - solo está presente en la instancia miltask del intérprete.

8.3.10. Agregando Parámetros Nombrados Predefinidos

El intérprete viene con un conjunto de parámetros nombrados predefinidos para acceso al estado interno desde el nivel NGC. Estos parametros son de solo lectura y globales, y por lo tanto pueden asignarse.

Se pueden agregar parámetros adicionales definiendo una función en el módulo `namedparams`. El nombre de la función define el nombre del nuevo parámetro nombrado predefinido, que ahora puede ser referenciado en expresiones arbitrarias.

Para agregar o redefinir un parámetro nombrado:

- agregue un módulo `namedparams` para que el intérprete lo pueda encontrar
- Definir nuevos parámetros por funciones (ver abajo). Estas funciones reciben `self` (la instancia del intérprete) como parámetro y así pueden acceder a estados aribtrarios. Las capacidades arbitrarias de Python se pueden usar para devolver un valor.
- importar ese módulo desde el script `TOPLEVEL`

```
# namedparams.py
# ejemplo trivial
def _pi(self):
    return 3.1415926535
```

```
#<circumference> = [2 * #<radius> * #<_pi>]
```

Se espera que las funciones en `namedparams.py` devuelvan un valor float o int. Si se devuelve una cadena, se establece el mensaje de error del intérprete y aborta la ejecución.

Sólo se agregan funciones con un guión bajo como parámetros, ya que esta es la convención RS274NGC para globales.

Es posible redefinir un parámetro predefinido existente agregando una función de Python con el mismo nombre que el módulo `namedparams`. En este caso, se genera una advertencia durante el inicio.

Si bien el ejemplo anterior no es terriblemente útil, tenga en cuenta que todo el estado interno del intérprete es accesible desde Python, por lo que los predicados arbitrarios se pueden definir de esta manera. Para un ejemplo algo más avanzado, vea `tests/remap/predefined-named-params`.

8.3.11. Rutinas estándar de union

Dado que muchas tareas de remapeado son muy similares, se comenzo a recopilar rutinas de prolog y epilog en un solo módulo de Python. Actualmente estas se puede encontrar en *ncfiles/remap_lib/python-stdglue/stdglue.py*, que proporciona las siguientes rutinas:

8.3.11.1. T: `prepare_prolog` y `prepare_epilog`

Estos envuelven un procedimiento NGC para Tx Tool Prepare.

Acciones de prepare_prolog Los siguientes parámetros se hacen visibles para el procedimiento NGC:

- `#<tool>` - el parámetro de la palabra `T`
- `#<pocket>` - la ranura correspondiente

Si se solicita el número cero de herramienta (lo que significa descargar la herramienta), la ranura correspondiente se pasa como -1.

Es un error si:

- no se da ningún número de herramienta como parámetro T

- la herramienta no se puede encontrar en la tabla de herramientas.

Tenga en cuenta que a menos que establezca el parámetro `[EMCIO] RANDOM_TOOLCHANGER=1`, la herramienta y el número de ranura son idénticos, y el número de ranura de la tabla de herramientas se ignora. Esto es actualmente una restricción.

ACCIONES DE `PREPARE_EPILOG`

- Se espera que el procedimiento NGC devuelva un valor positivo, de lo contrario se da un mensaje de error que contiene el valor de retorno y el el intérprete aborta.
- En caso de que el procedimiento NGC ejecutara el comando T (que luego se refiere al comportamiento incorporado en T), no se toma ninguna otra acción. Esto puede ser utilizado por ejemplo para ajustar mínimamente el comportamiento incorporado en lo que precede o sigue con algunas otras declaraciones.
- De lo contrario, se extraen los parámetros `#<tool>` y `#<pocket>` del espacio de parámetros de la subrutina. Esto significa que el procedimiento NGC podría cambiar estos valores, y el epilog tiene los valores modificados en cuenta.
- despues, se ejecuta el comando canononico `SELECT_POCKET(#<pocket>,#<tool>)`.

8.3.11.2. M6: `change_prolog` y `change_epilog`

Estos envuelven un procedimiento NGC para M6 Tool Change.

ACCIONES DE `CHANGE_PROLOG`

- Los siguientes tres pasos son aplicables solo si se utiliza el componente `iocontrol-v2`:
  - Si el parámetro 5600 (indicador de fallo) es mayor que cero, esto indica un fallo del cambiador de herramientas, que se maneja de la siguiente manera:
  - Si el parámetro 5601 (código de error) es negativo, esto indica un error hard y el prolog aborta con un mensaje de error.
  - Si el parámetro 5601 (código de error) es mayor o igual a cero, esto indica un fallo soft. Se muestra un mensaje informativo y prolog continúa.
- Si no había un comando T precedente que causára que no fue seleccionada una ranura, prolog aborta con un mensaje de error.
- Si la compensación del radio de corte está activada, prolog se cancela con un mensaje de error.

Luego, los siguientes parámetros se exportan al procedimiento NGC:

- `#<tool_in_spindle>` : el número de herramienta de la herramienta cargada actualmente
- `#<selected_tool>` : el número de herramienta seleccionado
- `#<selected_pocket>` : el número de ranura de la herramienta seleccionada

ACCIONES DE `CHANGE_EPILOG`

- Se espera que el procedimiento NGC devuelva un valor positivo; de lo contrario se da un mensaje de error que contiene el valor de retorno y el intérprete aborta.
- Si el parámetro 5600 (indicador de fallo) es mayor que cero, esto indica un fallo del cambiador de herramientas, que se maneja de la siguiente manera (solo para `iocontrol-v2`):
  - Si el parámetro 5601 (código de error) es negativo, esto indica un error hard y el epilog se anula con un mensaje de error.
  - Si el parámetro 5601 (código de error) es mayor o igual a cero, esto indica un fallo soft. Se muestra un mensaje informativo y el epilog continúa.

- En caso de que el procedimiento NGC ejecutara el comando M6 (que luego se refiere al comportamiento M6 incorporado), no se realiza ninguna otra acción. Esto puede ser utilizado por ejemplo para ajustar mínimamente el comportamiento incorporado en lo que precede o sigue con algunas otras declaraciones.
- De lo contrario, se extrae el parámetro `#<selected_pocket>` del espacio de parámetros de la subrutina, y se utiliza para establecer la variable `current_pocket` del intérprete. De nuevo, el procedimiento podría cambiar este valor, y el epilog toma en cuenta el valor cambiado .
- entonces, el comando canonico `CHANGE_TOOL(#<selected_pocket>)` es ejecutado.
- Se establecen los nuevos parámetros de la herramienta (desplazamientos, diámetro, etc.).

8.3.11.3. Ciclos de código G: `cycle_prolog` y `cycle_epilog`

Estos envuelven un procedimiento de NGC para que pueda actuar como un ciclo, lo que significa que el código de movimiento se conserva después de finalizar la ejecución. Si la siguiente linea solo contiene palabras de parámetros (por ejemplo, nuevos valores de X, Y), el código es ejecutado de nuevo con las nuevas palabras de parámetros fusionadas en el conjunto de los parámetros dados en la primera invocación.

Estas rutinas están diseñadas para trabajar en conjunto con un parámetro `argspec=<words>`. Mientras esto es fácil de usar, en un escenario realista usted evitaría argspec y haría una investigación más a fondo del bloque de forma manual para dar mejor mensaje de error.

El argspec sugerido es el siguiente:

```
REMAP=G<somecode> argspec=xyzabcuvwqplr prolog=cycle_prolog ngc=<ngc procedure> epilog= ↩
    cycle_epilog modalgroup=1
```

Esto permitirá a `cycle_prolog` determinar la compatibilidad de cualquier palabra de eje dada en el bloque, ver más abajo.

ACCIONES DE `CYCLE_PROLOG`

- Determine si las palabras pasadas desde el bloque actual cumplen las condiciones descritas en Errores en ciclos fijos.
  - exportar las palabras del eje como + <x> +, + # <y> + etc; falla si las palabras del eje de diferentes grupos (XYZ) (UVW) se utilizan juntos, o se da cualquiera de (ABC).
  - exportar *L*- como `#<l>`; por defecto a 1 si no se da.
  - exportar *P*- como `#<p>`; fallo si p es menor que 0.
  - exportar *R*- como `#<r>`; fallo si no se da r, o es menor o igual a 0 si se da.
  - fallo si la velocidad de avance es cero, o el avance de tiempo inverso o la compensación del cortador está activas.
- Determine si esta es la primera invocación de un código G de ciclo, en cuyo caso:
  - Agregue las palabras pasadas (según argspec) en un conjunto de parámetros sticky, que se conservan a través de varias invocaciones.
- Si no es así (una línea de continuación con nuevos parámetros):
  - fusionar las palabras pasadas en el conjunto existente de parámetros sticky.
- exportar el conjunto de parámetros sticky al procedimiento NGC.

ACCIONES DE `CYCLE_EPILOG`

- Determine si el código actual era en realidad un ciclo, si es así:
  - retenga el modo de movimiento actual para que una línea de continuación sin un código de movimiento ejecute el mismo código de movimiento.

8.3.11.4. S (Establecer velocidad): `setspeed_prolog` y `setspeed_epilog`

TBD

8.3.11.5. F (Establecer Alimentacion): `setfeed_prolog` y `setfeed_epilog`

TBD

8.3.11.6. M61 Establecer el número de herramienta: `settool_prolog` y `settool_epilog`

TBD

8.3.12. Ejecución del código remapeado

8.3.12.1. Entorno de llamada a procedimiento NGC durante remapeados

Normalmente, un procedimiento de palabra O se llama con parámetros posicionales. Este esquema es muy limitante, en particular en presencia de parámetros opcionales. Por lo tanto, la convención de llamada se ha extendido para utilizar algo remotamente similar al modelo de argumentos de palabras clave de Python.

vea LINKTO gcode/main Subrutinas: sub, endsub, return, call.

8.3.12.2. Códigos reasignados anidados

Los códigos reasignados se pueden anidar al igual que las llamadas de procedimiento, es decir, un código reasignado cuyo procedimiento NGC se refiere a algún otro código reasignado se ejecutará correctamente.

El máximo nivel de anidacion de remapeados es actualmente 10.

8.3.12.3. Número de secuencia durante remapeos

Los números de secuencia se propagan y se restauran con las llamadas a palabra O. Consulte `tests/remap/nested-remaps/word` para la prueba de regresión, que muestra el seguimiento de los números de secuencia durante el anidamiento de tres niveles de remapeado.

8.3.12.4. Banderas de depuracion

Los siguientes indicadores son relevantes para la remapeado y ejecución relacionada con Python:

```
EMC_DEBUG_OWORD             0x00002000 rastrea la ejecución de subrutinas O-word
EMC_DEBUG_REMAP             0x00004000 rastrea la ejecución del código relacionado ↩
    con la remapeado
EMC_DEBUG_PYTHON            0x00008000 llamadas a complemento de Python
EMC_DEBUG_NAMEDPARAM        0x00010000 rastrea acceso a parámetros nombrados
EMC_DEBUG_PYTHON_TASK       0x00040000 rastrea el complemento de Python de task
EMC_DEBUG_USER1             0x10000000 definido por el usuario - no interpretado  ↩
   por LinuxCNC
EMC_DEBUG_USER2             0x20000000 definido por el usuario - no interpretado  ↩
   por LinuxCNC
```

combine con *or* estas banderas en la variable `[EMC]DEBUG` según sea necesario. Para una lista actual de indicadores de depuración, vea *src/emc/nml_intf/debugflags.h*.

8.3.12.5. Depuración de código Python embebido

La depuración del código Python enbebido es más difícil que la depuración normal de scripts Python, y solo existe un suministro limitado de depuradores. Una solución basada en código abierto que funciona es utilizar el http://www.eclipse.org [IDE Eclipse], Eclipse plugin http://www.pydev.org [PydDev] y su característica de depuracion remota.

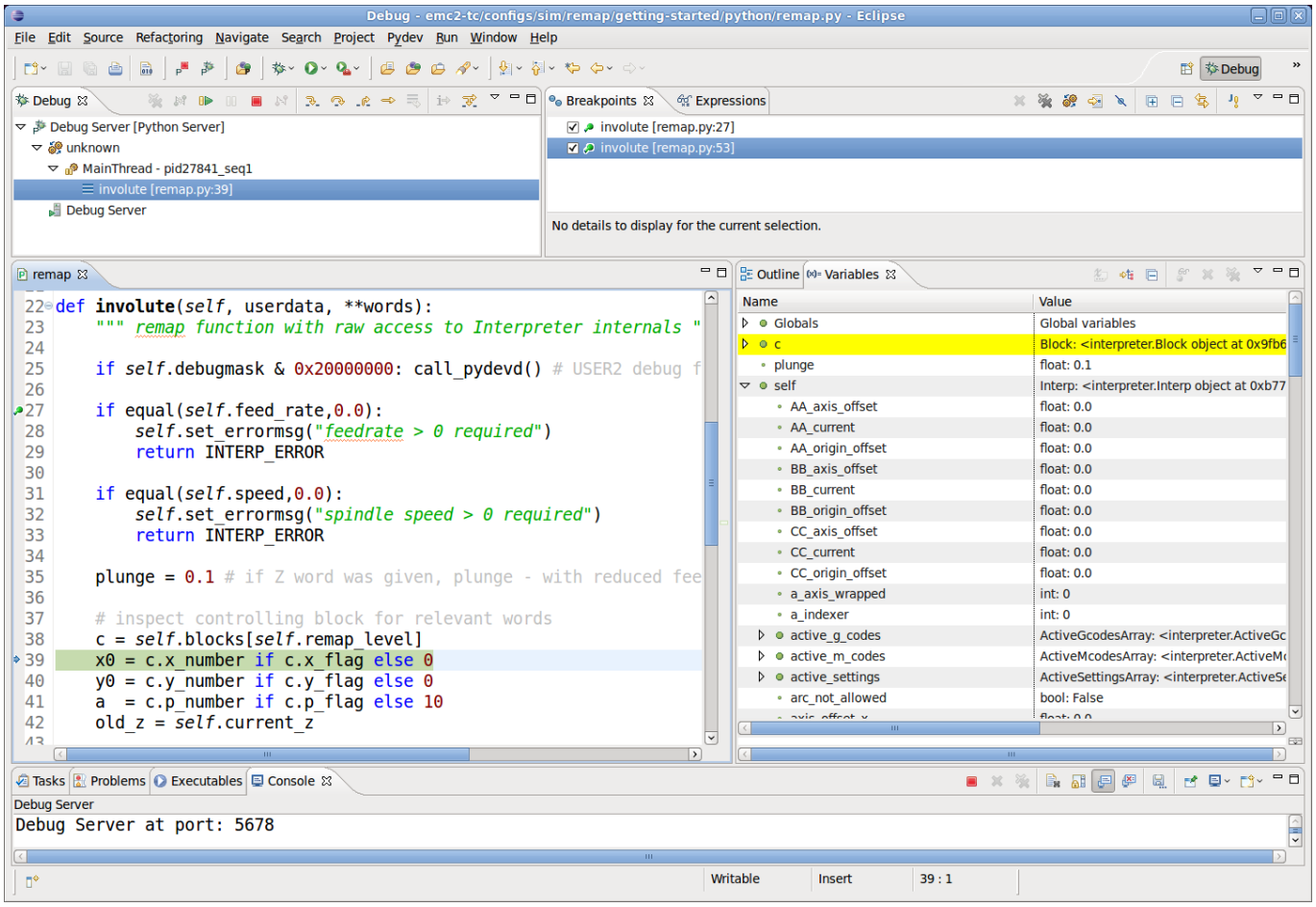
Para utilizar este enfoque:

- instale Eclipse a través del *Centro de software de Ubuntu* (elija primero selección)
- Instale el plugin PyDev desde Pydev Update Site
- configure el árbol fuente de LinuxCNC como un proyecto de Eclipse
- inicie el servidor de depuración Pydev en Eclipse
- asegúrese de que el código Python incrustado pueda encontrar el módulo `pydevd.py` que viene con ese plugin - está enterrado en algún lugar profundo debajo del directorio de instalación de Eclipse. Establezca la variable `pydevd` en `util.py` para reflejar esta ubicación del directorio.
- `import pydevd` en su módulo Python - vea los ejemplos `util.py` y `remap.py`
- llame a `pydevd.settrace()` en su módulo en algún punto para conectarse al servidor de depuración de Eclipse Python: aquí puede establecer puntos de interrupción en su Código, inspeccionar variables, pasos, etc., como de costumbre.

PRECAUCIÓN: `pydevd.settrace()` bloqueará la ejecución si Eclipse y el servidor de depuración Pydev no se ha iniciado.

Para cubrir los dos últimos pasos: el procedimiento `o<pydevd>` ayuda a entrar en el depurador desde el modo MDI. Véase también la función `call_pydevd` en `util.py` y su uso en` remap.involute` para establecer un punto de interrupción.

Aquí hay una captura de pantalla de Eclipse/PyDevd depurando el procedimiento `involute`:



Vea el código de Python en `configs/sim/axis/remap/getting-started/python` para más detalles.

8.3.13. Vista previa de Axis y Ejecución de código remapeado

Para obtener una vista previa completa del camino de la herramienta de un código remapeado, necesita tomar algunas precauciones. Para entender lo que está pasando, revisemos la vista previa y proceso de ejecución (esto cubre el caso de Axis, pero otros casos son similares):

Primero, tenga en cuenta que hay **dos** instancias de intérprete independientes involucradas:

- una instancia en el programa milltask, que ejecuta un programa cuando presiona el botón *Inicio* y hace que la máquina se mueva
- una segunda instancia en la interfaz de usuario cuyo propósito principal es generar la vista previa de ruta de herramienta. Éste *ejecuta* el programa una vez que está cargado, pero en realidad no causa movimientos de la máquina.

Ahora suponga que su procedimiento de remapeado contiene una operación de sonda G38, por ejemplo, como parte de un cambio de herramienta con touch-off de longitud de herramienta automático. Si la sonda falla, eso sería claramente un error, por lo que se debe mostrar un mensaje y abortar el programa.

Ahora, ¿qué pasa con la vista previa de este procedimiento?. En tiempo de vista previa, por supuesto no se sabe si la sonda tiene éxito o no, pero probablemente quiera ver cuál es la profundidad máxima de la sonda y suponga tiene éxito y continúa la ejecución hasta previsualizar nuevos movimientos. Además, no tiene sentido mostrar un mensaje de *sonda fallida* y abort **durante la vista previa**.

La forma de abordar este problema es probar en su procedimiento si se ejecuta en modo de vista previa o ejecución. Esto puede ser verificado probando el parámetro predefinido nombrado `#<_task>`- será 1 durante la ejecución real y 0 durante la vista previa. Ver *configs/sim/axis/remap/ manual-toolchange-with-tool-length-switch/nc_subroutines/manual_change.ngc* para un ejemplo de uso completo.

Dentro de Python embebido, la instancia de task puede ser verificada probando *self.task* - será 1 en la instancia de milltask, y 0 en la(s) instancia(s) de vista previa.

8.3.14. Códigos remapeables

8.3.14.1. Códigos existentes que pueden ser remapeados

El conjunto actual de códigos **existentes** abiertos para redefinición es:

- Tx (Preparar)
- M6 (Cambiar herramienta)
- M61 (Establecer número de herramienta)
- M0 (pausa temporalmente un programa en ejecución)
- M1 (pausar un programa en ejecución temporalmente si el interruptor de parada opcional está activado)
- M60 (intercambiar transbordadores de paletas y luego pausar un programa en ejecución temporalmente)
- S (ajuste la velocidad del husillo)
- F (ajuste de alimentación)

Tenga en cuenta que el uso de M61 actualmente requiere el uso de iocontrol-v2.

8.3.14.2. Códigos G actualmente sin asignar:

Los códigos G actualmente no asignados (para remapeado) deben seleccionarse de los espacios en blanco de las siguientes tablas. Todos los códigos G listados ya están definidos en la implementación actual de LinuxCNC y no se puede usar para reasignar nuevos códigos G (Se recomienda a los desarrolladores que agreguen nuevos códigos G a LinuxCNC que también agregue sus nuevos códigos G a estas tablas.)

Cuadro 8.1: Tabla de códigos G asignados_00-09

#	Gxx	Gxx.1	Gxx.2	Gxx.3	Gxx.4	Gxx.5	Gxx.6	Gxx.7	Gxx.8	Gxx.9
00	G00									
01	G01									
02	G02									
03	G03									
04	G04									
05	G05	G05.1	G05.2	G05.3						
06										
07	G07									
08	G08									
09										

Cuadro 8.2: Tabla de códigos G asignados_10-19

#	Gxx	Gxx.1	Gxx.2	Gxx.3	Gxx.4	Gxx.5	Gxx.6	Gxx.7	Gxx.8	Gxx.9
10	G10									
11										
12										
13										
14										
15										
16										
17	G17	G17.1								
18	G18	G18.1								
19	G19	G19.1								

Cuadro 8.3: Tabla de códigos G asignados_20-29

#	Gxx	Gxx.1	Gxx.2	Gxx.3	Gxx.4	Gxx.5	Gxx.6	Gxx.7	Gxx.8	Gxx.9
20	G20									
21	G21									
22										
23										
24										
25										
26										
27										
28	G28	G28.1								
29										

Cuadro 8.4: Tabla de códigos G asignados_30-39

#	Gxx	Gxx.1	Gxx.2	Gxx.3	Gxx.4	Gxx.5	Gxx.6	Gxx.7	Gxx.8	Gxx.9
30	G30	G30.1								
31										
32										
33	G30	G30.1								
34										
35										
36										
37										
38										
39										

Cuadro 8.5: Tabla de códigos G asignados_40-49

#	Gxx	Gxx.1	Gxx.2	Gxx.3	Gxx.4	Gxx.5	Gxx.6	Gxx.7	Gxx.8	Gxx.9
40	G40									
41	G41	G41.1								
42	G42	G42.1								
43	G43	G43.1								
44										
45										
46										
47										
48										
49	G40									

Cuadro 8.6: Tabla de códigos G asignados_50-59

#	Gxx	Gxx.1	Gxx.2	Gxx.3	Gxx.4	Gxx.5	Gxx.6	Gxx.7	Gxx.8	Gxx.9
50										
51										
52										
53	G53									
54	G54									
55	G55									
56	G56									
57	G57									
58	G58									
59	G59	G59.1	G59.2	G59.3						

Cuadro 8.7: Tabla de códigos G asignados_60-69

#	Gxx	Gxx.1	Gxx.2	Gxx.3	Gxx.4	Gxx.5	Gxx.6	Gxx.7	Gxx.8	Gxx.9
60	G60									
61	G61	G61.1								
62										
63										
64	G64									
65										
66										
67										
68										
69										

Cuadro 8.8: Tabla de códigos G asignados_70-79

#	Gxx	Gxx.1	Gxx.2	Gxx.3	Gxx.4	Gxx.5	Gxx.6	Gxx.7	Gxx.8	Gxx.9
70										
71										
72										
73										
74										
75										
76	G76									
77										
78										
79										

Cuadro 8.9: Tabla de códigos G asignados_80-89

#	Gxx	Gxx.1	Gxx.2	Gxx.3	Gxx.4	Gxx.5	Gxx.6	Gxx.7	Gxx.8	Gxx.9
80	G80									
81	G81									
82	G82									
83	G83									
84	G84									
85	G85									
86	G86									
87	G87									
88	G88									
89	G89									

Cuadro 8.10: Tabla de códigos G asignados_90-99

#	Gxx	Gxx.1	Gxx.2	Gxx.3	Gxx.4	Gxx.5	Gxx.6	Gxx.7	Gxx.8	Gxx.9
90	G90	G90.1								

Cuadro 8.10: (continued)

#	Gxx	Gxx.1	Gxx.2	Gxx.3	Gxx.4	Gxx.5	Gxx.6	Gxx.7	Gxx.8	Gxx.9
91	G91	G91.1								
92	G92	G92.1	G92.2	G92.3						
93	G93									
94	G94									
95	G95									
96	G96									
97	G97									
98	G98									
99	G99									

8.3.14.3. Códigos M actualmente sin asignar:

Estos códigos M actualmente no están definidos en la implementación actual de LinuxCNC y se puede utilizar para definir nuevos códigos M. (Desarrolladores que definan nuevos códigos M en LinuxCNC se recomienda que los elimine de esta tabla.)

Cuadro 8.11: Tabla de códigos M no asignados 00-99

#	Mx0	Mx1	Mx2	Mx3	Mx4	Mx5	Mx6	Mx7	Mx8	Mx9
00-09										
10-19	M10	M11	M12	M13	M14	M15	M16	M17	M18	
20-29	M20	M21	M22	M23	M24	M25	M26	M27	M28	M29
30-39		M31	M32	M33	M34	M35	M36	M37	M38	M39
40-49	M40	M41	M42	M43	M44	M45	M46	M47		
50-59					M54	M55	M56	M57	M58	M59
60-69										
70-79					M74	M75	M76	M77	M78	M79
80-89	M80	M81	M82	M83	M84	M85	M86	M87	M88	M89
90-99	M90	M91	M92	M93	M94	M95	M96	M97	M98	M99

Todos los códigos M de `M100` a` M199` ya son códigos M definidos por el usuario, y no deben ser remapeados.

Todos los códigos M de `M200` a` M999` están disponibles para la remapeado.

8.3.14.4. tiempo de lectura anticipada y tiempo de ejecución

foo

8.3.14.5. plugin / pickle hack

foo

8.3.14.6. Módulo, métodos, clases, etc. referencia.

foo

8.3.15. Introducción: Extender la ejecución de tareas

foo

8.3.15.1. ¿Por qué quieres cambiar la ejecución de tareas?

foo

8.3.15.2. Un diagrama: tarea, interp, iocontrol, UI (??)

foo

8.3.16. Modelos de ejecución de tareas

foo

8.3.16.1. Ejecución tradicional de iocontrol / iocontrolv2

foo

8.3.16.2. Redefiniendo procedimientos de IO

foo

8.3.16.3. Procedimientos de Python en tiempo de ejecución

foo

8.3.17. Una breve inspección de la ejecución del programa LinuxCNC

Para comprender el remapeado de códigos, podría ser útil inspeccionar la ejecución de task e intérprete en lo que se refiere al remapeado.

8.3.17.1. Estado del intérprete

Conceptualmente, el estado del intérprete consiste en variables que entran en las siguientes categorias:

1. información de configuración (típicamente del archivo INI)
2. El *modelo mundial*: una representación del estado real de la máquina.
3. Estado modal y ajustes
4. estado de ejecución del intérprete

(3) se refiere al estado que se "transfiere" entre la ejecución de códigos NGC individuales: por ejemplo, una vez que el husillo está encendido y la velocidad establecida, permanece en este ajuste hasta que se desactiva. Lo mismo ocurre con muchos códigos, como alimentación, unidades, modos de movimiento (alimentación o rápido) y así sucesivamente.

(4) contiene información sobre el bloque actualmente ejecutado, ya sea que esté en una subrutina, variables de intérprete, etc.

La mayor parte de este estado se agrega en una, bastante poco sistemática, `structure _setup` (vea interp_internals.hh).

8.3.17.2. Interacción de Task e Interpreter, puesta en cola y lectura anticipada

La parte Task de LinuxCNC es responsable de coordinar los comandos de máquina real - movimiento, interacciones HAL y así sucesivamente. No lo hace por si misma manejando el lenguaje RS274NGC. Task apela al intérprete para analizar y ejecutar el siguiente comando, ya sea desde MDI o el archivo actual.

La ejecución del intérprete genera operaciones canónicas de máquina, que son las que en realidad mueven algo. Estas, sin embargo, no se ejecutan de inmediato, sino que se ponen en una cola. La ejecución real de estos códigos ocurre en la parte Task de LinuxCNC: los comandos canonicos se extraen de la cola de intérprete y se ejecutan, dando como resultado movimientos reales de la máquina.

Esto significa que normalmente el intérprete está muy por delante de la ejecución real de comandos: el análisis del programa bien podría haber terminado antes de que comience cualquier movimiento notable. Este comportamiento es llamado *lectura anticipada*.

8.3.17.3. Predecir la posición de la máquina

Para calcular de antemano las operaciones canónicas de la máquina durante la lectura anticipada, el intérprete debe poder predecir la posición de la máquina después de cada línea de Gcode, y eso no siempre es posible.

Veamos un programa de ejemplo simple que hace movimientos relativos. (G91), y supongamos que la máquina comienza en x = 0, y = 0, z = 0. Movimientos relativos implica que el resultado del próximo movimiento se basa en la posición del anterior:

```
N10 G91
N20 G0 X10 Y-5 Z20
N30 G1 Y20 Z-5
N40 G0 Z30
N50 M2
```

Aquí el intérprete puede predecir claramente las posiciones de la máquina para cada línea:

Después de N20: x = 10 y = -5 z = 20; después de N30: x = 10 y = 15 z = 15; después de N40: x = 10 y = 15 z = 45

y así pueden analizar todo el programa y generar operaciones canónicas con mucha antelación

8.3.17.4. El destructor de colas rompe la predicción de la posición

Sin embargo, la lectura adelantada completa solo es posible cuando el intérprete puede predecir el impacto de la posición para **cada** línea en el programa de antemano. Veamos un ejemplo modificado:

```
N10 G91
N20 G0 X10 Y-5 Z20
N30 G38.3 Z-10
N40 O100 if [#5070 EQ 0]
N50    G1 Y20 Z-5
N60 O100 else
N70    G0 Z30
N80 O100 endif
N90 G1 Z10
N95 M2
```

Para pre-calcular el movimiento en N90, el intérprete debería saber dónde está la máquina después de la línea N80, y eso depende de si el comando de la sonda tuvo éxito o no, lo cual no se conoce hasta que en realidad se ejecuta.

Por lo tanto, algunas operaciones son incompatibles con una lectura anticipada. Estas se llaman *busters de cola* (cazadores o destructores de colas), y son:

- leer el valor de un pin HAL con M66: el valor del pin HAL no es predecible.
- Cargar una nueva herramienta con M6: la geometría de la herramienta no es predecible.
- ejecutar un sondeo con G38.n: la posición final y el éxito/fracaso no son predecibles.

8.3.17.5. ¿Cómo se tratan los busters de cola?

Cada vez que el intérprete se encuentra con un destructor de colas, debe detener la lectura anticipada y esperar hasta que el resultado relevante esté disponible. La manera en que esto funciona es:

- cuando se encuentra dicho código, el intérprete devuelve un código de retorno especial a la tarea (*INTERP_EXECUTE_FINISH*).
- este código de retorno señala a task la detencion de lectura anticipada por ahora, ejecutar todos los comandos canónicos en cola acumulados hasta el momento (incluido el último, que es el destructor de la cola), y luego *sincronizar el estado del intérprete con el modelo universal*. Técnicamente, esto significa actualizar variables internas para reflejar los valores de pines HAL, recargar geometrías de herramienta después de un M6, y transmitir los resultados de una sonda.
- El método *synch()* del intérprete es llamado por task y hace solo eso - lee todos los valores *reales* del modelo universal que sean relevantes para la posterior ejecución.
- En este punto, task continúa y llama al intérprete para obtener más información por lectura adelantada - hasta que el programa finalice o se encuentre otro destructor de colas.

8.3.17.6. Orden de palabras y orden de ejecución

Una o varias *palabras* pueden estar presentes en un *bloque* NGC si son compatibles (algunas se excluyen mutuamente y deben estar en diferentes líneas). Sin embargo, el modelo de ejecución prescribe una ordenación estricta de ejecución de códigos, independientemente de su aparición en la línea de origen. (Orden de ejecución de G-Code).

8.3.17.7. Análisis

Una vez que se lee una línea (en modo MDI o desde el archivo NGC actual), se analiza y los indicadores y parámetros se configuran en una *estructura* (struct _setup, member block1). Esta estructura contiene toda la información sobre la línea de origen actual, pero independiente del orden diferente de códigos en la línea actual: siempre que varios códigos sean compatibles, cualquier orden fuente resultará en las mismas variables establecidas en el bloque de estructura. Inmediatamente después del análisis, se comprueba la compatibilidad de todos los códigos en un bloque.

8.3.17.8. Ejecución

Después de analizar con éxito, el bloque se ejecuta mediante execute_block(), y aquí los diferentes elementos se manejan de acuerdo al orden de ejecución.

Si se encuentra un "destructor de cola", se establece una marca correspondiente en el estado del intérprete (toolchange_flag, input_flag, probe_flag) y el intérprete devuelve un valor de retorno INTERP_EXECUTE_FINISH, señalizando *Detener readahead por ahora, y volver a sincronizar* con el llamador (*task*). Si no se encuentran busters de cola después de que se ejecutan todos los elementos, se devuelve INTERP_OK, lo que indica que la lectura anticipada puede continuar.

Cuando la lectura anticipada continúa después de la sincronización, task comienza a ejecutar de nuevo las operaciones read() del intérprete. Durante la siguiente operación de lectura, se comprueban los indicadores mencionados anteriormente y las variables correspondientes son establecidas (porque se acaba de ejecutar un synch(), los valores son ahora los actuales. Esto significa que el siguiente comando ya se ejecuta en el contexto de variables correctamente establecidas.

8.3.17.9. Ejecución de procedimientos

Los procedimientos de O-word complican un poco el manejo de los generadores de colas. Un destructor de colas puede encontrarse en algún lugar en un procedimiento anidado, lo que resulta en una llamada a procedimiento semiacabada cuando es devuelto INTERP_EXECUTE_FINISH. Task se asegura de sincronizar el modelo universal y continua el análisis y ejecución siempre que haya un procedimiento en ejecución (call_level> 0).

8.3.17.10. Cómo funciona el cambio de herramienta actualmente

Las acciones que suceden en LinuxCNC están poco involucradas, pero son necesarias para tener una idea general de lo que sucede actualmente antes de adaptar esos trabajos a sus propias necesidades.

Tenga en cuenta que la remapeado de un código existente desactiva completamente todo el procesamiento interno de ese código. Eso significa que más allá del comportamiento deseado - probablemente descrito a través de una Oword NGC o procedimiento Python, necesita replicar esas acciones internas con el intérprete, que juntos resultan en un reemplazo completo del código existente. El código prolog y epilog es el lugar para hacer esto.

Cómo se comunica la información de la herramienta. Varios procesos están *interesados* en la información de la herramienta: task y su intérprete, así como la interfaz de usuario. Además, el proceso *halui*.

La información de la herramienta se mantiene en la estructura *emcStatus*, que es compartida por todas las partes. Uno de sus campos es la matriz *toolTable*, que contiene la descripción cargada desde el archivo de tabla de herramientas (numero de herramienta, diámetro, frontangle, backangle y orientación para torno, e información de compensación de la herramienta).

La fuente autorizada y el único proceso en realidad que "configura" la información de herramienta es esa estructura en el proceso *iocontrol*. Todos los otros procesos solo consultan la estructura. El intérprete mantiene en realidad una copia local de la tabla de herramientas.

Para los curiosos, se puede acceder a la estructura actual de emcStatus mediante sentencias de Python. La percepción del intérprete de la herramienta cargada actualmente, por ejemplo, es accedida por:

```
;py,from interpreter import *
;py,print this.tool_table[0]
```

Para ver los campos en la estructura global de emcStatus, intente esto:

```
;py,from emctask import *
;py,print emcstat.io.tool.pocketPrepped
;py,print emcstat.io.tool.toolInSpindle
;py,print emcstat.io.tool.toolTable[0]
```

Debe tener LinuxCNC iniciado desde una ventana de terminal para ver los resultados.

8.3.17.11. Cómo funciona Tx (Prepare Tool)

Acción de intérprete en un comando Tx

Todo lo que hace el intérprete es evaluar el parámetro toolnumber, busca su ranura correspondiente, lo recuerda en la variable `selected_pocket` para más tarde, y pone en cola un comando canonico (SELECT_POCKET). Consulte *Interp::convert_tool_select* en *src/emc/rs274/interp_execute.cc*.

Acción de Task en SELECT_POCKET Cuando task maneja un SELECT_POCKET, envía un mensaje EMC_TOOL_PREPARE al proceso iocontrol, que maneja la mayoría de acciones relacionadas con herramientas en LinuxCNC.

En la implementación actual, task realmente espera a que iocontrol complete la operación de posicionamiento del cambiador, (que creo que no es necesario) - esto rechaza la idea de que la preparación del cambiador y la ejecución del código pueden proceder en paralelo.

Acción de iocontrol sobre EMC_TOOL_PREPARE Cuando iocontrol ve el comando select pocket, hace el HAL relacionado movimiento de pines - establece el pin "número de preparación de la herramienta" para indicar cuál La herramienta es la siguiente, levanta el pin de "preparación de la herramienta" y espera a que El pin "preparado por la herramienta" irá alto.

Cuando el cambiador responde con "herramienta preparada", se considera completa la fase de preparación y se señala a task tarea que continue. (de nuevo, creo que esta *espera* no es estrictamente necesaria)

Construyendo prolog y epilog para Tx. Vea las funciones de Python `prepare_prolog` y` prepare_epilog` en `configs/sim/axis/remap/toolchange/python/toolchange.py`.

8.3.17.12. Cómo funciona M6 (cambio de herramienta)

Necesita entender esto completamente antes de poder adaptarlo. Es muy relevante para escribir un controlador de prolog y epilog para un M6 reasignado. Reasignar códigos existentes significa que se deshabilita los pasos internos que se ejecutarian normalmente, y se deben replicar hasta donde sea necesario para sus propios fines.

Incluso si no está familiarizado con C, le sugiero que mire el código *Interp::convert_tool_change* en *src/emc/rs274/interp_convert.cc*.

Acción del intérprete en un comando M6

Cuando el intérprete ve un M6, éste:

1. verifica si un comando T ya ha sido ejecutado (prueba si *settings->selected_pocket* es >= 0). y si falla da el mensaje *necesito preparada -Txx- para de cambio de herramientas*.
2. verifica que la compensación del cortador está activa y si es así, falla con el mensaje *No se puede cambiar las herramientas con la compensación del radio de corte activa*.
3. detiene el husillo, excepto si se establece la opción "TOOL_CHANGE_WITH_SPINDLE_ON" en el ini.
4. genera un movimiento rápido de *Z up* si se establece la opción "TOOL_CHANGE_QUILL_UP" en el ini.
5. si se estableció TOOL_CHANGE_AT_G30:
   a. mueve los indizadores A, B y C si corresponde
   b. genera un movimiento rápido a la posición G30
6. ejecuta un comando canonico CHANGE_TOOL, con el ranura seleccionada como parámetro. CHANGE_TOOL puede:
   a. generar un movimiento rápido a TOOL_CHANGE_POSITION si así lo establece el ini
   b. encolar un mensaje EMC_TOOL_LOAD NML a task.
7. Configura los parámetros numerados 5400-5413 de acuerdo con la nueva herramienta
8. señala a task que deje de llamar al intérprete para leer adelantadamente devolviendo INTERP_EXECUTE_FINISH, ya que M6 es un destructor de colas.

Qué hace task cuando ve un comando CHANGE_TOOL Nuevamente, no mucho más que pasar la carga a iocontrol enviándole un mensaje EMC_TOOL_LOAD, y espera hasta que iocontrol haya realizado su labor.

ACCIÓN DE IOCONTROL TRAS EMC_TOOL_LOAD

1. Establece el pin "tool-change"
2. espera a que el pin "tool-changed" se active
3. cuando eso ha sucedido:
   a. borra "tool-change"
   b. coloca los pines "tool-prep-number" y "tool-prep-pocket" a cero.
   c. ejecuta la función *load_tool()* con la ranura como parámetro.

El último paso realmente establece las entradas de herramientas en la estructura *emcStatus*. La acción real tomada depende de si la opción ini RANDOM_TOOLCHANGER se estableció, pero al final del proceso *toolTable[0]* refleja la herramienta que se encuentra actualmente en el husillo.

Cuando eso ha sucedido:

1. iocontrol señala a task que puede seguir adelante
2. task le dice al intérprete que ejecute una operación *synch()*, para actualizar los cambios.

3. *synch()* del intérprete extrae toda la información necesaria del modelo universal, entre ellos la tabla de herramientas cambiada.

A partir de ahí, el intérprete tiene un conocimiento completo del modelo universal y continúa con la lectura adelantada.

Construyendo prolog y epilog para M6. Ver las funciones de Python `change_prolog` y` change_epilog` en `configs/sim/axis/remap/toolchange/python/toolchange.py`.

8.3.17.13. Cómo funciona M61 (Cambiar número de herramienta)

M61 requiere un parámetro `Q` no negativo (número de herramienta). Si es cero, significa *descargar herramienta*; si no, es *establecer el número de la herramienta actual en Q*.

Construyendo el reemplazo para M61 Un ejemplo de redefinición de Python para M61 se puede encontrar en la función `set_tool_number` en `configs/sim/axis/remap/toolchange/python/toolchange.py`.

8.3.18. Cambios

- el método para devolver mensajes de error y fallo solía ser *self.set_errormsg(texto)* seguido de *return INTERP_ERROR*. Esto ha sido reemplazado simplemente devolviendo una cadena desde un manejador de Python o subrutina oword. Esto establece el mensaje de error y aborta el programa. Anteriormente no había una forma clara de abortar una subrutina de palabra O de Python

8.3.19. Depuración

En la sección *[EMC]* del archivo ini, el parámetro *DEBUG* se puede cambiar para obtener varios niveles de mensajes de depuración cuando LinuxCNC se inicia desde un terminal.

```
Nivel de depuración, 0 significa que no hay mensajes. Vea src/emc/nml_intf/debugflags.h  ↩
    para otros.
DEBUG = 0x00000002 # configuración
DEBUG = 0x7FFFDEFF # no interp, oword
DEBUG = 0x00008000 # solo py
DEBUG = 0x0000E000 # py + remap + Oword
DEBUG = 0x0000C002 # py + remap + config
DEBUG = 0x0000C100 # py + remap + Intérprete
DEBUG = 0x0000C140 # py + remap + Intérprete + NML msgs
DEBUG = 0x0000C040 # py + remap + NML
DEBUG = 0x0003E100 # py + remap + Intérprete + oword + señales + namedparams
DEBUG = 0x10000000 # EMC_DEBUG_USER1 - declaraciones de rastreo
DEBUG = 0x20000000 # EMC_DEBUG_USER2 - captura en el depurador de Python
DEBUG = 0x10008000 # USER1, PYTHON
DEBUG = 0x30008000 # USER1, USER2, PYTHON # USER2 hará que la involuta intente conectarse a ↩
     pydev
DEBUG = 0x7FFFFFFF # Todos los mensajes de depuración
```

8.4. Componente moveoff

El componente Hal moveoff es un método exclusivo de Hal para implementar offsets. Ver la página de manual (*$ man moveoff*) para limitaciones y advertencias IMPORTANTES.

El componente moveoff se utiliza para offsets de posiciones articulares utilizando conexiones Hal personalizadas. Implementa una funcionalidad offset-mientras-programa-parado con conexiones apropiadas para los pines de entrada. Se soportan nueve articulaciones.

Los valores de los pines de offset de eje (offset-in-M) se aplican continuamente (respetando límites de valor, velocidad y aceleración) a los pines de salida (offset-current-M, pos-plusoffset-M, fb-minusoffset-M) cuando ambos pines (apply-offsets y move-enable) de entrada de habilitacion son TRUE. Las dos entradas habilitadoras estan asociadas internamente con un AND. Si el pin apply-offsets va a FALSE mientras se aplican los offsets, se activa un *pin de advertencia* y se emite un mensaje . El pin de advertencia permanece TRUE hasta que se eliminen los offsets o el pin apply-offsets vuelva a TRUE.

Normalmente, el pin move-enable está conectado a controles externos y el pin apply-offsets está conectado a halui.program.is-paused (solo para offsets mientras se está en pausa) o establecido en TRUE (para offsets aplicados continuamente).

Los offsets aplicados se *devuelven automáticamente* a cero (respetando los límites) cuando cualquiera de las entradas de habilitación está desactivada. El valor cero de tolerancia se especifica mediante el valor epsilon del pin de entrada.

Los puntos de referencia se registran cuando el componente moveoff está habilitado. Los puntos de referencia son gestionado con los pines waypoint-sample-secs y waypoint-threshold. Cuando el el pin backtrack-enable es TRUE, la ruta de retorno automático sigue los waypoints grabados. Cuando la memoria disponible para waypoints se agota, los offsets son congelados y el pin waypoint-limit va TRUE. Esta restricción se aplica independientemente del estado del pin backtrack-enable. Un pin de habilitación debe ser puesto a FALSE para permitir un retorno al original (posición sin offset).

Retroceder a través de puntos de ruta resulta en tasas de movimiento *más lentas* en la medida que los movimientos son punto a punto respetando la velocidad y la configuración de aceleración. Los pines de límite de velocidad y aceleración se pueden gestionar dinámicamente para controlar offsets en todo momento.

Cuando backtrack-enable es FALSE, el movimiento de retorno automático **NO** es coordinado; cada eje vuelve a cero a su propio ritmo. Si se desea una ruta controlada en esta condición, cada eje debe devolverse manualmente a cero antes de hacer FALSE un pin de habilitación.

Los pines waypoint-sample-secs, waypoint-threshold, y epsilon se evalúan solo cuando el componente está inactivo.

Se proporciona el pin de salida offsets-applied para indicar el estado actual a una GUI para que se pueda gestionar la reanudación del programa. Si el offset(s) no es cero cuando se desactiva el pin apply-offsets (por ejemplo, cuando se reanuda un programa tras offsets durante una pausa), los offsets se devuelven a cero (respetando límites) y se emite un mensaje de *Error*.

atención
Si los offsets están habilitados y aplicados y la máquina está apagada por cualquier razón, cualquier lógica Hal *externa* que maneje los pines habilitadores y las entradas offset-in-M es responsable de su estado cuando la máquina se vuelva a encender posteriormente.

LinuxCNC generalmente no conoce este medio de offsets solo-Hal ni está disponible en las pantallas de vista previa de las GUI. **No se proporciona protección** para movimientos con offset que exceden los límites soft administrados por LinuxCNC. Ya que los límites soft no se cumplen, un movimiento de offset puede encontrar límites fisicos (o un **CHOQUE** si no hay interruptores de límite). Se recomienda limitar la carrera con el uso de las entradas offset-min-M y offset-max-M. Disparar un límite fisico apagará la máquina - vea **Precaución** arriba.

Los valores de desplazamiento en M pueden establecerse con configuraciones inifile, controladas por una GUI, o gestionado por otros componentes y conexiones Hal. Los valores fijos pueden ser apropiado en casos simples donde la dirección y la cantidad de desplazamiento es bien definido pero se requiere un método de control para desactivar un pin de habilitación para devolver las compensaciones a cero. Las GUI pueden proporcionar medios para que los usuarios puedan establecer, incrementar, disminuir y acumular valores de compensación para cada eje y puedan establecer los valores de offset en M en cero antes de hacer FALSE un pin de habilitación.

Los valores predeterminados para accel, vel, min, max, epsilon, waypoint-sample-secs y waypoint-threshold pueden no ser adecuados para ninguna aplicación en particular. Este componente Hal desconoce los límites impuestos por LinuxCNC en otros lugares. Los usuarios deben probar el uso en una aplicación de simulador y comprender todos los riesgos antes de su uso en hardware.

Las configuraciones Sim que demuestran el componente y una gui (moveoff_gui) se encuentran en:

- configs/sim/axis/moveoff (axis-ui)
- configs/sim/touchy/ngcgui (touchy-ui)

8.4.1. Modificar una configuración existente

Se puede utilizar un halfile proporcionado por el sistema (LIB:hookup_moveoff.tcl) para adaptar una configuración existente para usar el componente moveoff. El archivo ini adicional de configuración admite el uso de una interfaz gráfica de usuario simple (moveoff_gui) para controlar los offsets.

Cuando el archivo HAL del sistema (LIB:hookup_moveoff.tcl) se especifica correctamente en un archivo ini de configuración, podrá:

1. Desconectar los pines joint.N.motor-pos-cmd and joint.N.motor-pos-fb
2. Cargar (loadrt) el componente moveoff (usando el nombre mv) con una personalidad configurada para acomodar todos los ejes identificados en el archivo ini
3. Agregar (addf) las funciones del componente moveoff en la secuencia requerida
4. Volver a conectar los pines joint.N.motor-pos-cmd y joint.N.motor-pos-fb para usar el componente
5. Establecer los parámetros operativos y los límites del componente moveoff para cada eje de acuerdo con la configuración adicional del archivo ini

Nota: La aplicación moveoff_gui admite configuraciones que utilizan módulos de cinemática conocidos con KINEMATICS_TYPE=KIN
Los módulos soportados incluyen: trivkins. Con cinemática de identidad, moveoff_gui asigna cada nombre de eje especificado con el parámetro de línea de comando *-axes axisnames* a la articulación correspondiente.

Modifique una configuración existente de la siguiente manera:

Asegúrese de que haya una entrada de archivo ini para [HAL]HALUI y cree una nueva entrada [HAL]HALFILE para LIB:hookup_moveo
La entrada para LIB:hookup_moveoff.tcl debe seguir todas las entradas HALFILE= para halfiles que conectan los pines para joint.N.motor-pos-cmd, joint.N.motor-pos-fb, y cualquier componente conectado a estos pines (pid y componentes de codificador en un sistema servo, por ejemplo).

```
[HAL]
HALUI = halui
MEDIO = existente_configuration_halfile_1
...
MEDIO = existente_configuration_halfile_n
HALFILE = LIB:hookup_moveoff.tcl
```

Agregue entradas de archivo ini para la configuración por eje para cada eje en uso (si una entrada no está definida, la entrada correspondiente de la sección [AXIS_n] será utilizada; si no se encuentra ninguna entrada, se utilizará lo predeterminado del componente moveoff. Nota: NO se recomienda la configuración usando los valores predeterminados o valores de sección [AXIS_n] para offset por eje.

```
[MOVEOFF_n]
MAX_LIMIT =
MIN_LIMIT =
MAX_VELOCITY =
MAX_ACCELERATION =
```

Agregar entradas de archivo ini para la configuración del componente moveoff (omitalo para usar los valores predeterminados):

```
[MOVEOFF]
EPSILON =
WAYPOINT_SAMPLE_SECS =
WAYPOINT_THRESHOLD =
```

moveoff_gui se usa para hacer conexiones requeridas adicionales y proporcionar una gui emergente para:

1. Proporcionar un botón de control para habilitar/deshabilitar las compensaciones
2. Proporcionar un botón de control para activar/desactivar el seguimiento

3. Proporcionar botones de control para aumentar/disminuir/Cero cada offset de eje
4. Mostrar el valor actual de offset de cada eje
5. Mostrar el estado de offset actual (deshabilitado, activo, eliminando, etc.)

Los botones de control provistos son opcionales dependiendo del estado del pin move-enable del componente. Se proporcionan tanto una pantalla como controles para habilitar offsets si el pin mv.move-enable NO está conectado cuando se inicia moveoff_gui. Para este caso, moveoff_gui administra el pin de habilitación de movimiento del componente (denominado mv.move-enable), así como los offsets (mv.move-offset-in-M) y la habilitacion de backtracking (mv.backtrack-enable)

Si el pin mv.move-enable está conectado cuando moveoff_gui se inicia, moveoff_gui proporcionará una pantalla pero NO controles. Este modo admite configuraciones que usan una rueda de selección u otros métodos de controlar las entradas de offsets y los pines de habilitación (mv.offset-in-M, mv.move-enable, mv.backtrack-enable).

Moveoff_gui realiza las conexiones necesarias para los pines del componente moveoff; mv.power_on y mv.apply-offsets. El pin mv.power_on está conectado al pin motion.motion-enabled (una nueva señal se crea automáticamente si es necesario). mv.apply-offsets está conectado a halui.program.is-paused o se establece en 1 dependiendo de la opción de línea de comando -mode [onpause | siempre]. Una nueva señal se crea automáticamente si es necesario.

Para usar moveoff_gui, agregue una entrada en la sección del archivo ini [APPLICATIONS] de la siguiente manera:

```
[APPLICATIONS]
# Nota: puede ser requerido un retraso (especificado en segundos) si las conexiones
# se hacen usando halfiles postgui ([HAL]POSTGUI_HALFILE=)
DELAY = 0
APP = moveoff_gui opción1 opción2 ...
```

Cuando el archivo hal LIB:hookup_moveoff.tcl se usa para cargar y conectar el componente moveoff, el pin mv.move-enable no se conectará y se utilizarán los controles locales proporcionados por moveoff_gui. Este es el método más simple para probar o demostrar el componente cuando se ha modificado una configuración ini existente.

Para habilitar controles externos mientras se usa la pantalla moveoff_gui para valores de offsets y estado, los halfiles que siguen a LIB:hookup_moveoff.tcl debe hacer conexiones adicionales. Por ejemplo, las demostraciones proporcionadas (configs/sim/axis/moveoff/*.ini) usan un halfile de sistema simple (llamado LIB:moveoff_external.hal) para conectar los pines mv.move-enable, mv.offset-in-M mv.bactrack-enable a señales:

```
[HAL]
HALUI = halui
...
HALFILE = LIB:hookup_moveoff.tcl
HALFILE = LIB:moveoff_external.hal
```

Las conexiones realizadas por LIB:moveoff_external.hal (para una configuración de tres ejes) son:

```
net external_enable mv.move-enable

net external_offset_0 mv.offset-in-0
net external_offset_1 mv.offset-in-1
net external_offset_2 mv.offset-in-2

net external_backtrack_en mv.backtrack-enable
```

Estas señales (external_enable, external_offset_M, external_backtrack_en) pueden ser administradas por HALFILES posteriores (incluidos POSTGUI_HALFILEs) para proporcionar control personalizado del componente mientras se usa la pantalla moveoff_gui para valores de offset y estado actuales.

Moveoff_gui se configura con opciones de línea de comando. Para detalles sobre la operación de moveoff_gui, vea la página del manual:

```
$ man moveoff_gui
```

Para obtener una breve lista de las opciones de línea de comando para moveoff_gui, use el comando opción de ayuda de línea:

```
$ moveoff_gui --help

Uso:
moveoff_gui [Opciones]

Opciones:
~~~~[--ayuda | -? | -- -h] (Este texto)

~~~~[-modo [enpausa | siempre]] (predeterminado: enpausa)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(enpausa: muestra la gui cuando el programa está en pausa ↩
    )
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(siempre: muestra gui siempre)

~~~~[-axes axisnames]        (predeterminado: xyz (sin espacios))
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(letras del conjunto de: x y z a b c u v w)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(ejemplo: -axes z)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(ejemplo: -axes xz)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(ejemplo: -axes xyz)
~~~~[-inc incrementvalue]    (predeterminado: 0.001 0.01 0.10 1.0)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(especifique uno por cada (hasta 4))
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(ejemplo: -inc 0.001 -inc 0.01 -inc 0.1)
~~~~[entero de tamaño]       (predeterminado: 14
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(El tamaño general emergente de la gui se basa en el tamaño  ↩
    de la fuente)
~~~~[-loc center | +x+y]     (predeterminado: centro)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(ejemplo: -loc +10+200)
~~~~[-autoresume]            (predeterminado: no utilizado)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(reanudar el programa cuando move-enable sea false)
~~~~[-delay delay_secs]      (predeterminado: 5 (retraso de reanudación))

Opciones para casos especiales:
~~~~[-noentry]               (predeterminado: no utilizado)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(no cree widgets de entrada)
~~~~[-no_resume_inhibit]     (predeterminado: no utilizado)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(no use un pin de reanudar inhibición)
~~~~[-no_pause_requirement]  (predeterminado: no utilizado)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(sin verificación de halui.program.is-paused)
~~~~[-no_cancel_autoresume]  (predeterminado: no utilizado)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(útil para retractaciones con simple)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(control externo                      )
~~~~[-no_display]            (predeterminado: no utilizado)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(Usar cuando tanto los controles externos como las pantallas)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~(están en uso (ver Nota)))

Nota: Si el pin mv.move-enable está conectado cuando
~~~~~~moveoff_gui se inicia, se requieren controles externos y solo
~~~~~~se proporciona pantallas.
```

8.5. Configuración de steppers

8.5.1. Introducción

La forma preferida de configurar una máquina de pasos estándar es con el Asistente de Configuración de Pasos. Ver el capítulo Asistente de Configuración Stepconf .

Este capítulo describe algunas de las configuraciones más comunes para sistemas basado en pasos. Estos sistemas utilizan motores paso a paso con controladores que aceptan señales de paso y dirección.

Es una de las configuraciones más simples, porque los motores funcionan en bucle abierto (la retroalimentación no proviene de los motores), pero el sistema debe ser configurado correctamente para que los motores no se detengan o pierdan pasos.

La mayor parte de este capítulo se basa en una configuración de muestra publicada junto con LinuxCNC. La configuración se llama stepper_inch, y se puede encontrar ejecutando el Selector de configuración.

8.5.2. Velocidad de paso máxima

Con la generación de pasos de software, la tasa de paso máxima es un paso por cada dos BASE_PERIODs para salida de paso y dirección. La velocidad de pasos máxima solicitada es el producto *MAX_VELOCITY* del eje y su *INPUT_SCALE*. Si la tasa de pasos solicitada no es alcanzable, ocurriran los siguientes errores, particularmente durante movimientos jog y movimientos G0.

Si su controlador paso a paso puede aceptar entrada en cuadratura, use este modo. Con una señal de cuadratura, es posible un paso para cada BASE_PERIOD, duplicando la velocidad de paso máxima.

Los otros remedios son disminuir uno o más de: BASE_PERIOD (establecerlo demasiado bajo hará que la máquina deje de responder o incluso se bloquee), INPUT_SCALE (si puede seleccionar diferentes tamaños de paso en su controlador paso a paso, cambie las relaciones de las poleas o el paso del tornillo de avance), o MAX_VELOCITY y STEPGEN_MAXVEL.

Si no hay una combinación válida de BASE_PERIOD, INPUT_SCALE y MAX_VELOCITY aceptable, entonces considere usar la generación de pasos de hardware (como con el controlador paso a paso universal compatible con LinuxCNC, las tarjetas Mesa y otros.)

8.5.3. Pinout

Uno de los principales defectos en EMC fué que no se podía especificar el pinout sin recompilar el código fuente. LinuxCNC es mucho más flexible, y ahora (gracias a la Capa de Abstracción de Hardware o HAL) se puede especificar fácilmente qué señal va a donde. Vea el << cha:basic-hal-reference, Tutorial básico de HAL>> para más información sobre HAL.

Tal como se describe en la Introducción y el tutorial de HAL, dentro del HAL tenemos señales, pines y parámetros.

NOTA: Estamos centrandonos en un solo eje para ser concisos; todos los demás son similares.

Lo relevante para nuestro pinout es:

```
señales: Xstep, Xdir y Xen
pines: parport.0.pin-XX-out y parport.0.pin-XX-in
```

Dependiendo de lo que haya elegido en su archivo .ini, está utilizando standard_pinout.hal o xylotex_pinout.hal. Estos son dos archivos que le indican a HAL cómo vincular las diferentes señales y pines. Después investigaremos standard_pinout.hal.

8.5.3.1. Pinout estándar HAL

Este archivo contiene varios comandos HAL, y generalmente se ve así:

```
# archivo de configuración de pinout estándar para steppers de 3 ejes
# usando un parport para E/S
#
# primero carga el controlador parport
loadrt hal_parport cfg = "0x0378"
#
# a continuación, conecta las funciones de parport a hilos
# primero, leer entradas
addf parport.0.read base-thread 1
# lo ultimo, escribir salidas
addf parport.0.write base-thread -1
#
# finalmente, conectar los pines físicos a las señales
net Xstep => parport.0.pin-03-out
net Xdir => parport.0.pin-02-out
```

```
net Ystep => parport.0.pin-05-out
net Ydir => parport.0.pin-04-out
net Zstep => parport.0.pin-07-out
net Zdir => parport.0.pin-06-out

# crear una señal para el loopback de estop
net estop-loop iocontrol.0.user-enable-out iocontrol.0.emc-enable-in

# crear señales para el loopback de carga de herramientas
net tool-prep-loop iocontrol.0.tool-prepare iocontrol.0.tool-prepare
net tool-change-loop iocontrol.0.tool-change iocontrol.0.tool-changed

# conectar el pin del controlador de movimiento "spindle on" a un pin físico
net spindle-on spindle.0.on => parport.0.pin-09-out

###
### Puede usar algo como esto para habilitar unidades chopper cuando la máquina está  ↩
    ENCENDIDA
### la señal Xen se define en core_stepper.hal
###

# net Xen => parport.0.pin-01-out

###
### Si desea activo bajo para este pin, inviértalo así:
###

# setp parport.0.pin-01-out-invert 1

###
### Un interruptor home de muestra en el eje X (eje 0). Hacer una señal
### y vincular el pin parport entrante a la señal, luego vincular la señal
### al pin de entrada del interruptor home del eje 0 de LinuxCNC
###

# net Xhome parport.0.pin-10-in => joint.0.home-sw-in

###
### ¿comparte todos los interruptores home en un pin del puerto paralelo?
### de acuerdo, conecte la misma señal a todos los ejes, pero asegúrese de
### establece HOME_IS_SHARED y HOME_SEQUENCE en el archivo ini.
###

# net homeswitches <= parport.0.pin-10-in
# net homeswitches => joint.0.home-sw-in
# net homeswitches => joint.1.home-sw-in
# net homeswitches => joint.2.home-sw-in

###
### Muestra de interruptores de límite separados en el eje X (eje 0)
###

# net X-neg-limit parport.0.pin-11-in => joint.0.neg-lim-sw-in
# net X-pos-limit parport.0.pin-12-in => joint.0.pos-lim-sw-in

###
### Al igual que en el ejemplo de los interruptores home compartidos, puede conectar juntos
### finales de carrera. Tenga cuidado si activa uno, LinuxCNC se detendrá pero no puede  ↩
    decirle
### qué interruptor/eje se ha disparado. Tenga cuidado al reiniciar despues de esto.
###
```

```
# net Xlimits parport.0.pin-13-in => joint.0.neg-lim-sw-in joint.0.pos-lim-sw-in
```

Las líneas que comienzan con # son comentarios, y su único propósito es guiar al lector a través del archivo.

8.5.3.2. Descripción general

Hay un par de operaciones que se ejecutan cuando se ejecuta/interpreta standard_pinout.hal:

- El controlador Parport se carga (ver el Capítulo Parport para más detalles)
- Las funciones de lectura y escritura del controlador parport se asignan al hilo base [1]
- Las señales de paso y dirección para los ejes X, Y, Z se vinculan a los pines en el parport
- Se conectan más señales de E /S (loopback de estop, loopback del cambiador de herramientas)
- Se define una señal de husillo y se vincula a un pin parport

8.5.3.3. Cambiar standard_pinout.hal

Si desea cambiar el archivo standard_pinout.hal, todo lo que necesita es un editor de texto. Abra el archivo y localice las partes que desea cambiar.

Si desea, por ejemplo, cambiar los pines de señales para el eje X Step y Dirección, todo lo que necesita hacer es cambiar el número en el nombre *parport.0.pin-XX-out*:

```
net Xstep parport.0.pin-03-out
net Xdir parport.0.pin-02-out
```

se puede cambiar a:

```
net Xstep parport.0.pin-02-out
net Xdir parport.0.pin-03-out
```

o básicamente cualquier otro pin *out* que interese.

Sugerencia: asegúrese de no tener más de una señal conectada al mismo pin.

8.5.3.4. Cambio de polaridad de una señal

Si el hardware externo espera una señal de "activo bajo", configure el parámetro *-invert* correspondiente. Por ejemplo, para invertir la señal de control del husillo:

```
setp parport.0.pin-09-invert TRUE
```

8.5.3.5. Agregar control de velocidad PWM al husillo

Si su husillo puede ser controlado por una señal PWM, use el componente *pwmgen* para crear la señal:

```
loadrt pwmgen output_type = 0
addf pwmgen.update servo-thread
addf pwmgen.make-pulses base-thread
net spindle-speed-cmd spindle.0.speed-out => pwmgen.0.value
net spindle-on spindle.0.on => pwmgen.0.enable
net spindle-pwm pwmgen.0.pwm => parport.0.pin-09-out
setp pwmgen.0.scale 1800  # Cambia a la velocidad máxima de su husillo en RPM
```

Esto supone que la respuesta del controlador del husillo a PWM es simple: 0 % PWM da 0 RPM, 10 % PWM da 180 RPM, etc. Si hay un mínimo PWM que se requiere para que el husillo gire, siga el ejemplo en configuración de muestra *nist-lathe* para usar un componente *scale*.

[1] el subproceso más rápido en la configuración de LinuxCNC, generalmente su código se ejecuta cada pocas decenas de microsegundos

8.5.3.6. Agregar una señal de habilitación

Algunos amplificadores (controladores) requieren una señal de habilitación antes de aceptar ordenes de movimiento de los motores. Por esta razón ya hay señales definidas llamadas *Xen*, *Yen*, *Zen*.

Para conectarlas use el siguiente ejemplo:

```
net Xen parport.0.pin-08-out
```

Puede tener un solo pin que habilite todas las unidades o varios, dependiendo de la configuración que tenga. Tenga en cuenta, sin embargo, que generalmente cuando falla un eje, todas las demás unidades también se desactivarán, por lo que tener una sola señal/pin de habilitación para todas las unidades es una práctica común.

8.5.3.7. Botón externo ESTOP

El archivo standard_pinout.hal supone que no hay un botón ESTOP externo. Para más información sobre un E-Stop externo, consulte la página del manual estop_latch.

Capítulo 9

Paneles de Control

Capítulo 10

Interfaces de Usuario

10.1. Ejemplos Halui

Para que funcionen los ejemplos de Halui, debe agregar la siguiente línea a la sección [HAL] del archivo ini.

```
HALUI = halui
```

10.1.1. Arranque remoto

Para conectar un botón de inicio de programa remoto a LinuxCNC, utilice los pines `halui.program.run` y `halui.mode.auto`. Debe asegurarse de que se puede ejecutar utilizando el pin `halui.mode.is-auto`. Esto se hace con un componente `and2`. La siguiente figura muestra cómo son las conexiones. Cuando se presiona el botón de ejecución remota, se conecta tanto a `halui.mode.auto` como a `and2.0.in0`. Si está habilitado el modo automático, el pin `halui.mode.is-auto` será true. Si ambas entradas al componente `and2.0` son true, `and2.0.out` se activara e iniciará el programa a traves de `halui.program.run`.

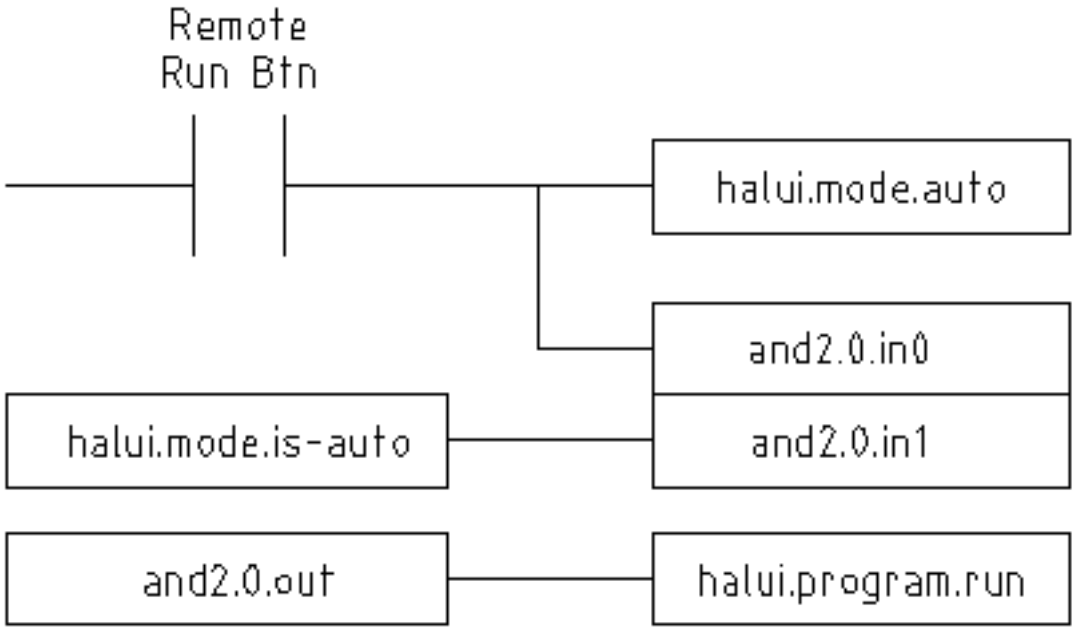


Figura 10.1: Ejemplo de arranque remoto

Los comandos hal necesarios para lograr lo anterior son:

```
net program-start-btn halui.mode.auto and2.0.in0 <= <su pin de entrada>
net program-run-ok and2.0.in1 <= halui.mode.is-auto
net remote-program-run halui.program.run <= and2.0.out
```

Observe que en la primera línea hay dos pines de lectura; esto se puede dividir en dos líneas como estas:

```
net program-start-btn halui.mode.auto <= <su pin de entrada>
net program-start-btn and2.0.in0
```

10.1.2. Pausar y Reanudar

Este ejemplo fue desarrollado para permitir que LinuxCNC mueva un eje rotativo a una señal de una máquina externa. La coordinación entre los dos sistemas será proporcionada por dos componentes de Halui:

- halui.program.is-paused
- halui.program.resume

En su archivo hal personalizado, agregue las dos líneas siguientes, que se conectarán a su E/S para activar la pausa del programa o para reanudar cuando el sistema externo requiera que LinuxCNC continúe.

```
net ispaused halui.program.is-paused => "su pin de salida"
net resume halui.program.resume <= "su pin de entrada"
```

Sus pines de entrada y salida están conectados a los pines del otro controlador. Pueden ser pines de puerto paralelo o cualquier otro pin de E/S al que tenga acceso.

Este sistema funciona de la siguiente manera. Cuando se encuentra un M0 en el código G, la señal `halui.program.is-paused` se hace true. Esto activa su pin de salida para que el controlador externo sepa que LinuxCNC está en pausa.

Para reanudar el programa gcode en LinuxCNC, cuando el controlador externo esté listo, hará que su salida sea true. Esto indicará LinuxCNC que debería reanudar la ejecución de Gcode.

Dificultades en el tiempo

- La señal de retorno de entrada "resume" no debe ser más larga que el tiempo requerido para obtener el código g corriendo de nuevo.
- La salida "is-paused" ya no debería estar activa para cuando termina la señal de "resume".

Estos problemas de tiempo podrían evitarse utilizando ClassicLadder para activar la salida "is-paused" a través de un temporizador monoestable para entregar un pulso de salida estrecho. El pulso "reanudar" también podría recibirse a través de un temporizador monoestable.

10.2. Interfaz de Python

Esta documentación describe el módulo python *linuxcnc*, que proporciona una API Python para LinuxCNC.

10.2.1. El módulo linuxcnc Python

Las interfaces de usuario controlan la actividad de LinuxCNC enviando mensajes NML al controlador de tareas LinuxCNC y supervisan los resultados observando la estructura de estado de LinuxCNC, así como el canal de informe de errores.

El acceso programático a NML es a través de una API de C ++; sin embargo, la mayoría partes importantes de la interfaz NML para LinuxCNC también están disponibles para programas de Python a través del módulo `linuxcnc`.

Más allá de la interfaz NML para los canales de comando, estado y error, el módulo `linuxcnc` también contiene:

- soporte para leer valores de archivos ini

10.2.2. Patron de uso de la interfaz NML LinuxCNC

El patrón general para el uso de `linuxcnc` es, en lineas generales, así:

- importar el módulo `linuxcnc`
- establecer conexiones a los canales NML de comandos, estado y errores según sea necesario
- sondear el canal de estado, ya sea periódicamente o según sea necesario
- antes de enviar un comando, determinar desde el estado si está en uno correcto para hacerlo (por ejemplo, no tiene sentido enviar un comando *Ejecutar* si la tarea está en el estado ESTOP o el intérprete no está inactivo)
- envíar el comando utilizando uno de los métodos del canal de comandos `linuxcnc`

Para recuperar mensajes del canal de errores, sondee el canal de errores periódicamente y procese los mensajes recuperados.

- sondear el canal de errores, ya sea periódicamente o según sea necesario
- imprimir cualquier mensaje de error y explore el código de excepción

`linuxcnc` también define el tipo de excepción Python `error` para admitir informes de errores.

10.2.3. Lectura del Estado de LinuxCNC

Aquí hay un fragmento de Python para explorar el contenido del objeto `linuxcnc.stat` que contiene más de 80 valores (correr mientras linuxcnc se está ejecutando para valores típicos):

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import sys
import linuxcnc
try:
    s = linuxcnc.stat() # crea una conexión al canal de estado
~~~~s.poll() # obtener valores actuales
except linuxcnc.error, detail:
    print "error", detail
    sys.exit(1)
for x in dir(s):
    if not x.startswith("_"):
        print x, getattr(s,x)
```

Linuxcnc usa la ruta de compilacion predeterminada para el archivo de configuración NML a menos que se sobreescriba; vea lectura valores de archivo ini para un ejemplo.

10.2.3.1. Atributos en linuxcnc.stat

acceleration
(devuelve float) - aceleración predeterminada, refleja la entrada ini [TRAJ]DEFAULT_ACCELERATION.

active_queue
(devuelve entero) - Número de mezclas en motion.

actual_position
(devuelve tupla de floats) - posición de trayectoria actual, (x y z a b c u v w), en unidades máquina.

adaptive_feed_enabled
(devuelve boolean) - estado de anulación de avance adaptativo (0/1).

ain
(devuelve tupla de floats) - valor actual de los pines de entrada analógica.

angular_units
(devuelve float) - unidades angulares de máquina para grados, refleja el valor ini [TRAJ]ANGULAR_UNITS.

aout
(devuelve tupla de floats) - valor actual de los pines de salida analógica.

axes
(devuelve entero) - número de ejes derivado del valor ini [TRAJ]COORDINATES.

axis
(devuelve tupla de diccionarios) - reflejando los valores actuales del eje. Ver El diccionario del eje.

axis_mask
(devuelve entero) - máscara de eje disponible según lo definido por [TRAJ]COORDINATES en el archivo ini. Devuelve la suma de los ejes X = 1, Y = 2, Z = 4, A = 8, B = 16, C = 32, U = 64, V = 128, W = 256.

block_delete
(devuelve boolean) - el estado actual de eliminar bloque.

command
(devuelve cadena) - comando actualmente en ejecución.

current_line
(devuelve entero) - línea actualmente en ejecución.

current_vel
(devuelve float) - velocidad actual en unidades de usuario por segundo.

cycle_time
(devuelve float) - período de hilo

debug
(devuelve entero) - bandera de depuración desde el archivo ini.

delay_left
(devuelve float) - tiempo restante en el comando dwell (G4), segundos.

din
(devuelve tupla de enteros) - valor actual de los pines de entrada digital.

distance_to_go
(devuelve float) - distancia restante del movimiento actual, según lo informado por el planificador de trayectoria.

dout
(devuelve tupla de enteros) - valor actual de los pines de salida digital.

dtg
(devuelve tupla de floats) - distancia restante del movimiento actual para cada eje, según lo informado por el planificador de trayectoria.

echo_serial_number
(devuelve entero) - El número de serie del último comando completado enviado por una UI a task. Todos los comandos llevan un número de serie. Una vez que el comando ha sido ejecutado, su número de serie se refleja en `echo_serial_number`.

enabled
(devuelve boolean) - bandera de habilitacion de planificador de trayectoria.

estop
(devuelve entero) - Devuelve STATE_ESTOP o no.

exec_state
(devuelve entero) - estado de ejecución task. Uno de EXEC_ERROR, EXEC_DONE, EXEC_WAITING_FOR_MOTION, EXEC_WAITING_FOR_MOTION_QUEUE, EXEC_WAITING_FOR_IO, EXEC_WAITING_FOR_MOTION_AND_IO, EXEC_WAITING_FOR_DELAY, EXEC_WAITING_FOR_SYSTEM_CMD, EXEC_WAITING_FOR_SPINDLE_ORIENTED.

feed_hold_enabled
(devuelve boolean) - habilitar la bandera para retención de alimentación.

feed_override_enabled
(devuelve boolean) - habilitar bandera para ajuste de alimentación.

feedrate
(devuelve float) - ajuste de avance actual, 1.0 = 100%.

file
(devuelve cadena) - nombre de archivo gcode cargado actualmente, con ruta.

flood
(devuelve entero) - Estado de inundación, FLOOD_OFF o FLOOD_ON.

g5x_index
(devuelve entero) - sistema de coordenadas actualmente activo, G54 = 1, G55 = 2, etc.

g5x_offset
(devuelve tupla de floats) - desplazamiento del sistema de coordenadas actualmente activo.

g92_offset
(devuelve tupla de floats) - pose del offset g92 actual.

gcodes
(devuelve tupla de enteros) - Códigos G activos para cada grupo modal. Constantes de código G G_0, G_1, G_2, G_3, G_4, G_5, G_5_1, G_5_2, G_5_3, G_7, G_8, G_100, G_17, G_17_1, G_18, G_18_1, G_19, G_19_1, G_20, G_21, G_28, G_28_1, G_30, G_30_1, G_33, G_33_1, G_38_2, G_38_3, G_38_4, G_38_5, G_40, G_41, G_41_1, G_42, G_42_1, G_43, G_43_1, G_43_2, G_49, G_50, G_51, G_53, G_54, G_55, G_56, G_57, G_58, G_59, G_59_1, G_59_2, G_59_3, G_61, G_61_1, G_64, G_73, G_76, G_80, G_81, G_82, G_83, G_84, G_85, G_86, G_87, G_88, G_89, G_90, G_90_1, G_91, G_91_1, G_92, G_92_1, G_92_2, G_92_3, G_93, G_94, G_95, G_96, G_97, G_98, G_99

homed
(devuelve tupla de enteros) - articulaciones actualmente con/sin home, 0 = sin home, 1 = con home.

id
(devuelve entero) - ID de movimiento actualmente en ejecución.

inpos
(devuelve boolean) - bandera machine-in-position.

input_timeout
(devuelve boolean) - bandera de temporizador M66 en progreso.

interp_state
(devuelve entero) - estado actual del intérprete RS274NGC. Uno de INTERP_IDLE, INTERP_READING, INTERP_PAUSED, INTERP_WAITING.

interpreter_errcode
(devuelve entero) - código de retorno actual del intérprete RS274NGC. Uno de INTERP_OK, INTERP_EXIT, INTERP_EXECUT INTERP_ENDFILE, INTERP_FILE_NOT_OPEN, INTERP_ERROR. ver src/emc/nml_intf/interp_return.hh

joint
(devuelve tupla de diccionarios) - refleja los valores de articulación actuales. Ver El diccionario de articulaciones.

joint_actual_position
(devuelve tupla de floats) - Posiciones articulares reales.

joint_position
(devuelve tupla de floats) - Posiciones articulares deseadas.

joints
(devuelve entero) - Número de articulaciones. Refleja [KINS]JOINTS del ini.

kinematics_type
(devuelve entero) - El tipo de cinemática. Uno de:

- KINEMATICS_IDENTITY
- KINEMATICS_FORWARD_ONLY
- KINEMATICS_INVERSE_ONLY
- KINEMATICS_BOTH

limit
(devuelve tupla de enteros) - Máscaras de límites de eje. minHardLimit = 1, maxHardLimit = 2, minSoftLimit = 4, maxSoftLimit = 8.

linear_units
(devuelve float) - unidades lineales de máquina por mm, refleja valor ini [TRAJ]LINEAR_UNITS.

lube
(devuelve entero) - bandera de "lubricación ON".

lube_level
(devuelve entero) - refleja *iocontrol.0.lube_level*.

max_acceleration
(devuelve float) - máxima aceleración, refleja [TRAJ]MAX_ACCELERATION.

max_velocity
(devuelve float) - velocidad máxima refleja [TRAJ]MAX_VELOCITY.

mcodes
(devuelve tupla de 10 enteros) - códigos M actualmente activos.

mist
(devuelve entero) - Estado de la niebla, ya sea MIST_OFF o MIST_ON

motion_line
(devuelve entero) - número de línea fuente del movimiento que se está ejecutando actualmente. Relación con `id` no aclarada.

motion_mode
(devuelve entero) - Este es el modo del controlador de movimiento. Uno de TRAJ_MODE_COORD, TRAJ_MODE_FREE, TRAJ_MODE_TELEOP.

motion_type
(devuelve entero) - El tipo de movimiento que se está ejecutando actualmente. Uno de: * MOTION_TYPE_TRAVERSE * MOTION_TYPE_FEED * MOTION_TYPE_ARC * MOTION_TYPE_TOOLCHANGE * MOTION_TYPE_PROBING * MOTION_TYPE_INDEXROTARY * 0 si no hay movimiento en este momento.

optional_stop
(devuelve entero) - bandera de stop opcional.

paused
(devuelve boolean) - bandera de `movimiento en pausa`.

pocket_prepped
(devuelve entero) - comando Tx completado, la ranura está preparada. -1 si la ranura no está preparada.

***poll() ***
- *(función incorporada)* Método para actualizar los atributos del estado actual.

position
(devuelve tupla de floats) - posición en trayectoria

probe_tripped
(devuelve boolean) - bandera, True si la sonda se ha disparado (con latch)

probe_val
(devuelve entero) - refleja el valor del pin `motion.probe-input`.

probed_position
(devuelve la tupla de floats) - posición donde se disparó la sonda.

sondeo
(devuelve booleano) - bandera, True si hay una operación de sonda en progreso.

program_units
(devuelve entero) - uno de CANON_UNITS_INCHES=1 , CANON_UNITS_MM=2, CANON_UNITS_CM=3

queue
(devuelve entero) - tamaño actual de la cola del planificador de trayectoria.

queue_full
(devuelve boolean) - la cola del planificador de trayectoria está llena.

rapidrate
(devuelve float) - escala de ajustes de rápidos.

read_line
(devuelve entero) - línea que el intérprete RS274NGC está leyendo actualmente.

rotación_xy
(devuelve float) - ángulo de rotación XY actual alrededor del eje Z.

settings
(devuelve tupla de floats) - configuración actual del intérprete. settings[0] = número de secuencia, settings[1] = velocidad de avance, settings[2] = velocidad

spindle
(devuelve tupla de diccionarios) - devuelve el estado actual del husillo ver <sec:the-spindle-dictionary, El diccionario spindle>>

spindles
(devuelve entero) - Número de husillos. Refleja el valor ini [TRAJ]SPINDLES.

state
(devuelve entero) - estado actual de ejecución del comando. Uno de RCS_DONE, RCS_EXEC, RCS_ERROR.

task_mode
(devuelve entero) - modo de tarea actual. uno de MODE_MDI, MODE_AUTO, MODE_MANUAL.

task_paused
(devuelve entero) - bandera Task pausado.

task_state
(devuelve entero) - estado actual de la tarea. uno de STATE_ESTOP, STATE_ESTOP_RESET, STATE_ON, STATE_OFF.

tool_in_spindle
(devuelve entero) - Número de herramienta actual.

tool_offset
(devuelve tupla de floats) - valores de offsets de la herramienta actual.

tool_table
(devuelve tupla de tool_results) - lista de entradas de herramientas. Cada entrada es una secuencia de los siguientes campos: id, xoffset, yoffset, zoffset, aoffset, boffset, coffset, uoffset, voffset, woffset, diameter, frontangle, backangle, orientation. La identificación y orientación son enteros y el resto son floats.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import linuxcnc
s = linuxcnc.stat()
s.poll()
# para encontrar la información de la herramienta cargada en el índice 0 de la tabla de ↩
    herramientas
if s.tool_table[0].id != 0: # se carga una herramienta
    print s.tool_table[0].zoffset
else:
    print "sin herramienta cargada"
```

- velocity*:: *(devuelve float)* - Esta propiedad está definida, pero no tiene una interpretación útil.

10.2.3.2. El diccionario `axis`

La configuración de ejes y los valores de estado están disponibles a través de una lista de diccionarios por eje. Aquí hay un ejemplo de cómo acceder a un atributo de un eje particular:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import linuxcnc
s = linuxcnc.stat()
s.poll()
print "Joint 1 homed: ", s.joint[1]["homed"]
```

Tenga en cuenta que muchas propiedades que anteriormente estaban en el diccionario `axis` están ahora en el diccionario `joint`, porque en máquinas con cinemáticas no triviales estos elementos (como el backlash) no son propiedades de un eje.

max_position_limit
(devuelve float) - límite máximo (límite soft) para el movimiento del eje, en unidades máquina del parametro units.configuration, refleja [JOINT_n]MAX_LIMIT.

min_position_limit
(devuelve float) - límite mínimo (límite soft) para el movimiento del eje, en unidades máquina del parametro units.configuration, refleja [JOINT_n]MIN_LIMIT.

velocity
(devuelve float) - velocidad actual

10.2.3.3. El diccionario `joint`

Para cada articulación, están disponibles las siguientes claves de diccionario:

backlash
(devuelve float) - backlash en unidades máquina del parametro units.configuration, refleja [JOINT_n]BACKLASH.

enabled
(devuelve entero) - no cero significa habilitado.

fault
(devuelve entero) - no cero significa fallo del amplificador del eje.

ferror_current
(devuelve float) - error de seguimiento actual

ferror_highmark
(devuelve float) - magnitud del error de seguimiento máximo.

homed
(devuelve entero) - distintos de cero han sido homed.

homing
(devuelve entero) - distinto de cero significa homing en progreso.

inpos
(devuelve entero) - no cero significa en posición.

input
(devuelve float) - entrada de posición actual.

jointType
(devuelve entero) - parámetro del tipo de configuración del eje, refleja [JOINT_n]TYPE. LINEAL=1, ANGULAR=2. Ver configuración ini de articulacion para más detalles.

max_ferror
(devuelve float) - Máximo error de seguimiento. parámetro de configuración, refleja [JOINT_n]FERROR.

max_hard_limit
(devuelve entero) - distinto de cero significa que se ha excedido el límite hard máximo.

max_position_limit
(devuelve float) - Límite máximo (límite soft) para el movimiento articular, en unidades de máquina. parámetro de configuración, refleja [JOINT_n]MAX_LIMIT.

***max_soft_limit ***
distinto de cero significa que se ha excedido `max_position_limit`, int

min_ferror
(devuelve float) - parámetro de configuración, refleja [JOINT_n]MIN_FERROR.

min_hard_limit
(devuelve entero) - distinto de cero significa que se excedió el límite hard mínimo.

min_position_limit
(devuelve float) - límite mínimo (límite soft) para el movimiento articular, en unidades de máquina. parámetro de configuración, refleja [JOINT_n]MIN_LIMIT.

min_soft_limit
(devuelve entero) - distinto de cero significa que se ha excedido `min_position_limit`.

output
(devuelve float) - posición de salida ordenada.

override_limits
(devuelve entero) - no cero significa que los límites se han ajustado.

units
(devuelve float) - unidades de articulacion por mm, o por grado para juntas angulares.
(las unidades de articulacion son las mismas que las unidades d máquina, a menos que se establezca lo contrario por el parámetro de configuración [JOINT_n]UNITS)

velocity
(devuelve float) - velocidad actual

10.2.4. El diccionario `spindle`

brake
(devuelve entero) - valor de la bandera del freno del husillo.

direction
(devuelve entero) - dirección rotacional del huso. adelante = 1, atrás = -1.

enabled
(devuelve entero) - valor de la bandera de husillo habilitado.

homed
(no implementado actualmente)

increasing
(devuelve entero) - poco claro.

orient_fault
(devuelve entero)

orient_state
(devuelve entero)

override
(devuelve float) - Escala de ajuste de velocidad del husillo.

override_enabled
(devuelve boolean) - valor de la bandera habilitacion para ajustar el husillo.

speed
(devuelve float) - valor de velocidad del husillo, rpm,> 0: en sentido horario, <0: en sentido anti-horario.

10.2.5. Preparacion para enviar comandos

Siempre se pueden enviar algunos comandos, independientemente del modo y el estado. Por ejemplo, siempre se puede llamar al método `linuxcnc.command.abort()`.

Otros comandos pueden enviarse solo en el estado apropiado, y probarlos puede ser un poco complicado. Por ejemplo, un comando MDI solo se puede enviar si:

- ESTOP no se ha activado, y
- la máquina está encendida y
- los ejes tienen home y
- el intérprete no está funcionando y
- el modo está configurado en `modo MDI`

entonces una prueba apropiada antes de enviar un comando MDI a través de `linuxcnc.command.mdi()` podría ser:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import linuxcnc
s = linuxcnc.stat()
c = linuxcnc.command()

def ok_for_mdi():
    s.poll()
    return not s.estop and s.enabled and (s.homed.count(1) == s.joints) and (s.interp_state ↩
        == linuxcnc.INTERP_IDLE)
```

```
if ok_for_mdi():
    c.mode(linuxcnc.MODE_MDI)
    c.wait_complete() # espera hasta que se ejecute el cambio de modo
    c.mdi("G0 X10 Y20 Z30")
```

10.2.6. Enviar comandos a través de `linuxcnc.command`

Antes de enviar un comando, inicialice un canal de comando así:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import linuxcnc
c = linuxcnc.command()

# Ejemplos de uso para algunos de los comandos enumerados a continuación:
c.abort()

c.auto(linuxcnc.AUTO_RUN, program_start_line)
c.auto(linuxcnc.AUTO_STEP)
c.auto(linuxcnc.AUTO_PAUSE)
c.auto(linuxcnc.AUTO_RESUME)

c.brake(linuxcnc.BRAKE_ENGAGE)
c.brake(linuxcnc.BRAKE_RELEASE)

c.flood(linuxcnc.FLOOD_ON)
c.flood(linuxcnc.FLOOD_OFF)

c.home(2)

c.jog(linuxcnc.JOG_STOP,         jjogmode, joint_num_or_axis_index)
c.jog(linuxcnc.JOG_CONTINUOUS,   jjogmode, joint_num_or_axis_index, velocity)
c.jog(linuxcnc.JOG_INCREMENT,    jjogmode, joint_num_or_axis_index, velocity, increment)

c.load_tool_table()

c.maxvel(200.0)

c.mdi("G0 X10 Y20 Z30")

c.mist(linuxcnc.MIST_ON)
c.mist(linuxcnc.MIST_OFF)

c.mode(linuxcnc.MODE_MDI)
c.mode(linuxcnc.MODE_AUTO)
c.mode(linuxcnc.MODE_MANUAL)

c.override_limits()

c.program_open("foo.ngc")
c.reset_interpreter()

c.tool_offset(toolno, z_offset,  x_offset, diameter, frontangle, backangle, orientation)
```

10.2.6.1. Atributos `linuxcnc.command`

`serial`
el número de serie del comando actual

10.2.6.2. métodos `linuxcnc.command`:

`abort()`
enviar mensaje EMC_TASK_ABORT.

`auto(int [, int])`
ejecutar, escalonar, pausar o reanudar un programa.

`brake(int)`
engrane o suelte el freno del husillo.

`debug(int)`
establecer el nivel de depuración a través del mensaje EMC_SET_DEBUG.

`feedrate(float)`
establecer la velocidad de avance.

`flood(int)`
encender/apagar inundacion.
Sintaxis:
flood(command)
flood(linuxcnc.FLOOD_ON)
flood(linuxcnc.FLOOD_OFF)
Constants:
FLOOD_ON
FLOOD_OFF

`home(int)`
home una articulación determinada.

`jog(command-constant, bool, int[, float[, float]])`
Sintaxis:
jog(command, jjogmode, joint_num_or_axis_index, velocity[, distance]])
jog(linuxcnc.JOG_STOP, jjogmode, joint_num_or_axis_index)
jog(linuxcnc.JOG_CONTINUOUS, jjogmode, joint_num_or_axis_index, velocity)
jog(linuxcnc.JOG_INCREMENT, jjogmode, joint_num_or_axis_index, velocity, distance) Constantes de comando;
linuxcnc.JOG_STOP
linuxcnc.JOG_CONTINUOUS
linuxcnc.JOG_INCREMENT

jjogmode

True
solicitar jog de articulacion individual (requiere teleop_enable (0))

False
solicitud jog de eje de coordenadas cartesianas (requiere teleop_enable (1))

joint_num_or_axis_index

Para jog articular (jjogmode = 1)
joint_number

Para jog de eje de coordenadas cartesianas (jjogmode = 0)
índice de base cero de la coordenada del eje con respecto a las letras de coordenadas conocidas XYZABCUVW (x=>0,y=>1,z=>2,a=>3,b=>4,c=>5,u=>6,v=>7,w=>8)

`load_tool_table()`
Vuelve a cargar la tabla de herramientas.

`maxvel(float)`
establecer la velocidad máxima

mdi(string)
Enviar un comando MDI. Máximo 255 caracteres.

mist(int)
activa/desactiva la niebla.
Sintaxis:
mist(command)
mist(linuxcnc.MIST_ON)
mist(linuxcnc.MIST_OFF)
Constants:
MIST_ON
MIST_OFF

mode(int)
establecer modo (MODE_MDI, MODE_MANUAL, MODE_AUTO).

override_limits()
establecer la bandera de límites de ajuste de eje.

program_open(string)
Abrir un archivo NGC.

rapidrate()
establecer factor de ajuste de rápidos

reset_interpreter()
restablecer el intérprete RS274NGC

set_adaptive_feed(int)
establecer bandera de alimentación adaptativa

set_analog_output(int, float)
ajustar el pin de salida analógica al valor

set_block_delete(int)
establecer bloque eliminar bandera

set_digital_output(int, int)
configurar el pin de salida digital al valor

set_feed_hold(int)
activar/desactivar la retención de alimentación

set_feed_override(int)
activar/desactivar el ajuste de alimentación

set_max_limit(int, float)
establecer el límite de posición máxima para un eje dado

set_min_limit()
establecer el límite de posición mínima para un eje dado

set_optional_stop(int)
activar/desactivar parada opcional

set_spindle_override(int [, int])
establecer ajuste del husillo habilitado. Por defecto el husillo 0.

spindle(int [[float] [int] [float, int]])
establecer la dirección del husillo. Uno de SPINDLE_FORWARD, SPINDLE_REVERSE, SPINDLE_OFF, SPINDLE_INCREASE, SPINDLE_DECREASE o SPINDLE_CONSTANT.

```
#!/usr/bin/env python
import linuxcnc
c = linuxcnc.command()

# Aumente la velocidad del husillo 0 en 100 rpm. El husillo debe estar encendido primero
c.spindle(linuxcnc.INCREASE)

# Aumente la velocidad del husillo 2 en 100 rpm. El husillo debe estar encendido primero
c.spindle(linuxcnc.SPINDLE_INCREASE, 2)

# Establecer la velocidad del husillo de 0 a 1024 rpm
c.spindle.(linuxcnc.SPINDLE_FORWARD, 1024)

# Establecer la velocidad del husillo 1 a -666 rpm
c.spindle.(linuxcnc.SPINDLE_REVERSE, 666, 1)

# Detener husillo 0
c.spindle.(linuxcnc.SPINDLE_OFF)

# Detener el eje 0 explícitamente
c.spindle.(linuxcnc.SPINDLE_OFF, 0)
```

spindleoverride(float [, int])
establecer el factor de ajuste del husillo. Por defecto el husillo 0.

state(int)
establecer el estado de la máquina. El estado de la máquina debe ser STATE_ESTOP, STATE_ESTOP_RESET, STATE_ON o STATE_OFF

task_plan_sync()
al finalizar esta llamada, el archivo var en el disco se actualiza con valores vivos del intérprete.

teleop_enable(int)
habilitar/deshabilitar el modo teleop (deshabilitar para jogging articular).

tool_offset(int, float, float, float, float, float, int)
establecer offset de herramienta. Ver ejemplo de uso arriba.

traj_mode(int)
establecer el modo de trayectoria. El modo es uno de MODE_FREE, MODE_COORD o MODE_TELEOP.

unhome(int)
dejar sin home una articulación determinada.

wait_complete([float])
Esperar a que se complete el último comando enviado. Si el tiempo de espera en segundos no esta especificado, el valor predeterminado es 5 segundos. Devuelve -1 si se agota el tiempo de espera, devuelva RCS_DONE o RCS_ERROR según el estado de ejecución del comando

10.2.7. Lectura del canal de error

Para manejar mensajes de error, conéctese al canal de error y periódicamente ejecute poll().

Tenga en cuenta que el canal NML para mensajes de error tiene una cola (distinta a los canales de comando y estado), lo que significa que el primer consumidor de un mensaje de error elimina ese mensaje de la cola; si es otro consumidor de mensajes de error (por ejemplo, Axis) "verá" el mensaje dependiendo del timing. Se recomienda tener solo una tarea de lector de canal de error en una configuración.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import linuxcnc
e = linuxcnc.error_channel()

error = e.poll()

if error:
    kind, text = error
    if kind in (linuxcnc.NML_ERROR, linuxcnc.OPERATOR_ERROR):
        typus = "error"
    else:
        typus = "info"
    print typus, text
```

10.2.8. Lectura de valores de archivo ini

Aquí hay un ejemplo para leer valores de un archivo ini a través del objeto `linuxcnc.ini`:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# run as:
# python ini-example.py ~/emc2-dev/configs/sim/axis/axis_mm.ini

import sys
import linuxcnc

inifile = linuxcnc.ini(sys.argv[1])

# inifile.find () devuelve None si no se encontró la clave: el
# following idiom es útil para establecer un valor predeterminado:

machine_name = inifile.find("EMC", "MACHINE") or "unknown"
print "machine name: ", machine_name

# inifile.findall () devuelve una lista de coincidencias o una lista vacía
# si no se encontró la clave:

extensions = inifile.findall("FILTER", "PROGRAM_EXTENSION")
print "extensions: ", extensions

# anular el archivo NML predeterminado por parámetro ini si se proporciona
nmlfile = inifile.find("EMC", "NML_FILE")
if nmlfile:
    linuxcnc.nmlfile = os.path.join(os.path.dirname(sys.argv[1]), nmlfile)
```

10.2.9. El tipo `linuxcnc.positionlogger`

Algunas sugerencias de uso se pueden obtener de `src/emc/usr_intf/gremlin/gremlin.py`.

10.2.9.1. miembros

`npts`
: número de puntos.

10.2.9.2. métodos

`start(float)`
iniciar el registrador de posición y ejecutar cada ARG segundos

`clear()`
borrar el registrador de posición

`stop()`
detener el registrador de posición

`call()`
plot el backplot ahora.

`last([int])`
Devuelve el punto más reciente de la trama o ninguno. ,

Capítulo 11

Drivers

11.1. Controlador de puerto paralelo

El componente hal_parport es un controlador para el puerto paralelo PC tradicional. El puerto tiene un total de 17 pines físicos. El puerto paralelo original dividia esos pines en tres grupos: datos, control y estado. El grupo de datos consta de 8 pines de salida, el grupo de control consta de 4 pines, y el El grupo de estado consta de 5 pines de entrada.

A principios de la década de 1990, se introdujo el puerto paralelo bidireccional, que permite que el grupo de datos se use para salida o entrada. El controlador HAL admite el puerto bidireccional, y permite al usuario configurar el grupo de datos como entrada o salida. Si se configura como *out*, un puerto proporciona un total de 12 salidas y 5 entradas. Si se configura como *in*, proporciona 4 salidas y 13 entradas.

En algunos puertos paralelos, los pines del grupo de control son de colector abierto, que pueden también ser puestos en BAJO por una puerta externa. En una tarjeta con pines de control de colector abierto, si se configura como *x*, proporciona 8 salidas y 9 entradas.

En otros puertos paralelos, el grupo de control tiene controladores push-pull y no pueden ser utilizados como entrada.

HAL y colectores abiertos
HAL no puede determinar automáticamente si los pines bidireccionales del modo *x* son efectivamente de colector abierto (OC). Si no lo son, no pueden usarse como entradas, e intentar conducirlos BAJOS desde una fuente externa puede dañar el hardware.
Para determinar si su puerto tiene pines *open collector*, cargue hal_parport en modo *x* Sin un dispositivo conectado, HAL debería leer el pin como VERDADERO. Despues, inserte una resistencia de 470 ohmios desde uno de los pines de control a GND. Si el resultado de la tensión en el pin de control está cerca de 0V, y HAL ahora lee el pin como FALSO, entonces tienes un puerto OC. Si el voltaje resultante está lejos de 0V, o HAL no lee el pin como FALSE, entonces su puerto no puede usarse en el modo *x*.
El hardware externo que controla los pines de control también debe usar puertas de colector abierto (por ejemplo, 74LS05).
En algunas computadoras, la configuración del BIOS puede afectar a si el modo *x* puede ser usado. El modo *SPP* es el más probable que funcione.

No se admiten otras combinaciones, y no se puede cambiar un puerto de entrada a salida una vez que el controlador está instalado.

El controlador parport puede controlar hasta 8 puertos (definido por MAX_PORTS en hal_parport.c). Los puertos están numerados comenzando en cero.

11.1.1. Carga

El controlador hal_parport es un componente en tiempo real, por lo que debe cargarse en el hilo de tiempo real con *loadrt*. La cadena de configuración describe los puertos paralelo que se utilizarán y (opcionalmente) sus tipos. Si la cadena de configuración no describe al menos un puerto, es un error.

```
loadrt hal_parport cfg="port [type] [port [type] ...]"
```

Especificando el Puerto Los números inferiores a 16 se refieren a puertos paralelos detectados por el sistema. Este es la forma más simple de configurar el controlador hal_parport, y coopera con el driver Linux parport_pc, si está cargado. El puerto 0 es el primer puerto paralelo detectado en el sistema, 1 es el siguiente, y así sucesivamente.

Configuracion basica Esto usará el primer puerto paralelo que Linux detecta:

```
loadrt hal_parport cfg="0"
```

Usando la dirección del puerto La dirección del puerto puede especificarse usando la notación hexadecimal *0x* y luego la dirección.

```
loadrt hal_parport cfg="0x378"
```

Tipo Para cada puerto paralelo manejado por el controlador hal_parport, se puede especificar un *tipo* opcionalmente. El tipo es uno de *in*, *out*, *epp* o *x*.

1. Dirección de Puerto Paralelo

Pin	in	out/epp	x
1	out	out	in
2	in	out	out
3	in	out	out
4	in	out	out
5	in	out	out
6	in	out	out
7	in	out	out
8	in	out	out
9	in	out	out
10	in	in	in
11	in	in	in
12	in	in	in
13	in	in	in
14	in	out	in
15	in	in	in
16	out	out	in
17	out	out	in

Si el tipo no está especificado, el valor predeterminado es *out*.

El tipo *epp* es igual a *out*, pero el controlador hal_parport pide que el puerto cambie al modo EPP. El controlador hal_parport no **usa** el protocolo de bus EPP, pero en algunos sistemas el modo EPP cambia las características del puerto de una manera que puede hacer que algún hardware marginal funcione mejor. Se sabe que la bomba de carga del Gecko G540 requiere esto en un puerto paralelo.

Consulte la Nota anterior sobre el modo *x*.

Ejemplo con dos puertos paralelos Esto habilitará dos puertos paralelos detectados por el sistema, el primero en modo de salida y el segundo en modo de entrada:

```
loadrt hal_parport cfg = "0 out 1 in"
```

Funciones También debe indicar a LinuxCNC que ejecute las funciones *read* y *write*.

```
addf parport.0.read base-thread
addf parport.0.write base-thread
```

11.1.2. Dirección del puerto PCI

Entre las buena tarjetas PCI parport estan las que contienen el chipset Netmos 9815. Tiene buenas señales de + 5V, y puede venir en puertos únicos o duales.

Para encontrar las direcciones de E/S para tarjetas PCI, abra una ventana de terminal y use el comando que lista pci:

```
lspci -v
```

Busque la entrada con "Netmos" en ella. Ejemplo de una tarjeta de 2 puertos:

```
0000:01:0a.0 Communication controller: \
      Netmos Technology PCI 9815 Multi-I/O Controller (rev 01)
Subsystem: LSI Logic / Symbios Logic 2POS (2 port parallel adapter)
Flags: medium devsel, IRQ 5
I/O ports at b800 [size=8]
I/O ports at bc00 [size=8]
I/O ports at c000 [size=8]
I/O ports at c400 [size=8]
I/O ports at c800 [size=8]
I/O ports at cc00 [size=16]
```

Por experimentación, se ha encontrado que el primer puerto (el puerto en la tarjeta) usa el la tercera dirección enumerada (c000) y el segundo puerto (la que se conecta con un cable de cinta) utiliza la primera dirección enumerada (b800). El siguiente ejemplo muestra el puerto paralelo de la placa madre y un puerto paralelo PCI que usa el valor predeterminado fuera de la dirección.

```
loadrt hal_parport cfg = "0x378 0xc000"
```

Tenga en cuenta que sus valores pueden ser diferentes. Las tarjetas Netmos son Plug-N-Play, y podrían cambiar su configuración dependiendo de qué ranura en la que se pongan, así que si le gusta "meterse debajo del capó" y reorganizar las cosas, asegúrese de verificar estos valores antes de que usted inicie LinuxCNC.

11.1.3. Pines

Para cada pin, *<p>* es el número de puerto, y *<n>* es el número de pin físico en el conector D-shell de 25 pines.

- *parport.<p>.pin-<n>-out* (bit) Maneja el pin de salida física.
- *parport.<p>.pin-<n>-in* (bit) Realiza el seguimiento de un pin de entrada física.
- *parport.<p>.pin-<n>-in-not* (bit) Realiza el seguimiento de un pin de entrada física, pero invertido.

Para cada pin de salida física, el controlador crea un solo pin HAL, por ejemplo: *parport.0.pin-14-out*.

Para cada pin de entrada física, el controlador crea dos pines HAL, por ejemplo: *parport.0.pin-12-in* y *parport.0.pin-12-in-not*.

El pin HAL *-in* es VERDADERO si el pin físico es alto, y FALSO si el pin físico es bajo. El pin HAL *-in-not* está invertido y es FALSO si el pin físico es alto

11.1.4. Parámetros

- *parport.<p>.pin-<n>-out-invert* (bit) Invierte un pin de salida.
- *parport.<p>.pin-<n>-out-reset* (bit) (solo para *out* pins) TRUE si el pin debe reiniciarse cuando se ejecuta la función *-reset*.
- *parport.<p>.reset-time* (u32) El tiempo (en nanosegundos) entre que un pin se establece mediante *write* y se restablece mediante la función *reset* si está habilitado.

El parámetro *-invert* determina si un pin de salida está activo alto o activo bajo. Si *-invert* es FALSE, establecer el pin HAL *-out* TRUE lleva al pin físico a alto, y FALSO lo conduce bajo. Si *-invert* es TRUE, entonces al establecer el pin HAL *-out* TRUE, el pin físico estará bajo.

11.1.5. Funciones

- *parport.<p>.read* (funct) Lee los pines de entrada física del puerto *<portnum>* y actualiza los pines HAL *-in* y *-in-not*.
- *parport.read-all* (funct) Lee los pines de entrada física de todos los puertos y actualiza los pines HAL *-in* y *-in-not*.
- *parport.<p>.write* (funct) Lee los pines HAL *-out* del puerto *<p>* y actualiza los pines de salida física del puerto.
- *parport.write-all* (funct) Lee los pines HAL *-out* de todos los puertos y actualiza todos los pines de salida física.
- *parport.<p>.reset* (funct) Espera hasta que *reset-time* halla transcurrido desde la *write* asociada, luego restablece los pines a los valores indicados por las configuraciones *-out-invert* y *-out-invert*. *reset* debe ser, en el mismo hilo, posterior a *write*. Si *-reset* es TRUE, entonces la función *reset* configurará el pin al valor de *-out-invert*. Esta puede usarse junto con *doublefreq* de stepgen para producir un paso por periodo El stepgen steppace para ese pin debe establecerse en 0 para habilitar doublefreq.

Las funciones individuales se proporcionan para situaciones en las que un puerto necesita ser actualizado en un hilo muy rápido, pero otros puertos pueden ser actualizado en un subproceso más lento para ahorrar tiempo de CPU. Probablemente no sea una buena idea utilizar una función *-all* y una función individual al mismo tiempo.

11.1.6. Problemas comunes

Si cargando el módulo se informa

```
insmod: error inserting '../rtlib/hal_parport.ko':
-1 Device or resource busy
```

asegúrese de que el módulo kernel estándar *parport_pc* no este cargado [1] y que ningún otro dispositivo en el sistema ha reclamado los puertos de E/S.

Si el módulo se carga pero no parece funcionar, entonces el puerto o la dirección puede ser incorrecta

11.1.7. Usando DoubleStep

Para configurar DoubleStep en el puerto paralelo, debe agregar la función parport.n.reset después de parport.n.write y configurar stepspace en 0 y el tiempo de reset deseado. Entonces, ese paso puede verificarse en cada período en HAL y luego desactivado por parport después de ser verificado durante el tiempo especificado por parport.n.reset-time.

Por ejemplo:

```
loadrt hal_parport cfg="0x378 out"
setp parport.0.reset-time 5000
loadrt stepgen step_type=0,0,0
addf parport.0.read base-thread
addf stepgen.make-pulses base-thread
addf parport.0.write base-thread
addf parport.0.reset base-thread
addf stepgen.capture-position servo-thread
...
setp stepgen.0.steplen 1
setp stepgen.0.stepspace 0
```

Puede encontrar más información sobre DoubleStep en wiki.

[1] En los paquetes de LinuxCNC para Ubuntu, el archivo /etc/modprobe.d/emc2 generalmente evita que *parport_pc* se cargue automáticamente.

Capítulo 12

Ejemplos de Drivers

12.1. Puerto paralelo PCI

Cuando agrega un segundo puerto paralelo a su bus PCI, debe encontrar la dirección antes de poder usarlo con LinuxCNC.

Para encontrar la dirección de su tarjeta de puerto paralelo, abra una ventana de terminal y escriba

```
lspci -v
```

Verá algo similar a esto, así como información sobre todo lo demás en el bus PCI:

```
0000:00:10.0 Communication controller: \
        NetMos Technology PCI 1 port parallel adapter (rev 01)
        Subsystem: LSI Logic / Symbios Logic: Unknown device 0010
        Flags: medium devsel, IRQ 11
        I/O ports at a800 [size=8]
        I/O ports at ac00 [size=8]
        I/O ports at b000 [size=8]
        I/O ports at b400 [size=8]
        I/O ports at b800 [size=8]
        I/O ports at bc00 [size=16]
```

En mi caso, la dirección fue la primera, así que cambié mi archivo .hal de

```
loadrt hal_parport cfg=0x378
```

a

```
loadrt hal_parport cfg="0x378 0xa800 in"
```

(Tenga en cuenta las comillas dobles que rodean las direcciones).

y luego agregó las siguientes líneas para que el parport se lea y se escriba:

```
addf parport.1.read base-thread
addf parport.1.write base-thread
```

Después de hacer lo anterior, ejecute su configuración y verifique que el puerto paralelo se cargó en la ventana Máquina/Mostrar Configuración HAL.

LinuxCNC puede controlar hasta 8 husillos. El número se establece en el archivo INI. Todos los ejemplos a continuación se refieren a una configuración de husillo único con pines de control del husillo con nombres como spindle.0... En el caso de una máquina de husillos múltiples, todo lo que cambia es que existen pines adicionales con nombres como spindle.6...

12.2. Control de husillo

12.2.1. Velocidad del husillo 0-10v

Si la velocidad de su husillo está controlada por una señal analógica, (por ejemplo, por un VFD con una señal de 0 a 10 voltios) y estás usando una tarjeta DAC como la m5i20 para emitir la señal de control:

Primero debe calcular la escala de la velocidad del husillo para controlar la señal. Para este ejemplo, la velocidad máxima del eje de 5000 RPM es igual a 10 voltios

$$\frac{10\,Volts}{5000\,RPM} = \frac{0.002\,Volts}{1\,RPM}$$

Tenemos que agregar un componente de escala al archivo HAL para escalar spindle.N.speed-out de 0 a 10 que necesita el VFD si su DAC de la la tarjeta no hace escalado.

```
loadrt scale count=1
addf scale.0 servo-thread
setp scale.0.gain 0.002
net spindle-speed-scale spindle.0.speed-out => scale.0.in
net spindle-speed-DAC scale.0.out => <su nombre pin DAC>
```

12.2.2. Velocidad del husillo PWM

Si su husillo puede ser controlado por una señal PWM, use el componente pwmgen para crear la señal:

```
loadrt pwmgen output_type=0
addf pwmgen.update servo-thread
addf pwmgen.make-pulses base-thread
net spindle-speed-cmd spindle.0.speed-out => pwmgen.0.value
net spindle-on spindle.0.on => pwmgen.0.enable
net spindle-pwm pwmgen.0.pwm => parport.0.pin-09-out
# Establecer la velocidad máxima del husillo en RPM
setp pwmgen.0.scale 1800
```

Esto supone que la respuesta del controlador de husillo a PWM es simple: 0% PWM da 0 RPM, 10% PWM da 180 RPM, etc. Si hay un PWM mínimo requerido para hacer girar el husillo, siga el ejemplo en la configuración de muestra de torno nist para usar un componente de escala.

12.2.3. Habilitación del husillo

Si necesita una señal de habilitación de husillo, enlace su pin de salida a spindle.0.on. Para vincular estos pines a un pin de puerto paralelo, coloque algo como lo siguiente en su archivo .hal, asegurándose de elegir el pin que está conectado a su dispositivo de control.

```
net spindle-enable spindle.N.on => parport.0.pin-14-out
```

12.2.4. Dirección del husillo

Si tiene control de dirección de su husillo, los pines HAL spindle.N.forward y spindle.N.reverse están controlados por M3 y M4. La velocidad del husillo *Sn* debe establecerse en un valor positivo distinto de cero para que M3/M4 activen el movimiento del husillo.

Para vincular estos pines a un pin de puerto paralelo, coloque algo como lo siguiendo en su archivo .hal asegurándose de elegir el pin que está conectado a su dispositivo de control.

```
net spindle-fwd spindle.0.forward => parport.0.pin-16-out
net spindle-rev spindle.0.reverse => parport.0.pin-17-out
```

12.2.5. Arranque suave del husillo

Si necesita rampa en el comando de velocidad del eje y su control no tiene esa característica, se puede hacer en HAL. Básicamente necesita secuestrar la salida de spindle.N.speed-out y llevarla a través de un componente limit2 con la escala establecida, por lo que las rpm subiran en rampa desde spindle.N.speed-out al dispositivo que recibe las rpm. La segunda parte es informar a LinuxCNC cuando el husillo está a velocidad para que el movimiento pueda empezar.

En el ejemplo de 0-10 voltios, la línea *net spindle-speed-scale spindle.0.speed-out => scale.0.in* se cambia como se muestra en el siguiente ejemplo:

Introducción a los componentes HAL2 limit2 y near:

En caso de que no los haya visto antes, aquí damos una introducción rápida a los dos componentes HAL utilizados en el siguiente ejemplo.

- "limit2" es un componente HAL (coma flotante) que acepta un valor de entrada y proporciona una salida que se ha limitado a un rango máximo/mínimo, y también limitado a no exceder una tasa de cambio especificada.
- "near" es un componente HAL (coma flotante) con una salida binaria que dice si dos entradas son aproximadamente iguales.

Más información está disponible en la documentación de los componentes HAL, o desde las páginas del manual, simplemente escriba *man limit2* o *man near* en una terminal.

```
# carga en tiempo real un limit2 y un near con nombres para que sean más fáciles de seguir
loadrt limit2 names=spindle-ramp
loadrt near names=spindle-at-speed

# agregar las funciones a un hilo
addf spindle-ramp servo-thread
addf spindle-at-speed servo-thread

# establece el parámetro para la tasa máxima de cambio
# (aceleración / desaceleración máxima del husillo en unidades por segundo)
setp spindle-ramp.maxv 60

# secuestrar la velocidad del husillo y enviarla a spindle-ramp
net spindle-cmd <= spindle.0.speed-out => spindle-ramp.in

# la salida de spindle-ramp se envía a la escala
net spindle-ramped <= spindle-ramp.out => scale.0.in

# para saber cuándo comenzar el movimiento enviamos al componente near
# (denominado spindle-at-speed) la velocidad ordenada del husillo desde
# la señal spindle-cmd y la velocidad real del mismo
# siempre que su husillo pueda acelerar a la configuración maxv.
net spindle-cmd => spindle-at-speed.in1
net spindle-ramped => spindle-at-speed.in2

# la salida de spindle-at-speed se envía a spindle.0.at-speed
# y cuando esto sea cierto, comenzará el movimiento
net spindle-ready <= spindle-at-speed.out => spindle.0.at-speed
```

12.2.6. Feedback del husillo

12.2.6.1. Movimiento sincronizado del husillo

Spindle feedback is needed by LinuxCNC to perform any spindle coordinated motions like threading and constant surface speed. LinuxCNC can perform synchronised motion and CSS with any of up to 8 spindles. Which spindles are used is controlled from

G-code. CSS is possible with several spindles simultaneously.

The StepConf Wizard can perform the connections for a single-spindle configuration for you if you select Encoder Phase A and Encoder Index as inputs. LinuxCNC necesita retroalimentación del husillo para realizar cualquier movimiento coordinado con el husillo como roscado o velocidad de superficie constante (CSS). LinuxCNC puede realizar movimientos sincronizados y CSS con cualquiera de hasta 8 husillos. Los husillos que se utilizan se controlan desde el código G. CSS es posible con varios husillos simultáneamente.

El asistente StepConf puede realizar las conexiones para configurar un solo husillo si selecciona Encoder Fase A e Encoder Indice como entradas.

Suposiciones de hardware:

- Un codificador está conectado al husillo y emite 100 pulsos por revolución en la fase A
- La fase A del codificador está conectada al pin 10 del puerto paralelo
- El pulso de índice del codificador está conectado al pin 11 del puerto paralelo

Pasos básicos para agregar los componentes y configurarlos: [1] [2] [3]

```
# agregar el codificador a HAL y conéctarlo a los hilos.
loadrt encoder num_chan=1
addf encoder.update-counters base-thread
addf encoder.capture-position servo-thread

# establecer el codificador HAL en 100 pulsos por revolución.
setp encoder.3.position-scale 100

# establecer el codificador HAL en un recuento simple no en cuadratura usando solo A.
setp encoder.3.counter-mode true

# conectar las salidas del codificador HAL a LinuxCNC.
net spindle-position encoder.3.position => spindle.0.revs
net spindle-velocity encoder.3.velocity => spindle.0.speed-in
net spindle-index-enable encoder.3.index-enable <=> spindle.N.index-enable

# conectar las entradas del codificador HAL al codificador real.
net spindle-phase-a encoder.3.phase-A <= parport.0.pin-10-in
net spindle-phase-b encoder.3.phase-B
net spindle-index encoder.3.phase-Z <= parport.0.pin-11-in
```

12.2.6.2. Husillo a velocidad

Para permitir que LinuxCNC espere a que el eje esté a velocidad antes de ejecutar una serie de movimientos, debe establecer spindle.N.at-speed en true cuando el husillo está a la velocidad ordenada. Para hacer esto necesita retroalimentación de un codificador en el husillo. Puesto que la retroalimentación y la velocidad ordenada por lo general, no son "exactamente" iguales, debería usar un componente "near" para determinar que los dos números están lo suficientemente proximos.

Las conexiones necesarias son desde la señal de comando de velocidad del husillo a near.n.in1 y desde el codificador hasta near.n.in2. near.n.out está conectado a spindle.N.at-speed. near.n.scale debe establecerse para decir cuanto de cerca deben estar los dos números antes de activar la salida. Dependiendo de su configuración, es posible que deba ajustar la escala para que funcione con su hardware.

Lo siguiente son las adiciones típicas necesarias para su archivo HAL para habilitar Spindle At Speed. Si ya tiene near en su HAL, aumente count y ajuste el código para adaptarlo. Asegúrese de que los nombres de las señales sean los mismos en su archivo HAL.

[1] En este ejemplo, asumiremos que algunos codificadores ya tienen asignaciones a ejes/articulaciones 0, 1 y 2. Por tanto, el próximo codificador disponible para nosotros sería el número 3. Su situación puede ser diferente.

[2] La habilitación del índice del codificador HAL es una excepción a la regla ya que se comporta como una entrada y una salida, vea la sección del codificador para más detalles

[3] Ya que seleccionamos *conteo simple no en cuadratura* . . . , podemos salir con el conteo de *cuadratura* sin tener ninguna entrada B de cuadratura.

```
# cargar un componente near y adjuntarlo a un hilo
loadrt near
addf near.0 servo-thread

# conectar una entrada a la velocidad de husillo ordenada
net spindle-cmd => near.0.in1

# conectar una entrada a la velocidad del husillo medida por el codificador
net spindle-velocity => near.0.in2

# conectar la salida a la entrada de spindle-at-speed
net spindle-at-speed spindle.0.at-speed <= near.0.out

# configurar las entradas de velocidad del husillo para que estén de acuerdo si están ↩
    dentro del 1%
setp near.0.scale 1.01
```

12.3. Husillo con GS2

Este ejemplo muestra las conexiones necesarias para usar un VFD GS2 de Automation Direct para manejar un husillo. La velocidad y dirección del husillo son controladas por LinuxCNC.

El uso del componente GS2 tiene muy poco que configurar. Empezamos con una configuración generada por el asistente Stepconf. Asegúrese de que los pines con "Husillo CW" y "Husillo PWM" están configurados como no utilizados en la configuración del puerto paralelo.

En el archivo custom.hal colocamos lo siguiente para conectar LinuxCNC al GS2 y que controle la unidad.

GS2 Ejemplo

```
# cargar el componente de espacio de usuario para VFD Automation Direct GS2
loadusr -Wn spindle-vfd gs2_vfd -r 9600 -p none -s 2 -n spindle-vfd

# conecte el pin de dirección del husillo al GS2
net gs2-fwd spindle-vfd.spindle-fwd <= spindle.N.forward

# conecte el pin de arranque del husillo al GS2
net gs2-run spindle-vfd.spindle-on <= spindle.N.on

# conecte el GS2 a velocidad al movimiento a velocidad
net gs2-at-speed spindle.N.at-speed <= spindle-vfd.at-speed

# conecte RPM de husillo al GS2
net gs2-RPM spindle-vfd.speed-command <= spindle.N.speed-out
```

nota
La velocidad de transmisión puede ser más rápida dependiendo del entorno exacto. Tanto la unidad como las opciones de línea de comando deben coincidir. Para verificar errores de transmisión, agregue la opción *-v* a la línea de comando y ejecute desde un terminal.

Necesita configurar un par de cosas en el propio driver GS2 antes de que las comunicaciones modbus funcionen. Puede que sea necesario configurar otros parámetros basado en sus requisitos físicos, pero estos están más allá del alcance de este manual. Consulte el manual de GS2 que vino con la unidad para obtener más información sobre los parámetros del variador.

- Los interruptores de comunicación deben configurarse en RS-232C

- Los parámetros del motor deben configurarse para que coincidan con el motor
- P3.00 (Fuente del comando de operación) debe establecerse en Operación determinada por la interfaz RS-485, 03 o 04
- P4.00 (Fuente del comando de frecuencia) debe establecerse en Frecuencia determinada por la interfaz de comunicación RS232C/RS485, 05
- P9.01 (Velocidad de transmisión) debe establecerse en 9600 baudios, 01
- P9.02 (Protocolo de comunicación) debe establecerse en "Modo Modbus RTU, 8 bits de datos, sin paridad, 2 bits de parada ", 03

Un panel PyVCP basado en este ejemplo está aqui.

Capítulo 13

PLC

Capítulo 14

HAL

14.1. Introducción a HAL

HAL significa *capa de abstracción de hardware*. Al más alto nivel, es simplemente una manera de permitir que una serie de *bloques de construcción* sean cargados e interconectados para ensamblar un sistema complejo. El termino *hardware* es porque HAL fue diseñado originalmente para que fuese fácil configurar LinuxCNC para una amplia variedad de dispositivos de hardware. Mucho de los bloques de construcción son controladores de dispositivos hardware. Sin embargo, HAL puede hacer más que configurar controladores hardware.

14.1.1. HAL se basa en técnicas de diseño de sistemas tradicionales

HAL se basa en los mismos principios que se utilizan para diseñar circuitos y sistemas hardware, por lo que es útil examinar primero esos principios.

Cualquier sistema (incluida una máquina CNC, consiste en componentes interconectados. Para una máquina CNC, esos componentes podrían ser el controlador principal, servoamplificadores o drivers de pasos, motores, codificadores, interruptores de límite, volantes con pulsadores (los populares MPG), quizás un VFD para el husillo, un PLC para manejar un cambiador de herramientas, etc. El creador de la máquina debe seleccionar, montar y conectar estas piezas para formar un sistema completo.

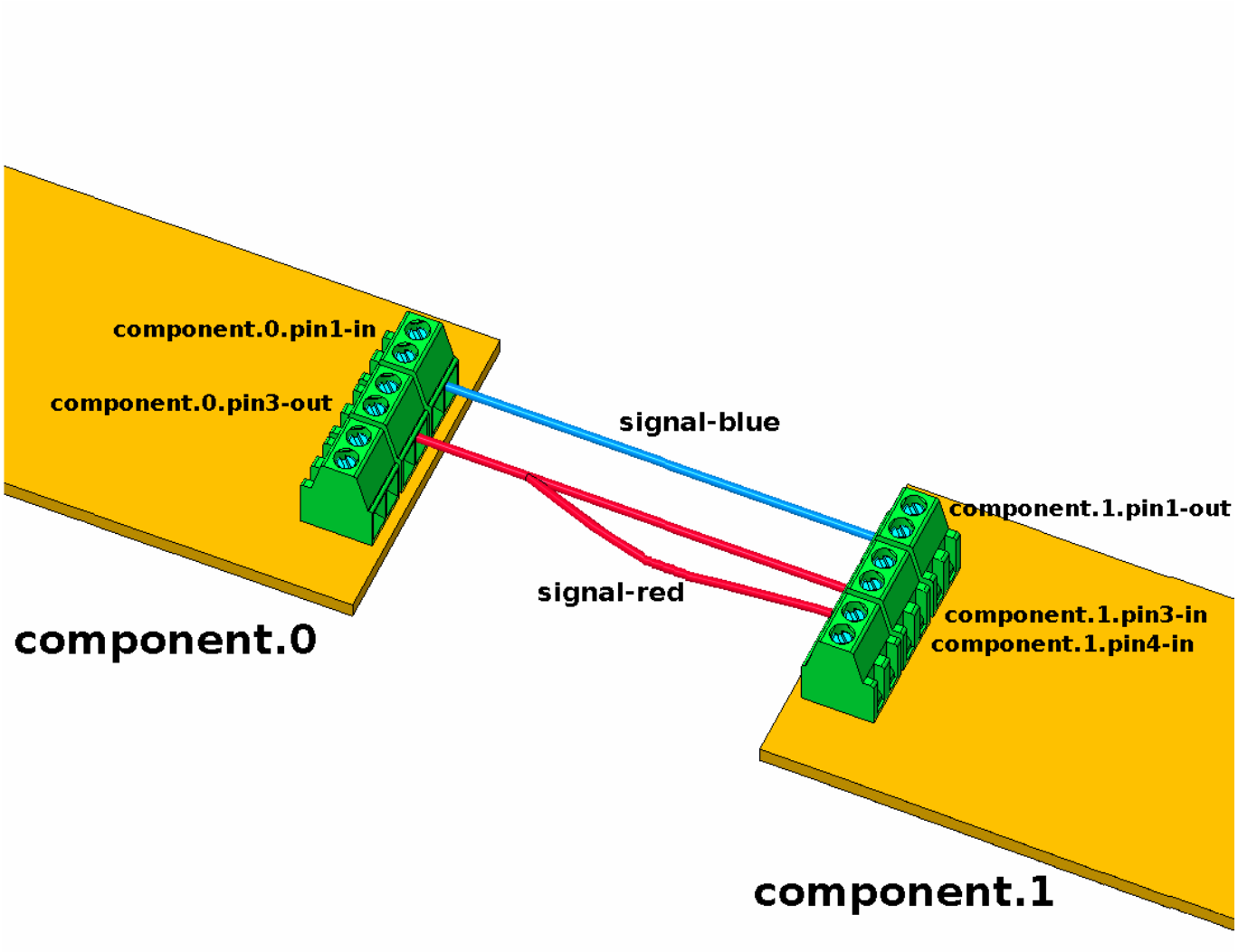


Figura 14.1: Concepto HAL - Conexiones como en un circuito electrico.

La figura se transcribe a codigo HAL de esta manera:

```
net signal-blue    component.0.pin1-in      component.1.pin1-out
net signal-red     component.0.pin3-out     component.1.pin3-in     component.1.pin4-in
```

14.1.1.1. Selección de componentes

El constructor de una máquina no necesita preocuparse de cómo funciona cada componente individualmente. Se tratan como cajas negras. Durante la etapa de diseño, se decide qué componentes se usarán: steppers o servos, que marca de servo-amplificador, qué tipo de interruptores de límite y cuántos, etc. Las decisiones del integrador sobre qué componentes específicos usar se basaran en lo que hace ese componente y las especificaciones suministradas por el fabricante del mismo. El tamaño de un motor y la carga que debe mover afectará a la elección del amplificador necesario para manejarlo. La elección del amplificador puede afectar al tipo de retroalimentación necesario y a las señales de velocidad o posición que deben enviarse al propio amplificador desde el controlador.

De igual forma, en el mundo HAL el integrador debe decidir qué componentes HAL son necesarios. Por lo general, cada tarjeta de interfaz requerirá un controlador. Pueden ser necesarios componentes adicionales para la generación de pulsos de pasos por software, funcionalidad PLC, y una amplia variedad de otras tareas.

14.1.1.2. Diseño de interconexiones

El diseñador de un sistema hardware no solo selecciona los componentes; también decide cómo interconectarlos. Cada caja negra tiene terminales, quizás solo dos como en un simple interruptor, o docenas para una unidad servo o un PLC. Los componentes necesitan estar interconectados. Los motores se conectan a los servo-amplificadores, los interruptores de límite se conectan al controlador, y así sucesivamente. Conforme el constructor de la máquina trabaja en el diseño, crea un diagrama de cableado completo que muestra cómo deben estar interconectados todos los componentes.

En HAL, los cables son las "señales". Cuando se usa HAL, los componentes software están interconectados por señales. El diseñador debe decidir qué señales son necesarias y como deben conectarse.

14.1.1.3. Implementación

Una vez que se completa el diagrama de cableado, es hora de construir la máquina. Las piezas deben ser adquiridas y montadas, y luego son interconectadas de acuerdo con el diagrama de cableado. En un sistema físico, cada interconexión es una trozo de cable, con unas caracteristicas determinadas, que deben conectarse a los terminales apropiados.

HAL proporciona una serie de herramientas para ayudar a *construir* un sistema HAL. Algunas de las herramientas le permiten *conectar* (o desconectar) un solo *cable*. Otras herramientas le permiten guardar una lista completa de todos los componentes, cables y otra información sobre el sistema, para que pueda ser *reconstruido* con un solo comando.

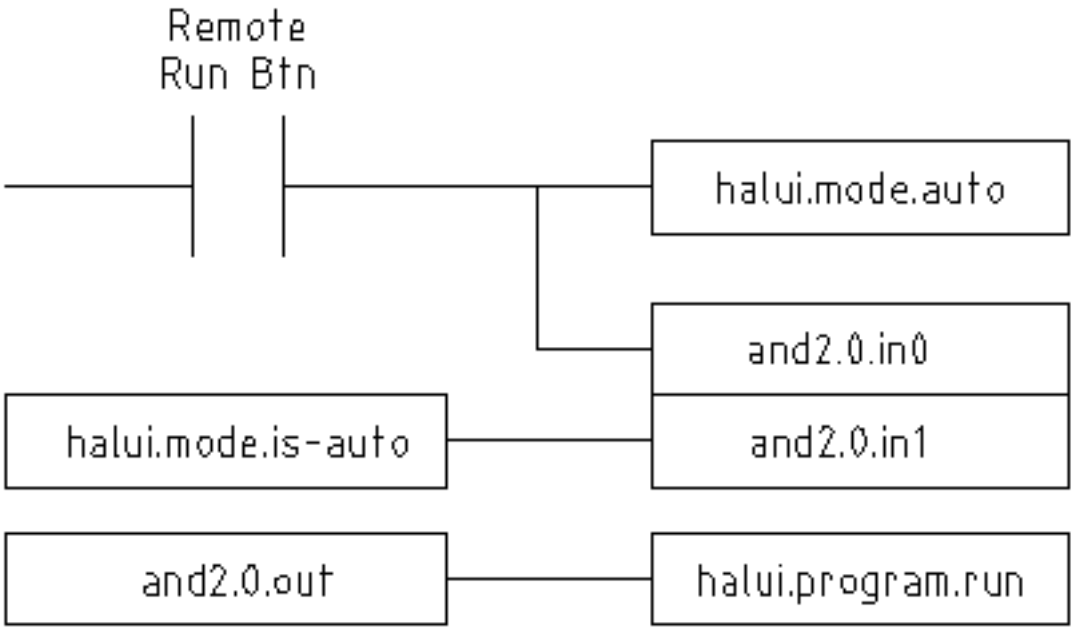
14.1.1.4. Pruebas

Muy pocas máquinas funcionan bien la primera vez. Mientras hace las pruebas pertinentes, el constructor puede usar un tester para ver si un interruptor de límite está funcionando o para medir la tensión de CC que va a un servomotor. Puede conectar un osciloscopio para verificar la sintonización de un driver, o para buscar ruido electrico. Puede encontrar un problema que requiera cambios en el diagrama de cableado; tal vez un componente debe estar conectado de manera diferente o ser reemplazado con otro completamente diferente.

HAL proporciona los equivalentes software de un voltímetro, un osciloscopio, un generador de señales y otras herramientas necesarias para probar y ajustar un sistema. Los mismos comandos utilizados para construir el sistema se pueden usar para hacer cambios según sea necesario.

14.1.1.5. Resumen

Este documento está dirigido a personas que ya saben cómo hacer este tipo de integración de sistemas hardware, pero que no saben cómo conectar el hardware a LinuxCNC. Vea la sección Ejemplo de Arranque remoto en la documentación de Ejemplos UI HAL.



El diseño de hardware tradicional, como se ha descrito arriba, termina en el limite del control principal. Fuera del control (el pc) hay un grupo relativamente simple de cajas, conectadas entre sí para hacer lo que se ha previsto que hagan. En el interior (el pc), el control es un gran misterio; una enorme caja negra que esperamos funcione.

HAL extiende este método de diseño de hardware tradicional al interior de la gran caja negra. Hace de los controladores de dispositivos, e incluso algunas partes internas del controlador, cajas negras más pequeñas que se pueden interconectar, e incluso reemplazar al igual que el hardware externo. HAL permite un *diagrama de cableado del sistema* que mostrará parte del controlador interno, en lugar de una gran caja negra. Y lo más importante; permite que el integrador pruebe y modifique el controlador utilizando los mismos métodos que usaría en el resto del hardware.

Para la mayoría de los integradores son familiares términos como motores, amplificadores y codificadores. Cuando hablamos de usar ocho conductores extraflexibles de cable blindado para conectar un codificador a la placa de entrada del servo en la computadora, el lector comprende de inmediato lo que es y le lleva a preguntas del tipo *¿qué tipo de conectores necesitaré en cada extremo?*. El mismo tipo de pensamiento es esencial para HAL, pero ese hilo de pensamiento específico puede tardar un poco en ponerse en marcha. Usando HAL, las palabras pueden parecer un poco extrañas al principio, pero el concepto de trabajar desde una conexión a la siguiente es el mismo.

Esta idea de extender el diagrama de cableado al interior del controlador es la adoptada por HAL. Si se siente cómodo con la idea de interconectar cajas negras de hardware, es probable que tenga pocos problemas al utilizar HAL para interconectar cajas negras software.

14.1.2. Conceptos HAL

Esta sección es un glosario que define los términos clave HAL, aunque es un poco diferente de un glosario tradicional porque estos términos no estan ordenados alfabéticamente sino por su relación o flujo dentro de la manera de hacer las cosas con HAL.

Componente

Cuando hablamos de diseño de hardware, nos referimos a las piezas individuales como *partes*, *bloques de construcción*, *cajas negras*, etc. El equivalente HAL es un *componente* o *componente HAL*. (Esta documento utiliza *componente HAL* cuando es posible que haya confusión con otros tipos de componentes, pero normalmente solo usa la palabra *componente*.) Un componente HAL es una pieza de software con entradas, salidas y comportamiento definido, que se puede instalar e interconectar según sea necesario.

Parámetro

Muchos componentes de hardware tienen mandos de ajustes que no están conectados a ningún otro componente, pero que deben tener alguna forma de acceso. Por ejemplo, los servoamplificadores a menudo tienen potenciometros, para permitir el afinado de su ajuste, y puntos de prueba donde puede ser conectado un medidor u osciloscopio para ver los resultados

del ajuste. Los componentes HAL también pueden tener tales cosas, que se conocen como *parámetros*. Hay dos tipos de parámetros: los parámetros de entrada son equivalentes a potenciometros; son valores que pueden ser ajustados por el usuario, y permanecen fijos si no son ajustados de nuevo. Los parámetros de salida no pueden ser ajustados por el usuario; son equivalente a los puntos de prueba que permiten monitorizar las señales internas.

Pin

Los componentes hardware tienen terminales que se utilizan para interconectarlos. El equivalente HAL es un "pin" o "pin HAL" (se usara *pin HAL* cuando sea necesario para evitar confusiones). Todos los pines HAL tienen nombre y esos nombres de pines se usan para interconectarlos. Los pines HAL son entidades de software que existen solo dentro de la computadora.

Pin fisico

Muchos dispositivos de E/S tienen pines físicos reales o terminales que se conectan a hardware externo. Por ejemplo, los pines del conector de puerto paralelo. Para evitar confusiones, estos se conocen como *Pines físicos*. Son esas cosas que se "enchufan" en el mundo real.

Señal

En una máquina física, los terminales reales de los componentes de hardware están interconectados por cables. El equivalente HAL de un cable es una *señal* o *señal HAL*. Las señales HAL conectan los pines HAL según lo requiera el constructor de la máquina. Las señales HAL pueden ser desconectadas y reconectadas a voluntad (incluso cuando la máquina está funcionando).

Tipo

Cuando se usa hardware real, no se conectaría una salida de relé de 24 volt a una entrada analógica +/-10V de un servoamplificador. Los pines HAL tienen las mismas restricciones, que se basan en su tipo. Tanto los pines como las señales tienen tipos, y las señales solo se pueden conectar a pines del mismo tipo. Actualmente hay 4 tipos, que son los siguientes:

- bit - un único valor, VERDADERO/FALSO o ENCENDIDO/APAGADO o 1/0
- float - un valor de punto flotante de 64 bits, con aproximadamente 53 bits de resolución y más de 1000 bits de rango dinámico.
- u32 - un entero sin signo de 32 bits, los valores legales son de 0 a 4.294.967.295
- s32 - un entero de 32 bits con signo, los valores legales son de -2.147.483.647 a +2.147.483.647

Función

Los componentes de hardware reales tienden a actuar de inmediato ante una entrada. Por ejemplo, si el voltaje de entrada a un servoamplificador cambia, la salida también cambia automáticamente. Sin embargo, los componentes de software no pueden actuar "automáticamente". Cada componente tiene un código específico que se debe ejecutar para hacer lo que ese componente se supone debe hacer. En algunos casos, ese código simplemente se ejecuta como parte del componente. Por otra parte, en la mayoría de los casos, especialmente en componentes en tiempo real, el código debe ejecutarse en una secuencia específica y en intervalos específicos. Por ejemplo, las entradas deben leerse antes de realizar los cálculos con los datos de entrada, y las salidas no deben escribirse hasta que los cálculos esten hechos. En estos casos, el código está disponible para el sistema en la forma de una o más *funciones*. Cada función es un bloque de código que realiza una acción específica. El integrador de sistema puede usar *hilos* para programar una serie de funciones que serán ejecutadas en un orden particular y en intervalos de tiempo específicos.

Hilo

Un *hilo* es una lista de funciones que se ejecuta en intervalos específicos como parte de una tarea en tiempo real. Cuando un hilo es creado por primera vez, tiene un intervalo de tiempo específico (período), pero no tiene funciones. Las funciones se pueden agregar al hilo, y se ejecutarán en orden cada vez que se ejecuta el hilo.

Como ejemplo, supongamos que tenemos un componente de puerto paralelo llamado hal_parport. Ese componente define uno o más pines HAL para cada pin físico. Los pines se describen en la sección de documentación de ese componente: sus nombres, cómo se relaciona cada pin HAL con pines físicos, si están invertidos, etc. Eso por si solo no hace que los datos de los pines HAL lleguen a los pines físicos. Se necesita código para hacer eso, y ahí es donde entran en escena las funciones. El componente hal_parport necesita al menos dos funciones: una para leer los pines de entrada física y actualizar los pines HAL, y otra para tomar datos de los pines HAL y escribirlo en pines de salida física. Ambas funciones son parte del driver parport.

14.1.3. Componentes HAL

Cada componente HAL es una pieza de software con entradas, salidas y comportamiento bien definidos, que se pueden instalar e interconectar como sea necesario. Esta sección enumera algunos de los componentes disponibles y una breve descripción de lo que hace cada uno. Los detalles completos para cada componente estan disponible más adelante en este documento.

14.1.3.1. Programas externos con "enganches" HAL

nota
Un enganche, o "hook", es la parte de codigo de un programa destinada a interceptar llamadas de función, mensajes o eventos pasados entre componentes de software.

motion
Un módulo en tiempo real que acepta mensajes NML [1] de comandos de movimiento e interactúa con HAL

iocontrol
Un módulo de espacio de usuario que acepta comandos de E/S NML e interactúa con HAL

classicladder
Un PLC que usa HAL para todas sus E/S

halui
Un programa de espacio de usuario que interactúa con HAL y envía comandos NML, que está destinado a funcionar como una interfaz de usuario completa utilizando botones e interruptores externos.

14.1.3.2. Componentes internos

stepgen
Generador software de impulsos de pasos, con lazo de posición. Ver la sección stepgen

encoder
Contador codificador basado en software. Ver la sección encoder

pid
Bucle de control Proporcional/Integral/Derivativo. Ver la sección pid

siggen
Un generador de ondas seno/coseno/triángular/cuadrada para pruebas. Ver la sección siggen

supply
una fuente simple para pruebas

14.1.3.3. Controladores de hardware

hal_ax5214h
Un controlador para la tarjeta de E/S digital Axiom Measurement & Control AX5241H

hal_gm
Placa General Mechatronics GM6-PCI

hal_m5i20
Placa Mesa Electronics 5i20

[1] El Language Neutral de Mensajes NML proporciona un mecanismo para el manejo de múltiples tipos de mensajes en el mismo búfer, así como simplifica la interfaz para codificar y decodificar memorias intermedias en formato neutral y su mecanismo de configuración.

hal_motenc
Placa Vital Systems MOTENC-100

hal_parport
Puerto paralelo de PC.

hal_ppmc
Familia de controladores de la familia Pico (PPMC, USC y UPC)

hal_stg
Tarjeta Servo To Go (versión 1 y 2)

hal_vti
Controlador Vigilant Technologies PCI ENCDAC-4

14.1.3.4. Herramientas y utilidades

Halcmd
Herramienta de línea de comandos para configuración y ajuste. Ver sección halcmd

halgui
Herramienta GUI para configuración y ajuste (aún no implementada).

halmeter
Un práctico multímetro para señales HAL. Ver la sección halmeter.

halscope
Un osciloscopio de almacenamiento digital completo para señales HAL. Ver la sección halscope.

Cada uno de estos bloques de construcción se describe en detalle en capítulos posteriores.

14.1.4. Problemas de sincronización en HAL

A diferencia de los modelos de cableado físico entre cajas negras en el que se dijo que está basado HAL, la simple conexion de dos pines con una señal HAL queda aun lejos de la acción del caso físico.

La lógica de relés consiste en relés conectados entre sí, y cuando el contacto se abre o se cierra, la corriente fluye (o se detiene) inmediatamente. Otras bobinas pueden cambiar de estado, etc., y todo simplemente *ocurre*. Pero un PLC de lógica de escalera no funciona de esa manera. Usualmente, en un solo pase a través de la escalera, cada escalón se evalúa en el orden en que aparece, y solo una vez por pase. Un ejemplo perfecto es una escalera de un solo escalón, con un contacto NC en serie con una bobina. El contacto y la bobina pertenecen al mismo rele

Si se tratara de un relé convencional, tan pronto como se active la bobina, los contactos comienzan a abrirse y a desenergizarlo. Eso significa que los contactos cerraran de nuevo, etc, etc. El rele se convierte en un zumbador.

Con un PLC, si la bobina está DESACTIVADA y el contacto está cerrado cuando el PLC comienza a evaluar el peldaño, cuando termina ese pase, la bobina estára encendida. El hecho de que al encender la bobina se abre el contacto alimentado es ignorado hasta el próximo pase. En la siguiente pasada, el PLC ve que el el contacto está abierto y desenergiza la bobina. Entonces el rele sigue cambiando rápidamente entre encendido y apagado, pero a un ritmo determinado por la frecuencia con que el PLC evalúa el escalon.

En HAL, la función es el código que evalúa el(los) escalón(es). De hecho, la versión en tiempo real HAL de ClassicLadder exporta una función que hace exactamente eso. Por otra parte, un hilo es lo que permite la ejecución de la función en intervalos de tiempo específicos. Al igual que puede elegir tener un PLC evalúando todos sus escalones cada 10 ms, o cada segundo, puede definir hilos HAL con diferentes períodos.

Lo que distingue a un hilo de otro *no* es lo que el hilo hace - eso está determinado por las funciones que son conectadas a el. La distinción real es simplemente la frecuencia con la que un hilo corre.

En LinuxCNC, puede tener un hilo de 50 us y un hilo de 1 ms. Estos se crearían en base a BASE_PERIOD y SERVO_PERIOD; los tiempos exactos dependeran de los valores en su archivo ini.

El siguiente paso es decidir qué debe hacer cada hilo. Algunas de estas decisiones son las mismas en (casi) cualquier sistema LinuxCNC - Por ejemplo, la funcion motion-command-handler siempre se agrega al hilo servo.

Otras conexiones seran hechas por el integrador. Estas pueden incluir conectar las funciones de lectura de codificador de un controlador STG y las de escritura de un DAC al hilo servo, o enganchar la función stepgen al hilo base, junto con con la(s) función(es) parport para escribir pasos en el puerto paralelo.

14.2. Referencia básica HAL

Este documento proporciona una referencia de los conceptos básicos de HAL.

14.2.1. Comandos HAL

Se puede encontrar información más detallada en la página del manual de *halcmd*; ejecutar *man halcmd* en una ventana de terminal.

Para ver la configuración de HAL y verificar el estado de los pines y parámetros, use la ventana de configuración de HAL en el menú *Máquina* en AXIS. Para ver el estado de un pin, abra la pestaña *Ver* y haga clic en cada pin de su interés, que se agregará a la ventana de observación.

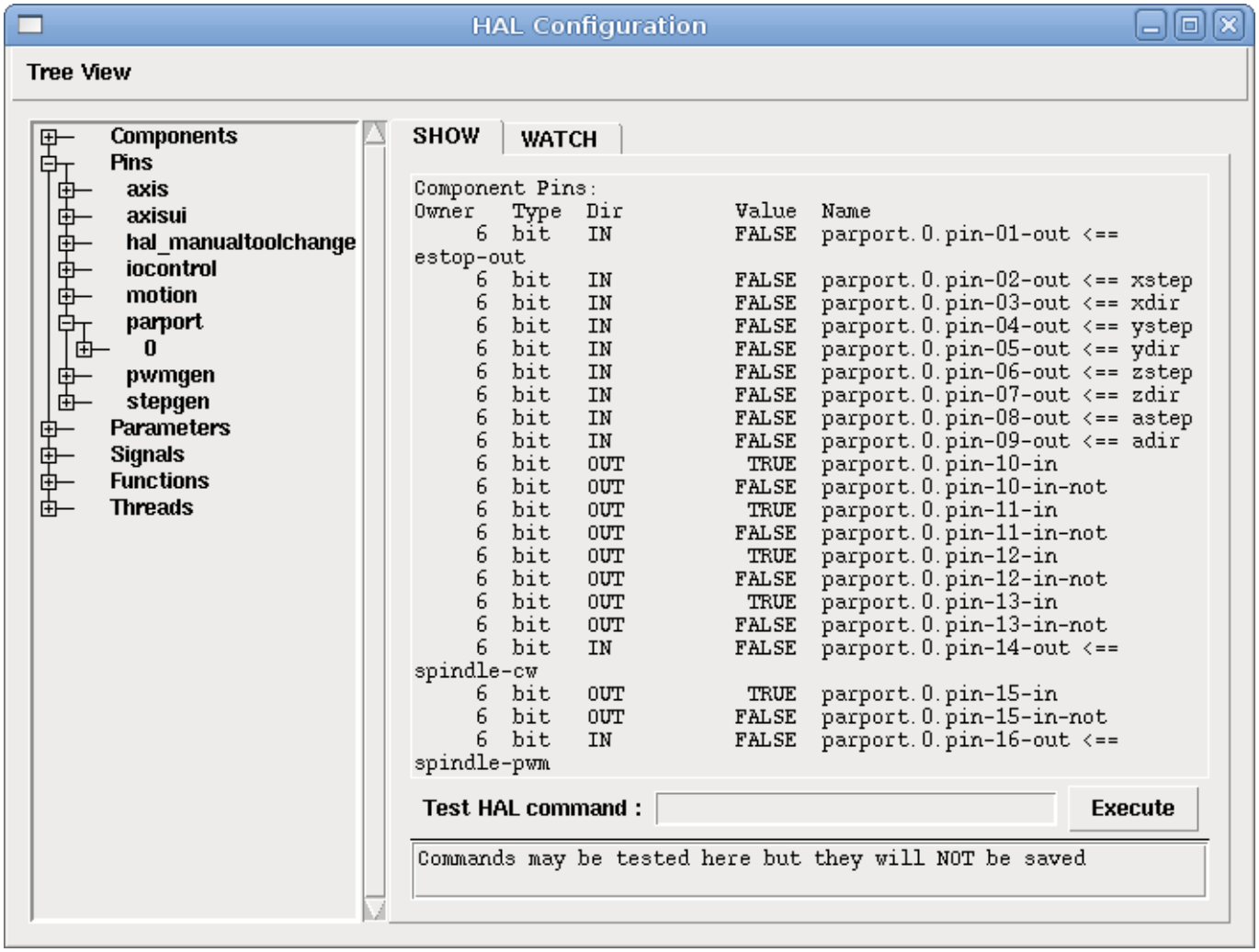


Figura 14.2: Ventana de configuración HAL

14.2.1.1. loadrt

El comando *loadrt* carga componentes HAL en tiempo real. Las funciones de los componentes de tiempo real deben agregarse a un hilo para ejecutarse a la velocidad de ese hilo. No puede cargar un componente de espacio de usuario en el espacio de tiempo real.

Sintaxis y ejemplo:

```
loadrt <componente> <opciones>

loadrt mux4 count=1
```

14.2.1.2. addf

Sintaxis y ejemplo:

```
addf <función> <thread>

addf mux4.0 servo-thread
```

Agrega la función *funct* al hilo *thread*. El valor predeterminado es agregar la función en el orden en que están en el archivo. Si se especifica *posición*, agrega la función en ese punto del hilo. Una posición negativa significa posición con respecto al final del hilo. Por ejemplo, *1* es al inicio del hilo, *-1* es al final de el hilo, *-3* es la tercera desde el final.

Es importante cargar algunas funciones en un orden determinado, como por ejemplo las funciones de lectura y escritura de puerto paralelo. El nombre de la función suele ser el nombre del componente más un número. En el siguiente ejemplo se carga el componente *or2* y *show function* muestra el nombre de la función or2

```
$ halrun
halcmd: loadrt or2
halcmd: show function
Exported Functions:
Owner   CodeAddr  Arg       FP   Users  Name
 00004  f8bc5000  f8f950c8  NO       0   or2.0
```

Debe agregar una función de un componente HAL en tiempo real a un hilo para que la función se ejecute a la velocidad del hilo.

Por lo general hay dos hilos, como se muestra en este ejemplo. Algunos componentes usan punto flotante matemático y se debe agregar a un hilo que admita punto flotante. La columna *FP* indica si la matemática de coma flotante es compatible con ese hilo.

```
$ halrun
halcmd: loadrt motmod base_period_nsec=55555 servo_period_nsec=1000000 num_joints=3
halcmd: show thread
Realtime Threads:
     Period  FP     Name               (     Time, Max-Time )
     995976  YES          servo-thread (        0,        0 )
      55332  NO            base-thread (        0,        0 )
```

- base-thread (el hilo de alta velocidad): este hilo maneja elementos que necesita una respuesta rápida, como hacer pulsos de paso o leer y escribir del/al puerto paralelo. No es compatible con matemática de coma flotante.
- servo-thread (el hilo de baja velocidad): este hilo maneja los elementos que puede tolerar una respuesta más lenta, como el controlador de movimiento, ClassicLadder, y el controlador de comandos de movimiento. Admite matemáticas de coma flotante.

14.2.1.3. loadusr

Sintaxis y ejemplos:

```
loadusr <componente> <opciones>

loadusr halui

loadusr -Wn spindle gs2_vfd -n spindle
```

Esto ultimo significa: *loadusr espera la carga del componente spindle, que es un componente gs2_vfd, nombrado como spindle*

El comando *loadusr* carga un componente HAL en espacio de usuario. Los programas en espacio de usuario tienen sus propios procesos separados que, opcionalmente, se comunican con otros componentes HAL a través de pines y parámetros. No puede cargar componentes de tiempo real en el espacio de usuario.

Las banderas pueden ser una o más de las siguientes:

-W	Esperar a que el componente esté listo. Se supone que el componente tienen el mismo nombre que el primer argumento del comando.
-Wn<nombre>	Esperar al componente, que tendrá el <nombre> dado. Esto solo se aplica si el componente tiene una opción de nombre.
-w	Esperar a que el programa termine.
-i	Ignorar el valor de retorno del programa (con -w).
-n	Nombrar un componente cuando es una opción válida para ese componente.

14.2.1.4. net

Sintaxis y ejemplo:

```
net nombre-señal nombre-pin <flecha opcional> <segundo nombre-pin opcional>

net home-x joint.0.home-sw-in <= parport.0.pin-11-in
```

El comando *net* crea una *conexión* entre una señal y uno o más pines. Si la señal no existe, *net* crea la nueva señal, evitando el uso del comando *newsig*. Las flechas de dirección opcionales <=, => y <=> sirven para que sea más fácil seguir la lógica al leer una línea de comando *net*, pero no son utilizadas por el propio comando. Las flechas de dirección debe estar separadas por un espacio de los nombres de los pines.

En el ejemplo, *home-x* es el nombre de la señal, *joint.0.home-sw-in* es un pin *Dirección IN*, <= es la flecha de dirección opcional, y *parport.0.pin-11-in* es un pin *Dirección OUT*. Esto puede parecer confuso pero las etiquetas de entrada y salida para un pin de puerto paralelo indican la forma física en la que el pin funciona, no como se maneja en HAL.

Se puede conectar un pin a una señal, si obedece las siguientes reglas:

- Un pin IN siempre se puede conectar a una señal.
- Un pin IO se puede conectar si no hay un pin OUT conectado a la señal.
- Un pin OUT se puede conectar solo si no hay otros pines OUT o IO en la señal

El mismo *nombre-señal* se puede usar en múltiples comandos *net* para conectar pines adicionales, siempre que se obedezcan las reglas anteriores.

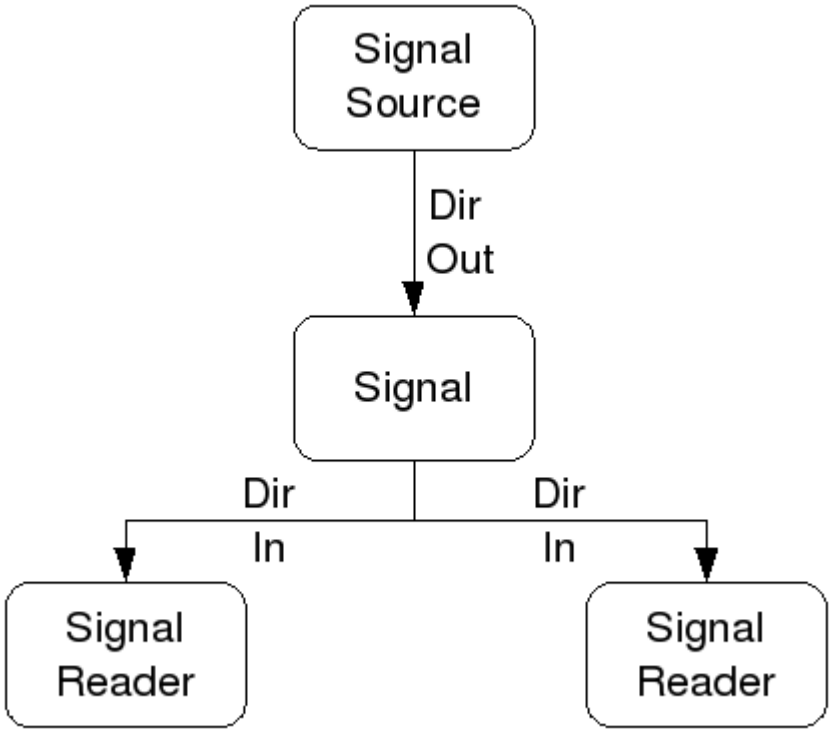


Figura 14.3: Dirección de señal

El siguiente ejemplo muestra la señal xStep, siendo la fuente stepgen.0.out, y con dos lectores, parport.0.pin-02-out y parport.0.pin-08-out. Básicamente, el valor de stepgen.0.out se envía a la señal xStep y ese valor se envía a parport.0.pin-02-out y parport.0.pin-08-out.

```
#   señal     fuente            destino1             destino2
net xStep stepgen.0.out => parport.0.pin-02-out parport.0.pin-08-out
```

Dado que la señal xStep ya contiene el valor de stepgen.0.out (la fuente) puede usar la misma señal nuevamente para enviar el valor a otro lector. Para hacer esto, simplemente use la señal, con los nuevos lectores, en otra línea.

```
net xStep => parport.0.pin-02-out
```

Pines I/O Un pin de E/S, como encoder.N.index-enable, se puede leer o establecer tal como lo permita el componente.

14.2.1.5. setp

Sintaxis y un ejemplo:

```
setp <nombre-pin/parámetro> <valor>

setp parport.0.pin-08-out TRUE
```

El comando *setp* establece el valor de un pin o parámetro. Los valores validos dependerán del tipo de pin o parámetro. Es un error si los tipos de datos no coinciden.

Algunos componentes tienen parámetros que deben establecerse antes de su uso. Los parámetros se pueden configurar antes de usarse o mientras se ejecuta el componente, según sea necesario. No puede usar *setp* en un pin que está conectado a una señal.

14.2.1.6. sets

El comando *sets* establece el valor de una señal.

Sintaxis y un ejemplo:

```
sets <nombre-señal> <valor>

net mysignal and2.0.in0 pyvcp.my-led

sets mysignal 1
```

Es un error si:

- El nombre de la señal no existe
- La señal ya tiene un escritor
- El valor no es el tipo correcto para la señal

14.2.1.7. unlinkp

El comando *unlinkp* desvincula un pin de la señal conectada. Si no se conectó el pin a una señal antes de ejecutar el comando, no sucede nada. No es necesario el nombre de la señal; el pin quedara aislado de cualquier señal. El comando *unlinkp* es útil para la resolución de problemas.

Sintaxis y ejemplo:

```
unlinkp <nombre-pin>

unlinkp parport.0.pin-02-out
```

14.2.1.8. Comandos obsoletos

Los siguientes comandos están desaconsejados y pueden eliminarse en futuras versiones. Cualquier nueva configuración debe usar el comando net. Estos comandos están incluidos para que las configuraciones más antiguas sigan funcionando.

linksp El comando *linksp* crea una *conexión* entre una señal y un pin.

Sintaxis y un ejemplo:

```
linksp <nombre-señal> <nombre-pin>
linksp X-step parport.0.pin-02-out
```

El comando *net* ha reemplazado al comando *linksp*.

linkps El comando *linkps* crea una *conexión* entre un pin y una señal. Es lo mismo que linksp pero los argumentos están invertidos.

Sintaxis y un ejemplo:

```
linkps <nombre-pin> <nombre-señal>

linkps parport.0.pin-02-out X-Step
```

El comando *net* ha reemplazado el comando *linkps*.

newsig el comando *newsig* crea una nueva señal HAL con el nombre <signame> y el tipo de datos <tipo>. El tipo debe ser *bit*, *s32*, *u32* o *float*. Es un error si *signame* ya existe.

Sintaxis y un ejemplo:

```
newsig <signame> <tipo>

newsig Xstep bit
```

Se puede encontrar más información en el manual de HAL o en las páginas man de halrun.

14.2.2. Datos HAL

14.2.2.1. Bit

Un valor de bit puede estar activado o desactivado.

- valores de bit = true o 1 y false o 0 (True, TRUE, true, son todos válidos)

14.2.2.2. Float

Un "float" es un número de punto flotante. En otras palabras, el punto decimal puede moverse según sea necesario.

- valores de float = un valor de punto flotante de 64 bits, con aproximadamente 53 bits de resolución y más de 1000 bits de rango dinámico.

Para obtener más información sobre los números de punto flotante, consulte:

http://en.wikipedia.org/wiki/Floating_point

14.2.2.3. s32

Un número *s32* es un número entero que puede tener un valor negativo o positivo.

- valores de s32 = números enteros de -2147483648 a 2147483647

14.2.2.4. u32

Un número *u32* es un número entero que es positivo solamente.

- valores u32 = números enteros de 0 a 4294967295

14.2.3. Archivos HAL

Si utilizó el Asistente de configuración de Steppers para generar su configuración, tendrá hasta tres archivos HAL en su directorio de configuración.

- mi-mill.hal (si ha llamado a su configuración *mi-mill*) Este archivo será cargado el primero y no debe cambiarse si usó el Asistente de configuración de Stepper.
- custom.hal Este archivo se carga a continuación y antes de que se cargue la GUI. En él se ponen los comandos HAL personalizados que se quieren cargar antes de que sea cargada la GUI.
- custom_postgui.hal Este archivo se carga después de que se cargue la GUI. Es donde se colocan los comandos HAL personalizados que desea cargar después de la carga de la GUI. Cualquier comando HAL que use widgets pyVCP debe ser colocado aquí.

14.2.4. Componentes HAL

A cada componente HAL, cuando es creado, se le agregan automáticamente dos parámetros. Estos parámetros permiten monitorizar el tiempo de ejecución de un componente.

`.time`

`.tmax`

time es la cantidad de ciclos de CPU necesarios para ejecutar la función.

tmax es la cantidad máxima de ciclos de CPU necesarios para ejecutar la función. *tmax* es un parámetro de lectura/escritura para que el usuario pueda configurarlo a 0 para deshacerse de la primera inicialización en la ejecución de la función.

14.2.5. Componentes lógicos

HAL contiene varios componentes lógicos en tiempo real. Un componentes lógico sigue una *tabla de verdad* que indica cuál es el resultado para cualquier entrada dada. Normalmente, son manipuladores de bits y siguen la lógica eléctrica de tablas de verdad de puertas lógicas.

14.2.5.1. and2

El componente *and2* es una puerta *and* de dos entradas. La tabla de verdad que sigue muestra el resultado en función de cada combinación de entrada.

Sintaxis

```
and2 [count=N] | [nombres=nombre1[, nombre2 ...]]
```

Funciones

```
and2.n
```

Pines

```
and2.N.in0 (bit, in)
and2.N.in1 (bit, in)
and2.N.out (bit, out)
```

Tabla de verdad

in0	in1	out
False	False	False
True	False	False
False	True	False
True	True	True

14.2.5.2. not

El componente *not* es un inversor.

Sintaxis

```
not [count=n] | [nombres=nombre1[,nombre2...]]
```

Funciones

```
not.all
not.n
```

Pines

```
not.n.in (bit, in)
not.n.out (bit, out)
```

Tabla de verdad

in	out
True	False
False	True

14.2.5.3. or2

El componente *or2* es una puerta OR de dos entradas.

Sintaxis

```
or2 [count=n] | [nombres=nombre1[,nombre2...]]
```

Funciones

```
or2.n
```

Pines

```
or2.n.in0 (bit, in)
or2.n.in1 (bit, in)
or2.n.out (bit, out)
```

Tabla de verdad

in0	in1	out
True	False	True
True	True	True
False	True	True
False	False	False

14.2.5.4. xor2

El componente *xor2* es una puerta XOR de dos entradas (O exclusivo).

Sintaxis

```
xor2 [count=n] | [nombres=nombre1[,nombre2...]]
```

Funciones

```
xor2.n
```

Pines

```
xor2.n.in0 (bit, in)
xor2.n.in1 (bit, in)
xor2.n.out (bit, out)
```

Tabla de verdad

in0	in1	out
True	False	True
True	True	False
False	True	True
False	False	False

14.2.5.5. Ejemplos de lógica

Ejemplo *and2* que conecta dos entradas a una salida.

```
loadrt and2 count=1

addf and2.0 servo-thread
```

```
net my-sigin1 and2.0.in0 <= parport.0.pin-11-in

net my-sigin2 and2.0.in1 <= parport.0.pin-12-in

net both-on parport.0.pin-14-out <= and2.0.out
```

En el ejemplo anterior, se carga una sola copia (count=1) de and2 en el espacio de tiempo real y se agrega (addf) al hilo servo. Despues, el pin 11 del puerto paralelo es conectado al bit 0 de la puerta and2.0 y el pin 12 se conecta al bit 1 de la puerta and2.0. Por último, se conecta el bit de salida del componente (and2.0.out) al pin 14 del puerto paralelo. Para ello se crea y usa la señal both-on. Entonces, siguiendo la tabla de verdad para and2, si el pin 11 y el pin 12 están encendidos, el pin de salida 14 estará encendido.

14.2.6. Componentes de conversión

14.2.6.1. weighted_sum

weighted_sum convierte un grupo de bits en un entero. La conversión es la suma de los "pesos" de los bits que valen 1, mas cualquier offset. El peso del bit m-ésimo es 2ˆm. Esto es similar a un binario codificado en decimal pero con más opciones. El bit *hold* (retención) detiene el procesamiento; si la entrada cambia, la *suma* no cambia.

La siguiente sintaxis se usa para cargar el componente weighted_sum.

```
loadrt weighted_sum wsum_sizes=tamaño[,tamaño,...]
```

Crea grupos de suma ponderada, cada uno con la cantidad dada de bits de entrada (size).

Para actualizar *weighted_sum* se necesita adjuntar la funcion *process_wsums* a un hilo.

```
addf process_wsums servo-thread
```

Esta funcion actualiza el componente weighted_sum.

En el siguiente ejemplo, recortado de la ventana HAL Configuration en Axis, Los bits *0* y *2* son verdaderos y no hay offset. El *peso* del bit 0 es 1 y el *peso* del bit 2 es 4, por lo que la suma es 5.

suma ponderada

```
Component Pins:
Owner   Type  Dir         Value  Name
    10  bit   In           TRUE  wsum.0.bit.0.in
    10  s32   I/O             1  wsum.0.bit.0.weight
    10  bit   In          FALSE  wsum.0.bit.1.in
    10  s32   I/O             2  wsum.0.bit.1.weight
    10  bit   In           TRUE  wsum.0.bit.2.in
    10  s32   I/O             4  wsum.0.bit.2.weight
    10  bit   In          FALSE  wsum.0.bit.3.in
    10  s32   I/O             8  wsum.0.bit.3.weight
    10  bit   In          FALSE  wsum.0.hold
    10  s32   I/O             0  wsum.0.offset
    10  s32   Out             5  wsum.0.sum
```

14.3. HAL TWOPASS

14.3.1. TWOPASS

Desde LinuxCNC 2.5 se admite el procesamiento TWOPASS de archivos de configuración hal. Esto puede ayudar a la modularización y legibilidad de los archivos hal. (Los archivos Hal se especifican en un archivo ini en la sección HAL como [HAL]HALFILE = nombre de archivo).

Normalmente, un conjunto de uno o más archivos de configuración hal deben usar una única línea, en uno de los archivos, para cargar un módulo de kernel que puede manejar múltiples instancias de un componente. Por ejemplo, si usted usa tres componentes AND de dos entradas (and2) en tres lugares diferentes en su configuración, bastará una sola línea en algún lugar para especificarlo:

```
loadrt and2 count=3
```

resultando los componentes and2.0, and2.1 y and2.2.

Las configuraciones son más legibles si especifican con names=opción en aquellos componentes donde se admita, por ejemplo:

```
loadrt and2 names=aa,ab,ac
```

dando como resultado las instancias de componente aa, ab y ac.

Al realizar un seguimiento de los componentes y sus nombres, puede darse un problema de mantenimiento cuando agrega (o elimina) un componente. Debe encontrar y actualizar la directiva loadrt única aplicable al componente.

El procesamiento TWOPASS se habilita al incluir un parámetro de archivo ini en la sección [HAL]:

```
[HAL]

TWOPASS = cualquier-cadena
```

Donde "cualquier-cadena" puede ser cualquier cadena que no sea nula. Con esta configuración, puede tener múltiples especificaciones como:

```
loadrt and2 names = aa
...
loadrt and2 names = ab, ac
...
loadrt and2 names = ad
```

Estos comandos pueden aparecer en HALFILES diferentes. Los HALFILES se procesan en el orden de su aparición en el archivo ini.

La opción TWOPASS se puede especificar con opciones para ampliar la salida para depuración (verbose) o para evitar la eliminación de archivos temporales (nodelete). Las opciones están separadas por comas.

Ejemplo:

```
[HAL]

TWOPASS = on, verbose, nodelete
```

Con el procesamiento de TWOPASS, todos los [HAL]HALFILES se leen primero y se acumulan las apariciones múltiples de las directivas loadrt para cada módulo. No se ejecutan comandos hal en esta pasada inicial.

Después de la pasada inicial, los módulos se cargan automáticamente con un número de instancias igual al número total, resultado de la opcion *count* =, o con todos los nombres individuales especificados al usar la opcion *names* =.

Luego se hace una segunda pasada para ejecutar todas las demás instrucciones HAL especificadas en los HALFILES. Los comandos addf que asocian las funciones de un componente con los hilos de ejecución, son ejecutadas en el orden de aparición con otros comandos durante esta segunda pasada.

Si bien puede usar las opciones count = o names =, éstas son mutuamente excluyentes - solo se puede especificar un tipo para un módulo dado.

El procesamiento TWOPASS es más efectivo cuando se usan opciones *names* =. Esta opción le permite proporcionar nombres únicos que son mnemotécnicos o relevantes para la configuración. Por ejemplo, si utiliza un componente derivativo para estimar la velocidades y aceleraciones en cada coordenada (x, y, z), usando el método *count* = dará nombres de componentes poco significativos, como ddt.0, ddt.1, ddt.2, etc.

Alternativamente, usando los *names* =, como:

```
loadrt dd names = xvel, yvel, zvel
...
loadrt dd names = xacel, yacel, zacel
```

resultan componentes llamados xvel, yvel, zvel, xacel, yacel, zacel.

Muchos componentes suministrados con la distribución se crean con una utilidad de creacion de componentes y soportan la opción names =. Estos incluyen los componentes lógicos comunes que son el "*pegamento*" de muchas configuraciones hal.

Los componentes creados por el usuario que usan la utilidad soportan automáticamente la opción names =. Además de los componentes generados con la utilidad, muchos otros componentes admiten la opción names =, entre ellos:at_pid, encoder, encoder_ratio, pid, siggen, y sim_encoder.

El procesamiento en dos pasos ocurre antes de la carga de la interfaz gráfica de usuario. Cuando se utiliza un [HAL]POSTGUI_HALFILE es conveniente colocar todo las declaraciones loadrt para los componentes necesarios en un halfile que se carge con anterioridad.

Ejemplo de una sección HAL cuando se usa un POSTGUI_HALFILE:

```
[HAL]

TWOPASS = on
HALFILE = core_sim.hal
HALFILE = sim_spindle_encoder.hal
HALFILE = axis_manualtoolchange.hal
HALFILE = simulated_home.hal
HALFILE = load_for_postgui.hal <- líneas loadrt para componentes en postgui.hal

POSTGUI_HALFILE = postgui.hal
HALUI = halui
```

14.3.2. Excluyendo archivos .hal

El procesamiento TWOPASS convierte los archivos *.hal* en archivos *.tcl* equivalentes y utiliza haltcl para encontrar los comandos loadrt y addf para acumularlos y consolidar su uso. Se esperan parámetros de loadrt que se ajustan a los parámetros *nombres* = (o *count* =) aceptados por el generador de componentes HAL (*halcompile*). Los elementos de parámetros más complejos incluidos en componentes hal especializados
no puede ser manejado adecuadamente.

Un archivo *.hal* puede ser excluido del procesamiento TWOPASS incluyendo una línea de comentarios especial en cualquier lugar del archivo. La línea de comentario debe comenzar con la cadena: *#NOTWOPASS*. Los archivos especificados con este comentario son tratados por halcmd usando las opciones *-k* (seguir adelante en caso de fallo) y *-v* (verbose).

Esta disposición de exclusión se puede utilizar para aislar problemas o para cargar cualquier componente hal que no requiere o no se beneficia del procesamiento TWOPASS que maneja múltiples instancias de componentes loadrt.

Ejemplo de archivo *.hal* excluido:

```
$ cat twopass_excluded.hal
# El siguiente 'comentario mágico' hace que este archivo
# sea excluido del procesamiento de dos fases:
# NOTWOPASS

loadrt mycomponent parm1="abc def" parm2=ghi
show pin mycomponent
```

nota
El caso y los espacios en blanco dentro de # NOTWOPASS se ignoran.

14.3.3. Post GUI

Algunas GUIs soportan halfiles que se procesan después de que se inicie la GUI, permitiendo conectar los pines hal que son creados por la propia GUI. Cuando se utiliza un halfile postgui con el procesamiento TWOPASS, incluya todos los elementos loadrt para los componentes agregados por los halfiles postgui en un halfile separado que se procese antes que la GUI. Los comandos addf también pueden ser incluido en el archivo. Ejemplo:

```
[HAL]
HALFILE = file_1.hal
...
HALFILE = file_n.hal
HALFILE = file_with_all_loads_for_postgui.hal
...
POSTGUI_HALFILE = the_postgui_file.hal
```

14.3.4. Ejemplos

Se incluyen ejemplos de uso de TWOPASS para simulador en los directorios:

`configs/sim/axis/twopass/`

`configs/sim/axis/simtcl/`

14.4. Tutorial HAL

14.4.1. Introducción

La configuración va de la teoría al dispositivo, es decir, al dispositivo HAL. Para aquellos que saben algo de programación informática, esta sección es el *Hola mundo* de HAL.

Halrun se puede usar para crear un sistema de trabajo. Es una línea de comando o herramienta de archivo de texto para configuración y sintonización. Los siguientes ejemplos ilustran su configuración y funcionamiento.

14.4.1.1. Notación

Los comandos del terminal se muestran sin la indicación del sistema, excepto cuando se esta corriendo *HAL*. La ventana del terminal está en el Menu de Aplicacione en Debian, o en *Applications/Accessories* desde la barra de menú principal de Ubuntu.

Ejemplo de comando de terminal

```
mi@computer:~linuxcnc$ halrun
(se mostrará la siguiente línea)
halcmd:
(el indicador halcmd: se mostrará cuando se ejecuta HAL)
halcmd: loadrt debounce
halcmd: show pin
```

14.4.1.2. Completado por tabulador

Su versión de halcmd puede incluir el completado por tabulador. En lugar de completar los nombres de archivo como lo hace un shell, completa los comandos con identificadores HAL. Tendrá que escribir suficientes letras para una coincidencia única. Intente presionar el tabulador después de iniciar un comando HAL (las lineas con > son entradas):

Completado por tabulador

```
>halcmd: loa<TAB>
halcmd: load
>halcmd: loadrt
>halcmd: loadrt deb<TAB>
halcmd: loadrt debounce
```

14.4.1.3. El entorno RTAPI

RTAPI significa Interfaz de programación de Aplicaciones de Tiempo Real. Muchos componentes HAL funcionan en tiempo real, y todos los componentes HAL almacenan datos en memoria compartida para que los componentes en tiempo real puedan acceder a ella. Un nucleo Linux "normal" no no es compatible con la programación en tiempo real o el tipo de memoria compartida que HAL necesita. Afortunadamente, hay sistemas operativos en tiempo real (RTOS) que proporcionar las extensiones necesarias para Linux. Lamentablemente, cada RTOS hace las cosas de manera un poco diferente.

Para abordar estas diferencias, el equipo de LinuxCNC ideó RTAPI, que proporciona una forma consistente para que los programas hablen con los RTOS. Si usted es un programador que quiere trabajar en el interior de LinuxCNC, es posible que desee estudiar *linuxcnc/src/rtapi/rtapi.h* para comprender la API. Pero si es un usuario normal, todo lo que necesita saber acerca de RTAPI es que debe ser cargado (tambien el RTOS) en la memoria de su computadora antes de hacer algo con HAL.

14.4.2. Un ejemplo simple

14.4.2.1. Cargando un componente

Para este tutorial, vamos a suponer que ha tenido éxito instalado el Live CD o, si usa RIP, [2] invocó el script *rip-environment* para preparar su shell. En ese caso, todo lo que necesita hacer es cargar los módulos RTOS y RTAPI necesarios en la memoria. Simplemente ejecute el siguiente comando desde una ventana de terminal:

Cargando HAL

```
cd linuxcnc
halrun
halcmd:
```

Con el sistema operativo en tiempo real y RTAPI cargados, podemos pasar al primer ejemplo. Observe que el mensaje ahora se muestra como *halcmd:*. Esto se debe a que los comandos posteriores se interpretarán como comandos HAL, no como comandos shell.

Para el primer ejemplo, usaremos un componente HAL llamado *siggen*, que es un generador de señal simple. Una descripción completa del componente *siggen* se puede encontrar en la sección Siggen de este manual. Es un componente en tiempo real, implementado como un módulo kernel de Linux. Para cargar *siggen* use el comando HAL *loadrt*.

Loading siggen

```
halcmd: loadrt siggen
```

14.4.2.2. Examinando el HAL

Ahora que el módulo está cargado, es hora de ver *halcmd*, la herramienta de línea de comandos utilizada para configurar HAL. Este tutorial introduce algunas características halcmd. Para una descripción más completa pruebe *man halcmd*, o vea la referencia en la sección Hal Commands de este documento. La primera función halcmd a ver es el comando *show*. Este comando muestra información sobre el estado actual de HAL. Para mostrar todos los componentes instalados:

Mostrar componentes

[2] Ejecutar en el lugar, cuando los archivos fuente se han descargado a un directorio de usuario y se han compilado alli.

```
halcmd: show comp

Loaded HAL Components:
ID      Type  Name                                PID   State
    3   RT    siggen                                    ready
    2   User  halcmd2177                          2177  ready
```

Como *halcmd* es un componente HAL, siempre aparecerá en la lista. El número después de halcmd en la lista de componentes es el ID del proceso. Es posible ejecutar más de una copia de halcmd al mismo tiempo (en diferentes ventanas, por ejemplo), por lo que el PID se agrega al final del nombre para hacerlos únicos. La lista también muestra el componente *siggen* que instalamos en el paso anterior. La *RT* en *Type* indica que ese *siggen* es un componente en tiempo real. El *User* en *Type* indica es un componente de espacio de usuario.

A continuación, veamos qué pines pone *siggen* a nuestra disposición:

Mostrar los pines

```
halcmd: show pin

Component Pins:
Owner   Type   Dir        Value  Name
     3  float  IN             1  siggen.0.amplitude
     3  bit    OUT        FALSE  siggen.0.clock
     3  float  OUT            0  siggen.0.cosine
     3  float  IN             1  siggen.0.frequency
     3  float  IN             0  siggen.0.offset
     3  float  OUT            0  siggen.0.sawtooth
     3  float  OUT            0  siggen.0.sine
     3  float  OUT            0  siggen.0.square
     3  float  OUT            0  siggen.0.triangle
```

Este comando muestra todos los pines en el HAL actual. Un sistema complejo podría tener docenas o cientos de pines. Pero en este momento solo hay nueve pines. De estos pines, ocho son de punto flotante y uno es de tipo bit (booleano). De ellos, seis llevan datos fuera del componente *siggen* (OUT) y tres se usan para transferir configuraciones al componente (IN). Como aún no hemos ejecutado el código contenido dentro del componente, algunos de los pines tienen un valor cero.

El siguiente paso es mirar los parámetros:

1. Mostrar parámetros

```
halcmd: show param

Parameters:
Owner   Type  Dir        Value   Name
     3  s32   RO             0   siggen.0.update.time
     3  s32   RW             0   siggen.0.update.tmax
```

El comando *show param* muestra todos los parámetros en HAL. Ahora mismo cada parámetro tiene el valor predeterminado que se le dio cuando el componente fue cargado. Tenga en cuenta la columna etiquetada *Dir*. Los parámetros etiquetados *-W* son grabables; nunca cambian por el componente en sí mismo. Por contra, están destinados a ser cambiados por el usuario para controlar el componente. Veremos cómo hacerlo más tarde. Los parámetros etiquetados *R-* (RO) son parámetros de solo lectura. Solo pueden ser cambiados por el propio componente. Finalmente, los parámetros etiquetados *RW* son parámetros de lectura-escritura. Eso significa que son cambiados por el componente, pero también puede ser cambiados por el usuario. Nota: los parámetros *siggen.0.update.time* y *siggen.0.update.tmax* son para propositos de depuración y no serán cubiertos en esta sección.

La mayoría de los componentes en tiempo real exportan una o más funciones para hacer correr realmente el código en tiempo real que contienen. Veamos qué función(es) a exportado *siggen*:

Mostrar funciones

```
halcmd: show funct

Exported Functions:
Owner   CodeAddr  Arg       FP   Users  Name
 00003  f801b000  fae820b8  YES      0   siggen.0.update
```

El componente siggen exportó una sola función. Requiere punto flotante. Actualmente no está vinculado a ningún subproceso, por lo que *users* es cero.

14.4.2.3. Hacer correr el código en tiempo real

Para ejecutar realmente el código contenido en la función *siggen.0.update*, necesitamos un hilo en tiempo real El componente llamado *threads* se usa para crear un nuevo hilo. Vamos a crear un hilo llamado *test-thread* con un período de 1 ms (1,000 us o 1,000,000 ns):

```
halcmd: loadrt threads name1=test-thread period1=1000000
```

Veamos si funcionó:

Mostrar hilos

```
halcmd: show thread

Realtime Threads:
     Period  FP     Name               (     Time, Max-Time )
     999855  YES           test-thread (        0,        0 )
```

Correcto. El período no es exactamente 1,000,000 ns debido a limitaciones del hardware, pero tenemos un hilo que se ejecuta aproximadamente a la velocidad correcta, y que puede manejar funciones de coma flotante. El siguiente paso es conectar la función al hilo:

Agregar función

```
halcmd: addf siggen.0.update test-thread
```

Hasta ahora, hemos estado usando *halcmd* solo para ver el HAL. Sin embargo, esta vez usamos el comando *addf* (añadir función) para cambiar algo en HAL. Hemos dicho a *halcmd* que agregue la función *siggen.0.update* al hilo *test-thread*, y si miramos de nuevo la lista de hilos, vemos que ha tenido éxito:

```
halcmd: show thread

Realtime Threads:
     Period  FP     Name               (     Time, Max-Time )
     999855  YES          test-thread   (        0,        0 )
                  1 siggen.0.update
```

Se necesita un paso más antes de que el componente *siggen* comience a generar señales. Cuando se inicia HAL por primera vez, el(los) hilo(s) no se están ejecutando realmente. Esto es para permitirle configurar completamente el sistema antes de que comience el código en tiempo real. Una vez que usted está contento con la configuración, puede iniciar el código en tiempo real asi:

```
halcmd: start
```

Ahora el generador de señal está funcionando. Veamos sus pines de salida:

```
halcmd: show pin

Component Pins:
Owner   Type  Dir         Value  Name
     3  float IN              1  siggen.0.amplitude
```

```
     3  bit   OUT         FALSE  siggen.0.clock
     3  float OUT    -0.1640929  siggen.0.cosine
     3  float IN             1  siggen.0.frequency
     3  float IN             0  siggen.0.offset
     3  float OUT    -0.4475303  siggen.0.sawtooth
     3  float OUT     0.9864449  siggen.0.sine
     3  float OUT            -1  siggen.0.square
     3  float OUT    -0.1049393  siggen.0.triangle
```

Si volvamos a mirar:

```
halcmd: show pin

Component Pins:
Owner   Type  Dir         Value  Name
     3  float IN             1  siggen.0.amplitude
     3  bit   OUT         FALSE  siggen.0.clock
     3  float OUT     0.0507619  siggen.0.cosine
     3  float IN             1  siggen.0.frequency
     3  float IN             0  siggen.0.offset
     3  float OUT     -0.516165  siggen.0.sawtooth
     3  float OUT     0.9987108  siggen.0.sine
     3  float OUT            -1  siggen.0.square
     3  float OUT    0.03232994  siggen.0.triangle
```

Las salidas de seno, coseno, diente de sierra y triángulo estan cambiando constantemente. La salida cuadrada también está funcionando, sin embargo simplemente cambia de +1.0 a -1.0 en cada ciclo.

14.4.2.4. Cambiar los parámetros

El verdadero poder de HAL es que puede cambiar cosas. Por ejemplo, nosotros podemos usar el comando *setp* para establecer el valor de un parámetro. Vamos a cambiar la amplitud del generador de señal de 1.0 a 5.0:

Set Pin

```
halcmd: setp siggen.0.amplitude 5
```

Verifique nuevamente los parámetros y los pines

```
halcmd: show param

Parameters:
Owner   Type  Dir         Value  Name
     3  s32   RO           1754  siggen.0.update.time
     3  s32   RW          16997  siggen.0.update.tmax

halcmd: show pin

Component Pins:
Owner   Type  Dir         Value  Name
     3  float IN             5  siggen.0.amplitude
     3  bit   OUT         FALSE  siggen.0.clock
     3  float OUT     0.8515425  siggen.0.cosine
     3  float IN             1  siggen.0.frequency
     3  float IN             0  siggen.0.offset
     3  float OUT      2.772382  siggen.0.sawtooth
     3  float OUT     -4.926954  siggen.0.sine
     3  float OUT             5  siggen.0.square
     3  float OUT      0.544764  siggen.0.triangle
```

Observe que el valor del parámetro *siggen.0.amplitude* ha cambiado a 5, y que los pines tienen ahora valores más grandes.

14.4.2.5. Guardar la configuración HAL

La mayor parte de lo que hasta ahora hemos hecho con *halcmd* es simplemente ver cosas con el comando *show*. Sin embargo, dos de los comandos realmente han cambiado cosas. Cuando se diseñan sistemas más complejos con HAL, utilizaremos muchos comandos para configurar las cosas tal como las queremos. HAL retendrá esa configuración hasta que lo apaguemos. Pero ¿que hay sobre la próxima vez?. No queremos ingresar manualmente un grupo de comandos cada vez que queremos usar el sistema. Podemos salvar la configuración de todo el HAL con un solo comando:

Salvar

```
halcmd: save
# components
loadrt threads name1=test-thread period1=1000000
loadrt siggen
# pin aliases
# signals
# nets
# parameter values
setp siggen.0.update.tmax 14687
# realtime thread/function links
addf siggen.0.update test-thread
```

La salida del comando *save* es una secuencia de comandos HAL. Si se comienza con un HAL *vacío* y se ejecutan todos estos comandos, obtendrá la configuración que existía antes de emitir el comando *save*. Para guardar estos comandos para un uso posterior, simplemente redirija la salida a un archivo:

Guardar en un archivo

```
halcmd: save all saved.hal
```

14.4.2.6. Saliendo de halrun

Cuando haya terminado con su sesión HAL, escriba *exit* en *halcmd:*. Esto lo devolverá al prompt del sistema y cerrará la sesion HAL. No cierre simplemente la ventana de la terminal sin apagar adecuadamente la sesión HAL.

Salir de HAL

```
halcmd: exit
```

14.4.2.7. Restaurando la configuración HAL

Para restaurar la configuración HAL almacenada en *saved.hal* necesitamos ejecuta todos esos comandos HAL. Para hacer eso, usaremos la opción *-f <nombre de archivo>* que lee comandos de un archivo, e *-I* que muestra el indicador halcmd después de ejecutar los comandos:

Ejecutar un archivo guardado

```
halrun -I -f saved.hal
```

Observe que no hay un comando *start* en saved.hal. Es necesario emitirlo nuevamente (o editar saved.hal para agregarlo).

14.4.2.8. Eliminando HAL de la memoria

Si se produce un cierre inesperado de una sesión HAL, es posible que tenga que descartar HAL completamente antes de que pueda comenzar otra sesión. Para hacer esto, escriba el siguiente comando en una ventana de terminal.

Eliminando HAL

```
halrun -U
```

14.4.3. Halmeter

Puede construir sistemas HAL muy complejos sin tener que utilizar un interfaz gráfico. Sin embargo, es muy satisfactorio ver graficamente el resultado de su trabajo. La primera y más simple herramienta GUI para HAL es halmeter. Es un programa muy simple, que es el equivalente HAL de un práctico multímetro.

Usaremos el componente siggen nuevamente para verificar halmeter. Si acaba de terminar el ejemplo anterior, puede cargar *siggen* usando el archivo guardado. Si no, podemos cargarlo como lo hicimos antes:

```
halrun
halcmd: loadrt siggen
halcmd: loadrt threads name1=test-thread period1=1000000
halcmd: addf siggen.0.update test-thread
halcmd: start
halcmd: setp siggen.0.amplitude 5
```

En este punto, tenemos el componente siggen cargado y en ejecución. Es momento de comenzar con halmeter.

Arrancar Halmeter

```
halcmd: loadusr halmeter
```

La primera ventana que verá es la ventana *Seleccionar elemento a sondear*.

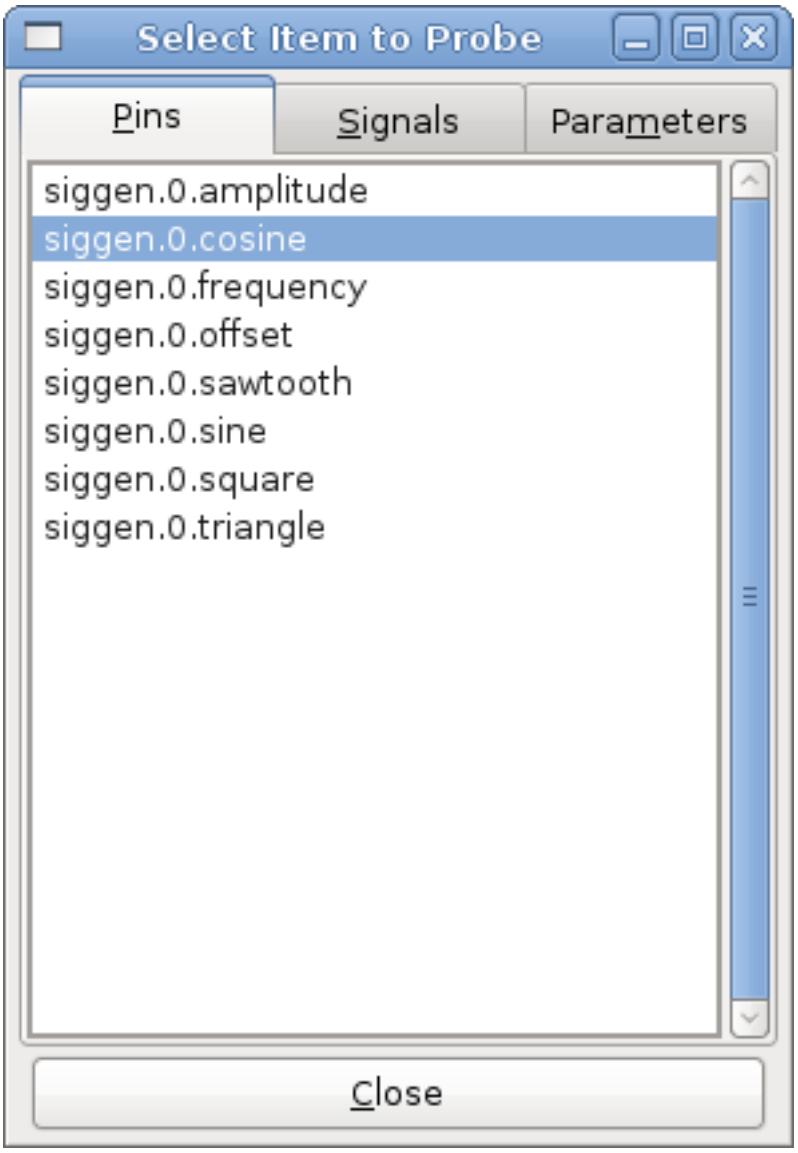


Figura 14.4: Ventana de selección de Halmeter

Este diálogo tiene tres pestañas. La primera pestaña muestra todos los pines HAL en el sistema. La segunda muestra todas las señales, y la tercera muestra todos los parámetros. Nos gustaría mirar primero el pin *siggen.0.cosine*. Haga clic en él y luego haga clic en el botón *Close*. El cuadro de diálogo de selección se cerrará, y el medidor se vera como en la siguiente figura.

Figura 14.5: Halmeter

Para cambiar lo que muestra el medidor presione el botón *Select* que devuelve la ventana de seleccion del elemento a sondear.

Debería ver el cambio de valor a medida que siggen genera su onda de coseno. Halmeter refresca su pantalla (su valor) aproximadamente 5 veces por segundo.

Para apagar halmeter, simplemente haga clic en el botón de salida.

Si desea ver más de un pin, señal o parámetro a la vez, puede lanzar más halmeters. La ventana de halmeter es intencionalmente muy pequeña para que se pueda tener muchas de ellas a la vez en la pantalla.

14.4.4. Ejemplo Stepgen

Hasta ahora solo hemos cargado un componente HAL. Pero toda la idea detrás de HAL es permitir cargar y conectar una serie de componentes simples para formar un sistema complejo. El siguiente ejemplo usará dos componentes.

Antes de que podamos comenzar a construir este nuevo ejemplo, queremos comenzar con una instancia limpia. Si se acaba de terminar uno de los ejemplos anteriores, necesitamos eliminar todos los componentes cargados y volver a cargar las bibliotecas RTAPI y HAL. Para la eliminacion, basta salir de HAL con el comando:

```
halcmd: exit
```

14.4.4.1. Instalación de los componentes

Ahora vamos a cargar el componente generador de impulsos de pasos. Para la descripción detallada de este componente, vease la sección del Manual del integrador. En este ejemplo usaremos el tipo de control de *velocidad* de stepgen. Por ahora, podemos omitir los detalles, y solo ejecutar los siguientes comandos.

```
halrun
halcmd: loadrt stepgen step_type=0,0 ctrl_type=v,v
halcmd: loadrt siggen
halcmd: loadrt threads name1=fast fp1=0 period1=50000 name2=slow period2=1000000
```

El primer comando, despues de halrun, carga dos generadores de pasos, ambos configurados para generar pasos tipo 0. El segundo comando carga a nuestro viejo amigo siggen, y el tercero crea dos hilos, uno rápido (que llamaremos *fast*) con un período de 50 microsegundos y uno lento (que llamaremos *slow*) con un período de 1 milisegundo. El rápido no admite (fp1=0) funciones de punto flotante.

Como antes, podemos usar *halcmd show* para echar un vistazo a HAL. Ahora tendremos muchos más pines y parámetros que antes:

```
halcmd: show pin

Component Pins:
Owner   Type  Dir         Value  Name
     4  float IN              1  siggen.0.amplitude
     4  bit   OUT         FALSE  siggen.0.clock
     4  float OUT             0  siggen.0.cosine
     4  float IN              1  siggen.0.frequency
     4  float IN              0  siggen.0.offset
     4  float OUT             0  siggen.0.sawtooth
     4  float OUT             0  siggen.0.sine
     4  float OUT             0  siggen.0.square
     4  float OUT             0  siggen.0.triangle
     3  s32   OUT             0  stepgen.0.counts
     3  bit   OUT         FALSE  stepgen.0.dir
     3  bit   IN          FALSE  stepgen.0.enable
     3  float OUT             0  stepgen.0.position-fb
     3  bit   OUT         FALSE  stepgen.0.step
     3  float IN              0  stepgen.0.velocity-cmd
     3  s32   OUT             0  stepgen.1.counts
```

```
     3  bit   OUT         FALSE  stepgen.1.dir
     3  bit   IN          FALSE  stepgen.1.enable
     3  float OUT             0  stepgen.1.position-fb
     3  bit   OUT         FALSE  stepgen.1.step
     3  float IN              0  stepgen.1.velocity-cmd


halcmd: show param

Parameters:
Owner   Type  Dir         Value  Name
     4  s32   RO              0  siggen.0.update.time
     4  s32   RW              0  siggen.0.update.tmax
     3  u32   RW     0x00000001  stepgen.0.dirhold
     3  u32   RW     0x00000001  stepgen.0.dirsetup
     3  float RO              0  stepgen.0.frequency
     3  float RW              0  stepgen.0.maxaccel
     3  float RW              0  stepgen.0.maxvel
     3  float RW              1  stepgen.0.position-scale
     3  s32   RO              0  stepgen.0.rawcounts
     3  u32   RW     0x00000001  stepgen.0.steplen
     3  u32   RW     0x00000001  stepgen.0.stepspace
     3  u32   RW     0x00000001  stepgen.1.dirhold
     3  u32   RW     0x00000001  stepgen.1.dirsetup
     3  float RO              0  stepgen.1.frequency
     3  float RW              0  stepgen.1.maxaccel
     3  float RW              0  stepgen.1.maxvel
     3  float RW              1  stepgen.1.position-scale
     3  s32   RO              0  stepgen.1.rawcounts
     3  u32   RW     0x00000001  stepgen.1.steplen
     3  u32   RW     0x00000001  stepgen.1.stepspace
     3  s32   RO              0  stepgen.capture-position.time
     3  s32   RW              0  stepgen.capture-position.tmax
     3  s32   RO              0  stepgen.make-pulses.time
     3  s32   RW              0  stepgen.make-pulses.tmax
     3  s32   RO              0  stepgen.update-freq.time
     3  s32   RW              0  stepgen.update-freq.tmax
```

14.4.4.2. Conexión de pines con señales

Lo que tenemos, de momento, son dos generadores de impulsos de pasos y un generador de señal. Es hora de crear algunas señales HAL para conectar los dos componentes. Se pretende que los dos generadores de impulsos de pasos conduzcan los ejes X e Y de una máquina. Queremos mover la mesa en círculos. Para hacer esto, enviaremos una señal coseno al eje X y una señal seno al eje Y. El módulo siggen crea las señales seno y coseno, pero necesitamos *cables* para conectar los módulos entre sí, de forma de los generadores creen los pasos al ritmo de los valores seno y coseno. En HAL, los *cables* se llaman *señales*. Necesitamos crear dos de ellas. Podemos llamarlos de cualquier forma, pero para este ejemplo serán *X-vel* y *Y-vel*. La señal *X-vel* está destinado a ejecutarse desde la salida coseno del generador de señales a la entrada de velocidad del primer generador de impulsos de paso. El primer paso es conectar la señal a la salida del generador de señales. Para conectar una señal a un pin usamos el comando net.

Comando net

```
halcmd: net X-vel <= siggen.0.cosine
```

Para ver el efecto del comando *net*, mostramos las señales nuevamente.

```
halcmd: show sig

Signals:
Type          Value  Name     (linked to)
float             0  X-vel <== siggen.0.cosine
```

Cuando una señal está conectada a uno o más pines, el comandos show lista pines inmediatamente después del nombre de la señal. La *flecha* muestra la dirección del flujo de datos; en este caso, los datos fluyen desde el pin *siggen.0.cosine* hasta la señal *X-vel*. Ahora conectemos *X-vel* a la entrada de velocidad de un generador de impulsos por pasos.

```
halcmd: net X-vel => stepgen.0.velocity-cmd
```

También podemos conectar la señal *Y-vel* del eje Y. Esta señal ira desde la salida seno del generador de señales a la entrada del segundo generador de impulsos de paso. El siguiente comando logra en una sola línea lo que dos comandos *net* lograron con *X-vel*.

```
halcmd: net Y-vel siggen.0.sine => stepgen.1.velocity-cmd
```

Ahora echemos un vistazo final a las señales y pines conectados a ellos.

```
halcmd: show sig

Signals:
Type          Value  Name     (linked to)
float             0  X-vel <== siggen.0.cosine
                            ==> stepgen.0.velocity-cmd
float             0  Y-vel <== siggen.0.sine
                            ==> stepgen.1.velocity-cmd
```

El comando *show sig* deja en claro exactamente cómo fluyen los datos en HAL. Por ejemplo, la señal *X-vel* proviene del pin *siggen.0.cosine*, y va al pin *stepgen.0.velocity-cmd*.

14.4.4.3. Configuración de la ejecución en tiempo real: hilos y funciones

Con la idea de que los datos fluyen a través de *cables* es bastante fácil de entender los pines y señales. Los hilos y las funciones son conceptos un poco más difíciles. Las funciones contienen las instrucciones de la computadora que en realidad hacen las cosas. El hilo es el método utilizado para hacer correr esas instrucciones cuando sea necesario. Primero veamos las funciones disponibles.

```
halcmd: show funct

Exported Functions:
Owner   CodeAddr  Arg       FP   Users  Name
 00004  f9992000  fc731278  YES      0   siggen.0.update
 00003  f998b20f  fc7310b8  YES      0   stepgen.capture-position
 00003  f998b000  fc7310b8  NO       0   stepgen.make-pulses
 00003  f998b307  fc7310b8  YES      0   stepgen.update-freq
```

En general, deberá consultar la documentación de cada componente para ver lo que hacen sus funciones. En este caso, la función *siggen.0.update* se usa para actualizar las salidas del generador de señales. Cada vez que se ejecuta, calcula los valores de las salidas seno, coseno, triángulo y cuadrado. Para sacar señales limpias, necesita ejecutarse a intervalos específicos.

Las otras tres funciones están relacionadas con los generadores de impulsos de pasos.

la primera, *stepgen.capture_position*, se usa para la realimentación de posición. Captura el valor de un contador interno que cuenta los pulsos de paso a medida que se generan. Suponiendo que no no se pierden pasos en sus motores, este contador podria indicar la posición.

La función principal del generador de impulsos de pasos es *stepgen.make_pulses*. Cada vez que se ejecuta *make_pulses*, decide si es hora de dar un nuevo paso y, si es así, establece las salidas en consecuencia. Para pulsos de paso suaves, debe ejecutarse con la mayor frecuencia posible. Debido a que necesita correr rápido, *make_pulses* está altamente optimizada y realiza solo unos pocos cálculos. A diferencia de las otras funciones, no necesita matemáticas de coma flotante.

La última función, *stepgen.update-freq*, es responsable de hacer el escalado y algunos otros cálculos que deben realizarse solo cuando el comando de frecuencia cambia.

Para nuestro ejemplo, esto significa que queremos ejecutar *siggen.0.update* a una velocidad moderada para calcular los valores de seno y coseno. Inmediatamente después de ejecutar *siggen.0.update*, queremos ejecutar *stepgen.update_freq* para cargar los

nuevos valores en el generador de impulsos de paso. Finalmente necesitamos ejecutar *stepgen.make_pulses* lo más rápido posible para pulsos suaves. Puesto que no usamos retroalimentacion de posición, no necesitamos ejecutar *stepgen.capture_position* en absoluto.

Ejecutamos las funciones agregándolas a hilos. Cada hilo se ejecuta a una tasa específica. Veamos qué hilos tenemos disponibles.

```
halcmd: show thread

Realtime Threads:
     Period  FP     Name               (     Time, Max-Time )
     996980  YES                  slow (        0,        0 )
      49849  NO                   fast (        0,        0 )
```

Los dos hilos se crearon cuando cargamos *threads*. El primero, *slow*, se ejecuta cada milisegundo y es capaz de ejecutar funciones en modo de punto flotante. Lo usaremos para *siggen.0.update* y *stepgen.update_freq*. El segundo hilo es *fast*, que se ejecuta cada 50 microsegundos, y no es compatible con punto flotante. Lo usaremos para *stepgen.make_pulses*. Para conectar las funciones al hilo apropiado, usamos el comando *addf*. Especificamos la función primero y el hilo despues.

```
halcmd: addf siggen.0.update slow
halcmd: addf stepgen.update-freq slow
halcmd: addf stepgen.make-pulses fast
```

Después de dar estos comandos, podemos ejecutar el comando *show thread* de nuevo para ver que ha pasado.

```
halcmd: show thread

Realtime Threads:
     Period  FP     Name               (     Time, Max-Time )
     996980  YES                  slow (        0,        0 )
                  1 siggen.0.update
                  2 stepgen.update-freq
      49849  NO                   fast (        0,        0 )
                  1 stepgen.make-pulses
```

Ahora cada hilo es seguido por los nombres de las funciones en el orden en que se ejecutarán.

14.4.4.4. Parámetros de configuración

Estamos casi listos para arrancar nuestro sistema HAL. Sin embargo, todavía tenemos que ajustar algunos parámetros. Por defecto, el componente siggen genera señales que oscilan desde +1 a -1. Para nuestro ejemplo, eso está bien, queremos que la velocidad de la mesa varíe de +1 a -1 pulgada por segundo. Sin embargo, la escala del generador de impulsos de pasos no esta del todo correcta. Por defecto, genera una frecuencia de salida de 1 paso por segundo para una entrada de 1.000. Es poco probable que un paso por segundo nos dé una pulgada por segundo de movimiento de la mesa. Supongamos que tenemos un husillo con 5 vueltas por pulgada de avance, conectado a un stepper de 200 pasos por revolucion, con microstepping 10x. Por tanto, se necesitan 2000 pasos para una revolución del tornillo, y 5 revoluciones para moverse una pulgada. Eso significa que la escala general es 10000 pasos por pulgada. Necesitamos multiplicar la entrada de velocidad al generador de impulsos de paso por 10000 para obtener la salida adecuada. Es exactamente para lo qué sirve el parámetro *stepgen.n.velocity-scale*. En este caso, tanto el eje X como el eje Y tienen la misma escala, por lo que establecemos los parámetros de escala para ambos en 10000. Tambien debemos "encender" los generadores. Esa es la mision de stepgen.X.enable.

```
halcmd: setp stepgen.0.position-scale 10000
halcmd: setp stepgen.1.position-scale 10000
halcmd: setp stepgen.0.enable 1
halcmd: setp stepgen.1.enable 1
```

Esta escala de velocidad significa que cuando el pin *stepgen.0.velocity-cmd* sea 1.000, el generador de pasos generará 10000 pulsos por segundo (10KHz). Con el motor y el tornillo de avance descritos anteriormente, eso dará como resultado que el eje se movera a exactamente 1,000 pulgada por segundo. Esto ilustra un concepto HAL clave: cosas como el escalado se hacen al menor nivel posible , en este caso en el generador de impulsos de pasos. La señal interna *X-vel* es la velocidad de la mesa en pulgadas por segundo, y otros componentes, como *siggen*, no saben (ni se preocupan) acerca de la escala. Si cambiamos el husillo, o el motor, cambiaríamos solo el parámetro de escala del generador de impulsos de pasos.

14.4.4.5. ¡Ejecutando!

Ahora tenemos todo configurado y estamos listos para ponerlo en marcha. Como en el primer ejemplo, usamos el comando *start*.

```
halcmd: start
```

Aunque parece no suceder nada, dentro de la computadora los generadores de pulsos de paso están generando impulsos, variando entre +10 KHz y -10 KHz cada segundo. Más adelante en este tutorial vamos a ver cómo sacar esas señales internas para mover motores en la realidad, pero primero queremos ver los pulsos y comprobar lo qué está sucediendo.

14.4.5. Halscope

El ejemplo anterior genera algunas señales muy interesantes. Pero mucho de lo que sucede es demasiado rápido para verlo con *halmeter*. Para ver lo que está sucediendo dentro de HAL, necesitamos un osciloscopio. Afortunadamente HAL tiene uno, llamado *halscope*.

Halscope tiene dos partes: una parte en tiempo real, que se carga como modulo kernel, y una parte de usuario que proporciona la GUI y la pantalla. Sin embargo, usted no necesita preocuparse por esto, porque la porción de espacio de usuario solicita automáticamente que se cargue la parte en tiempo real. La primera vez que se ejecuta halscope en un directorio da como resultado un mensaje de que el archivo *autosave.halscope* no se pudo abrir. Ello se debe a que aun no existe; puede ignorarlo esta primera vez.

Arranque de Halscope

```
halcmd: loadusr halscope
halcmd: halscope: config file 'autosave.halscope' could not be opened
```

Se abrirá la ventana GUI, seguida inmediatamente por el cuadro de diálogo *Realtime function not linked* que se aparece a la siguiente figura.

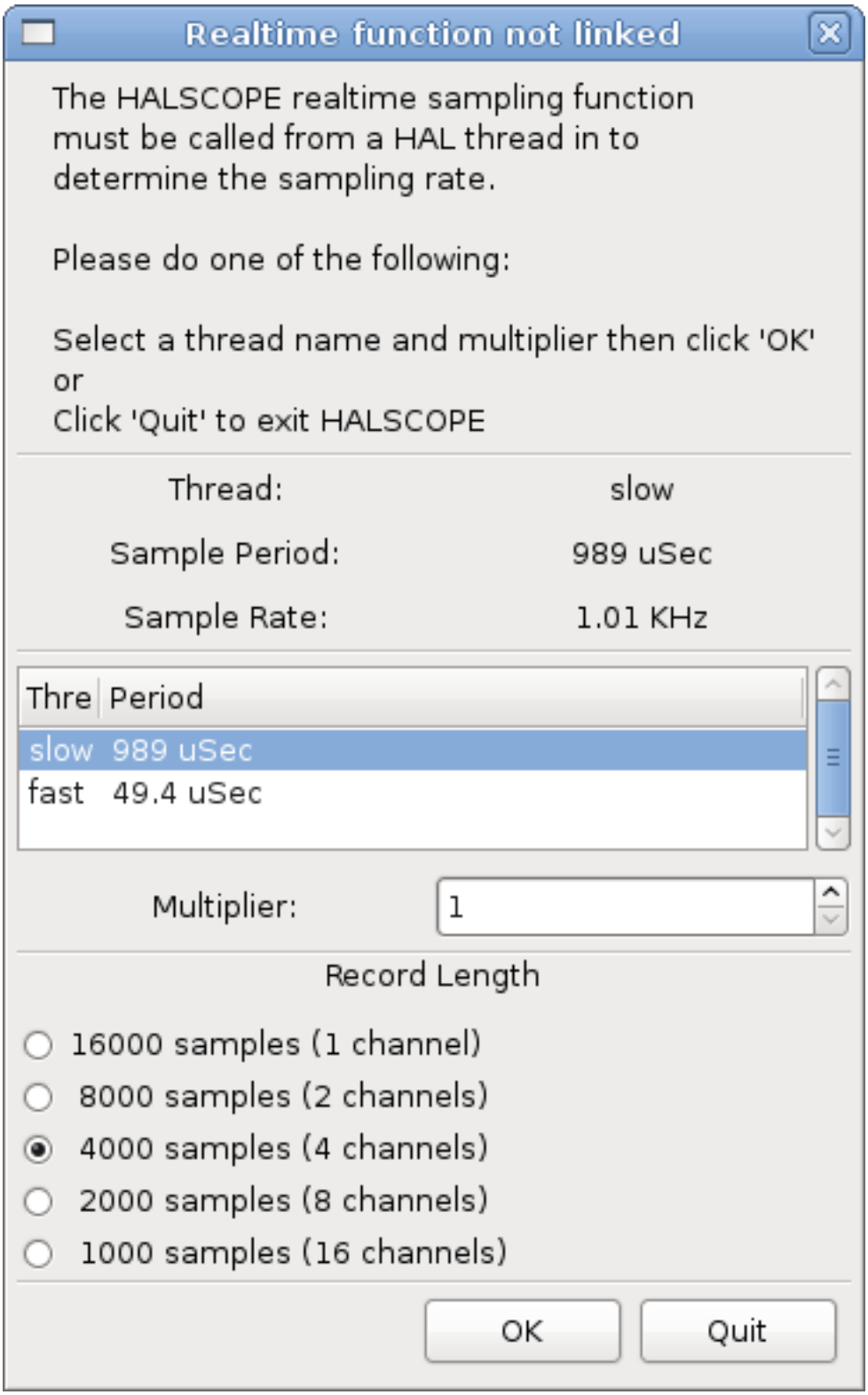


Figura 14.6: diálogo Función Realtime no vinculada

Este diálogo es donde se establece la frecuencia de muestreo para el osciloscopio. Por ahora, queremos muestrear una vez por milisegundo, así que haga clic en el hilo *slow* y deje el multiplicador en 1. También dejaremos la longitud de registro a 4000 muestras, para que podamos usar hasta cuatro canales a la vez. Cuando selecciona un hilo y luego hace clic en *Ok*, el diálogo

desaparece, y la ventana del osciloscopio aparece como en la siguiente figura.

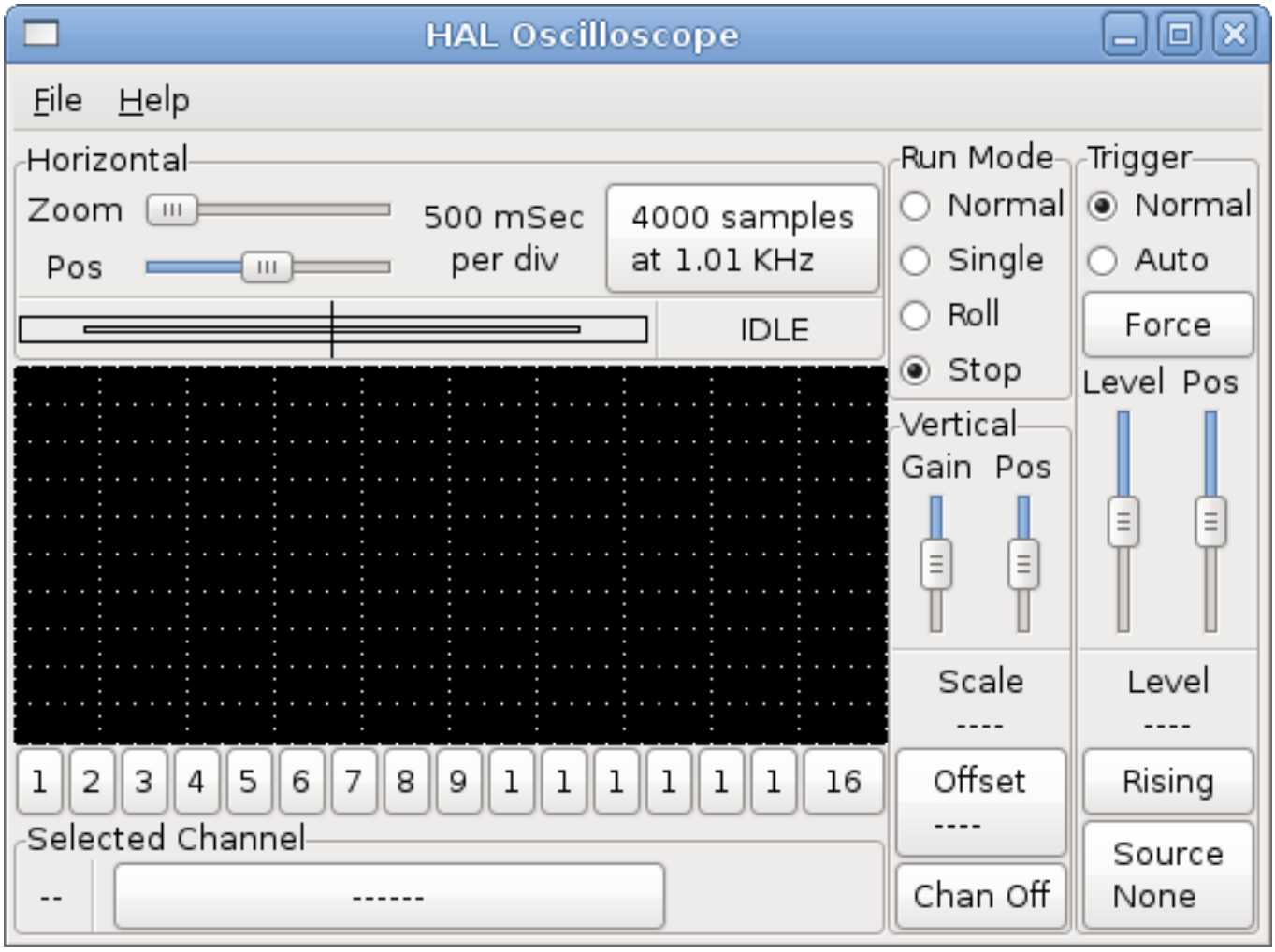


Figura 14.7: Vista inicial de halscope

14.4.5.1. Conectando las sondas

En este punto, Halscope está listo para usarse. Ya hemos seleccionado una frecuencia de muestreo y longitud de registro, por lo que el siguiente paso es decidir qué observar. Esto es equivalente a conectar *sondas virtuales* a HAL. Halscope tiene 16 canales, pero el número que puede usar en cualquier momento depende de la duración del registro; más canales significan menos registros, ya que la memoria disponible para el registro es fija, de aproximadamente 16,000 muestras.

Los botones de canal estan en la parte inferior de la pantalla de halscope. Haga clic en el botón *1* y verá el diálogo *Seleccionar fuente de canal*, como se muestra en la siguiente figura. Este diálogo es muy similar al utilizado por halmeter. Nos gustaría ver las señales que definimos antes. Hacemos clic en la pestaña "Señales" y el cuadro de diálogo muestra todas las señales en HAL (solo dos para este ejemplo).

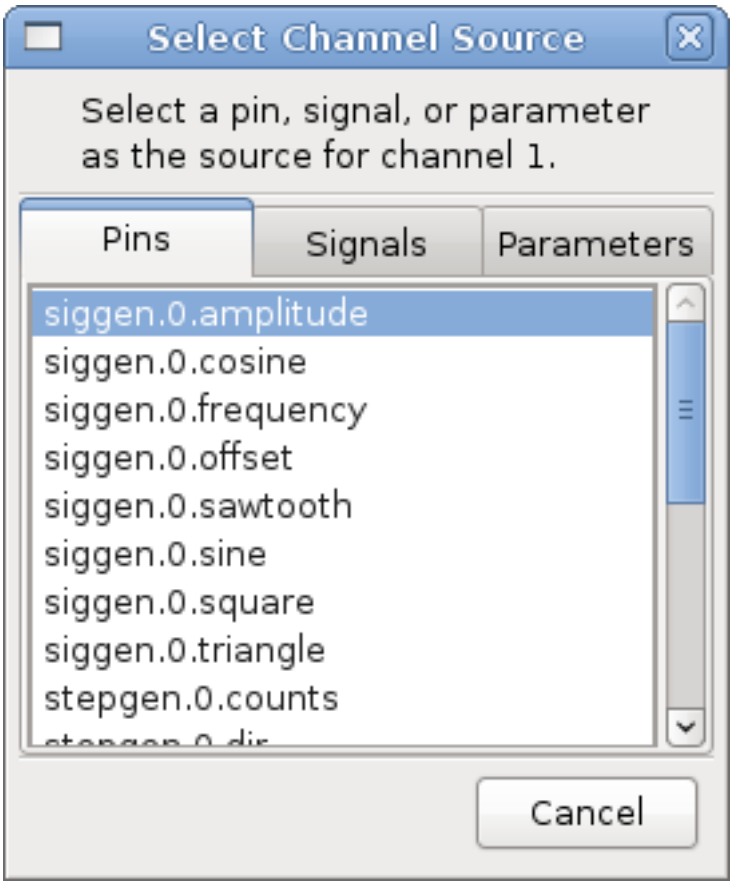


Figura 14.8: Seleccione Origen del canal

Para elegir una señal, simplemente haga clic en ella. En este caso, queremos que el canal 1 muestre la señal *X-vel*. Haga clic en la pestaña Señales y luego haga clic en *X-vel* y el diálogo se cerrara. El canal está ahora seleccionado.

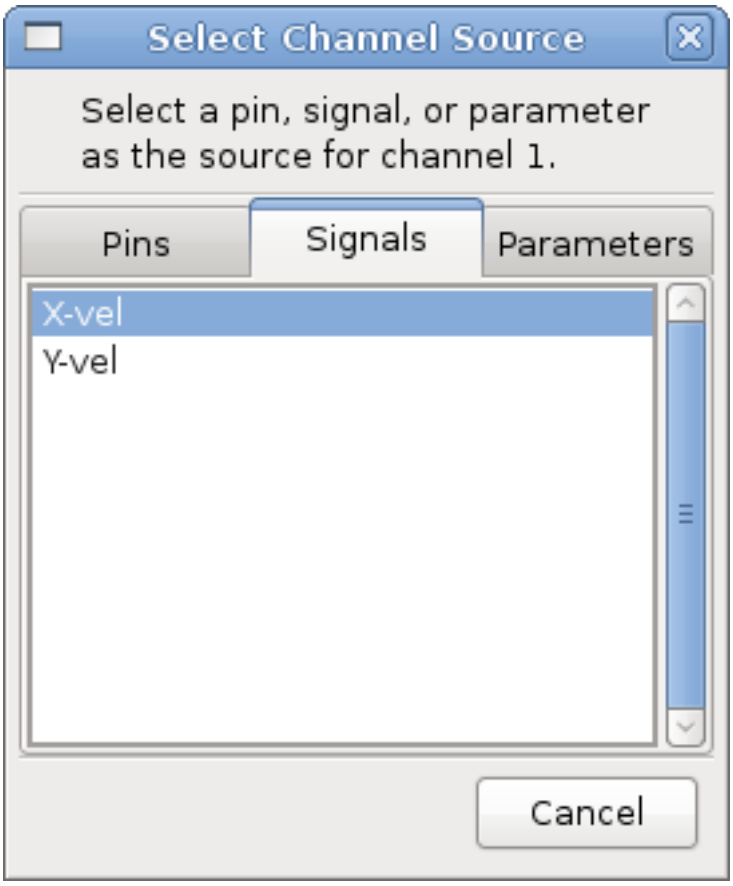


Figura 14.9: Seleccion de señal

El botón del canal 1 se ve "presionado", y el canal número 1 y el nombre *X-vel* aparece debajo de la fila de botones. Esa pantalla siempre indica el canal seleccionado; puede tener muchos canales en la pantalla, pero el seleccionado se resalta, y los diversos controles, como posicion vertical y escala, siempre funcionan en el canal seleccionado.

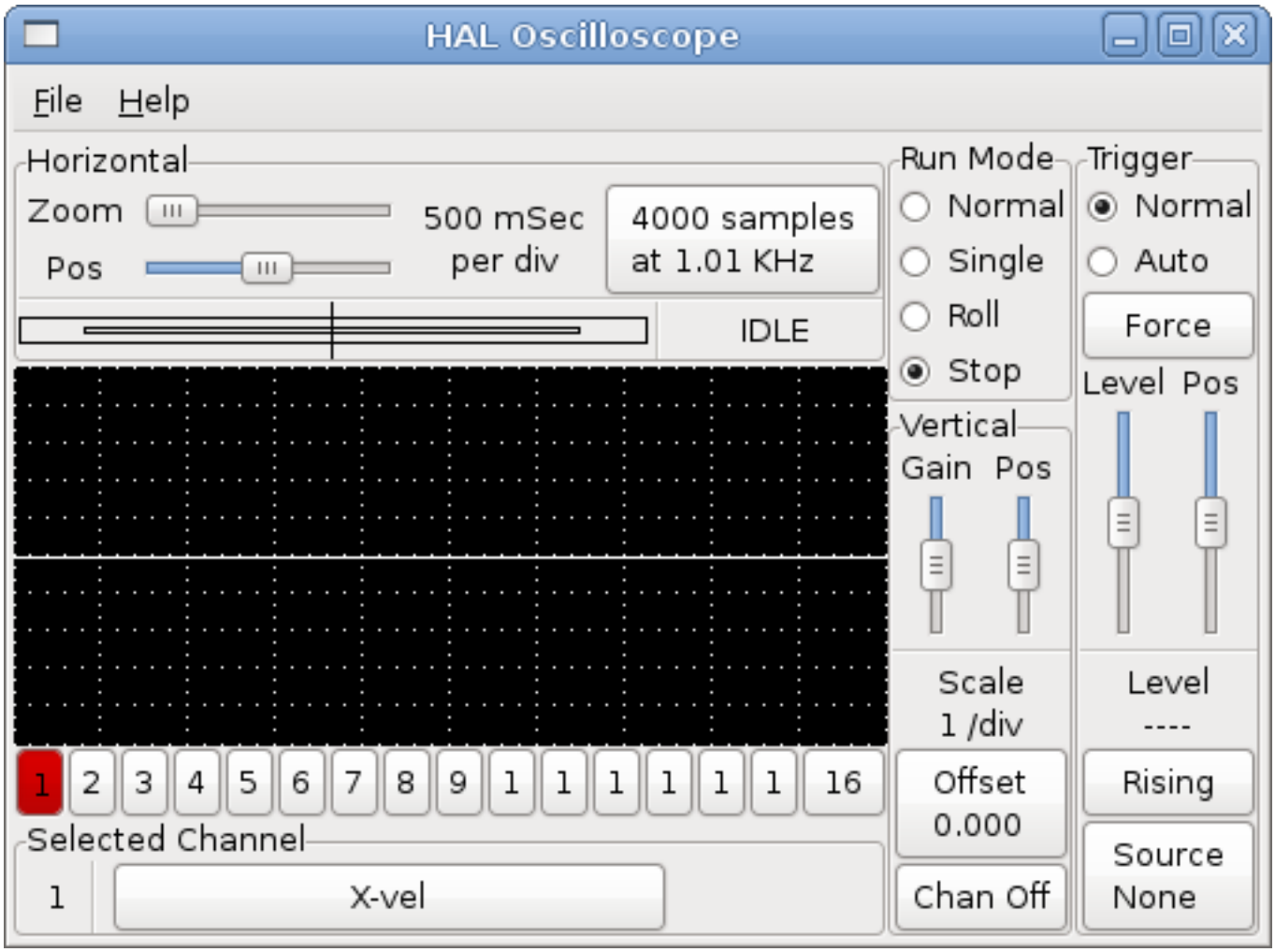


Figura 14.10: Halscope

Para agregar una señal al canal 2, haga clic en el botón *2*. Cuando el diálogo aparezca, haga clic en la pestaña *Señales*, luego haga clic en *Y-vel*. También queremos mirar las salidas de onda cuadradas y triangulares. No hay señales conectadas a esos pines, por lo que usamos la pestaña *Pins* en su lugar. Para el canal 3, seleccione *siggen.0.triangle* y para el canal 4, seleccione *siggen.0.square*.

14.4.5.2. Capturando nuestras primeras formas de onda

Ahora que tenemos varias sondas enganchadas a HAL, es hora de capturar algunas formas de onda. Para iniciar el osciloscopio, haga clic en el botón *Normal* en la sección *Run Mode* de la pantalla (arriba a la derecha). Como tenemos 4000 muestras de longitud de registro, y están adquiriendo 1000 muestras por segundo, Halscope tardara unos 2 segundos en llenar la mitad de su buffer. Durante ese tiempo, se mostrará una barra de progreso, justo encima de la pantalla principal, conforme se rellena el buffer. Una vez que el buffer está medio lleno, el osciloscopio espera un disparo. Como aún no hemos configurado uno, esperará indefinidamente. Para accionar el disparo manualmente, haga clic en el botón *Force* en la sección *Trigger* en la parte superior derecha. Debería ver el resto del relleno del buffer, y luego la pantalla mostrará las formas de onda capturadas. El resultado se verá como en la siguiente figura.

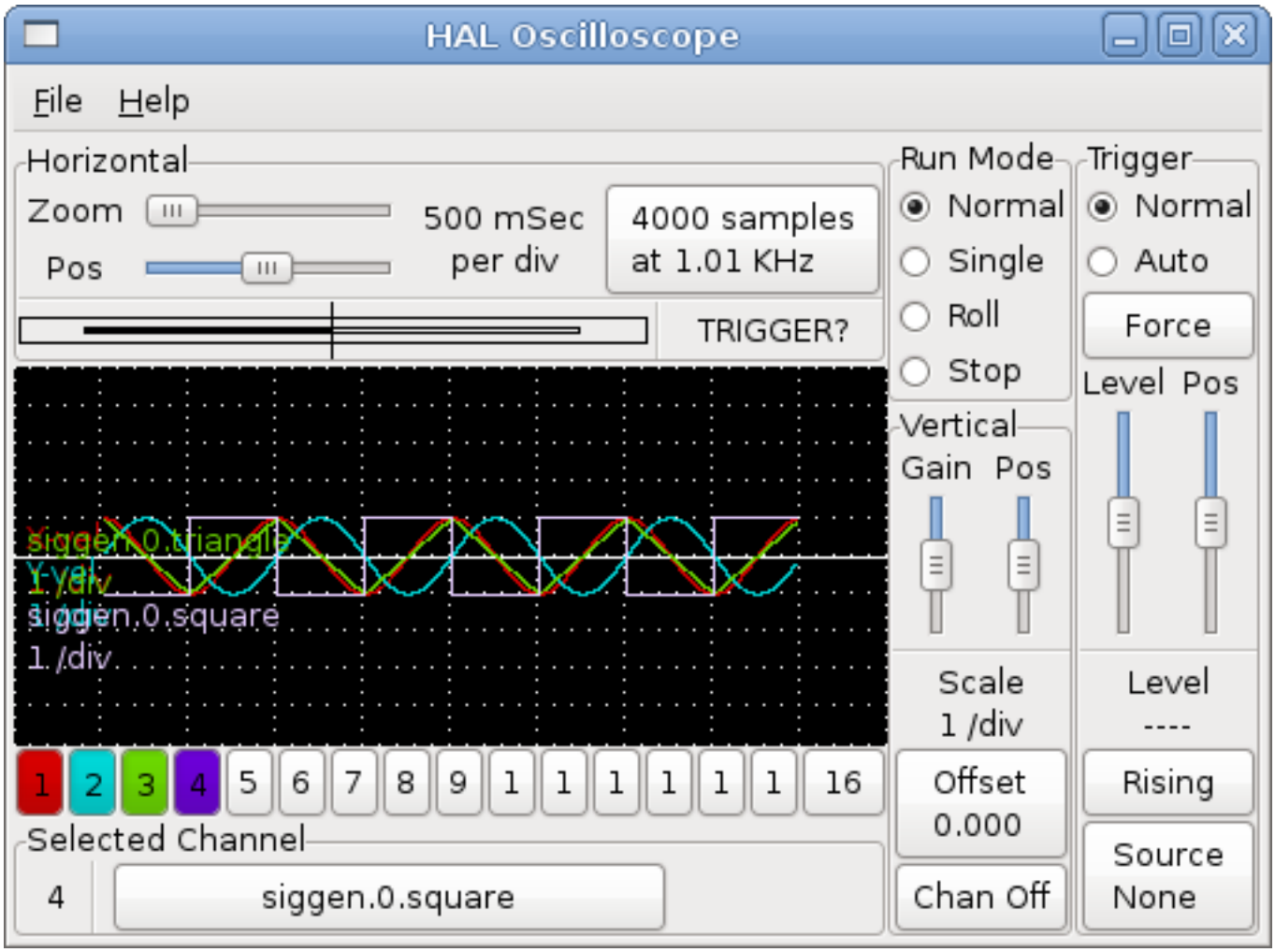


Figura 14.11: Capturando formas de onda

El cuadro "Selected Channel", en la parte inferior, le dice que la señal de color púrpura es el canal actualmente seleccionado, canal 4, que muestra el valor del pin *siggen.0.square*. Intente hacer clic en los botones de canal 1 hasta 3 para resaltar las otras tres señales.

14.4.5.3. Ajustes verticales

Las ondas son bastante difíciles de distinguir ya que las cuatro están una sobre otra. Para solucionar esto, usamos los controles *Vertical* en el cuadro derecho de la pantalla. Estos controles actúan en el canal actualmente seleccionado. Al ajustar la ganancia, observe que cubre un amplio rango. A diferencia de un osciloscopio real, este puede mostrar señales que van desde muy pequeñas (pico-unidades) a muy grandes (unidades Tera). El control de posición mueve el trazo mostrado arriba y abajo sobre la altura de la pantalla solamente. Para ajustes más grandes, se debe usar el botón *offset*.

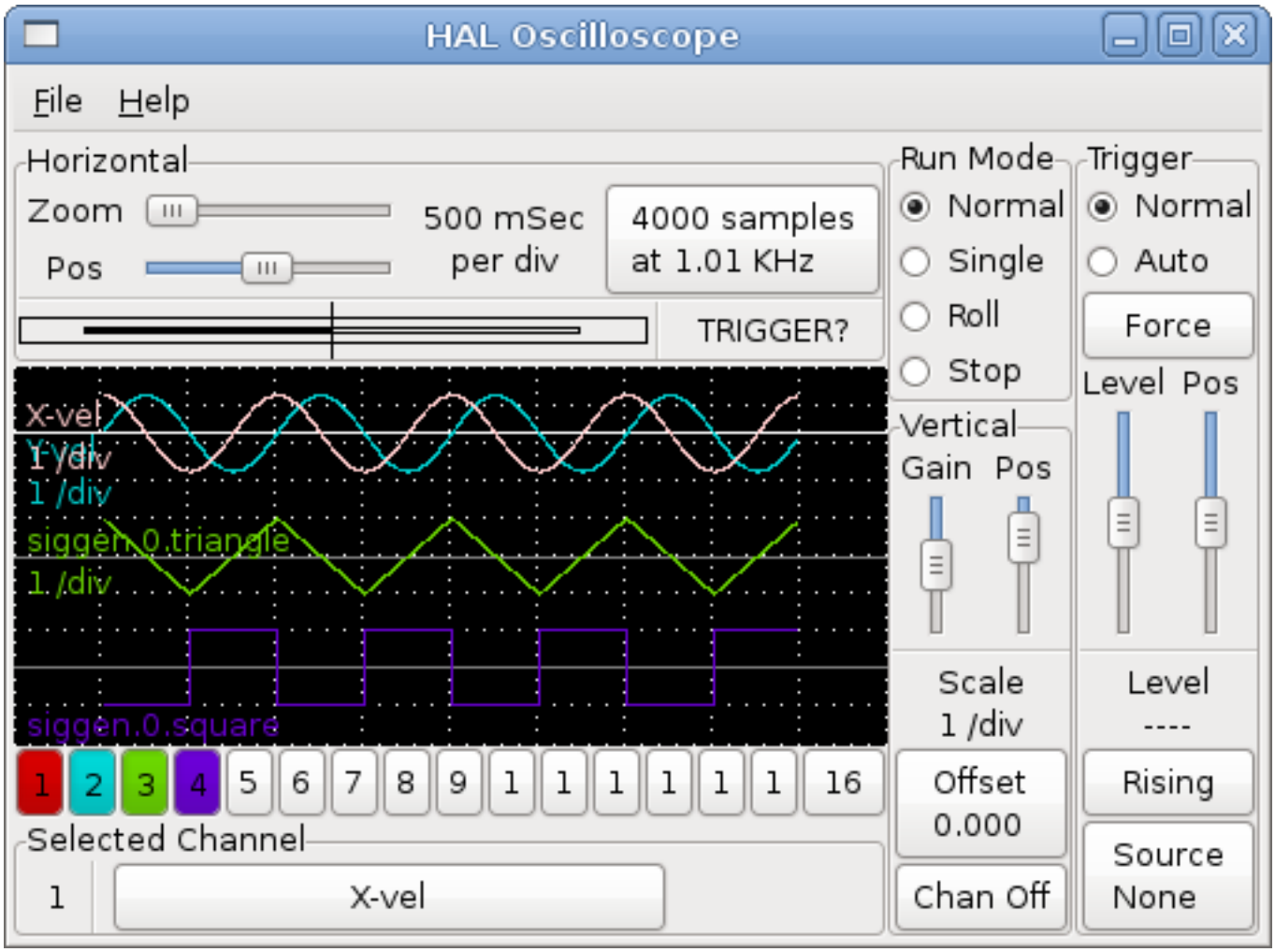


Figura 14.12: Ajuste vertical

14.4.5.4. Disparando

Usar el botón *Forzar* es una forma bastante burda de activar el osciloscopio. Para configurar un disparo real, haga clic en el botón *Source*, abajo a la derecha. Se abrirá el cuadro de diálogo *Trigger Source*, que es simplemente una lista de todas las sondas que están actualmente conectadas. Seleccione una sonda para usarla como disparador haciendo clic en ella. Para este ejemplo lo haremos con el canal 3, la onda triangular, como se muestra en la siguiente figura.

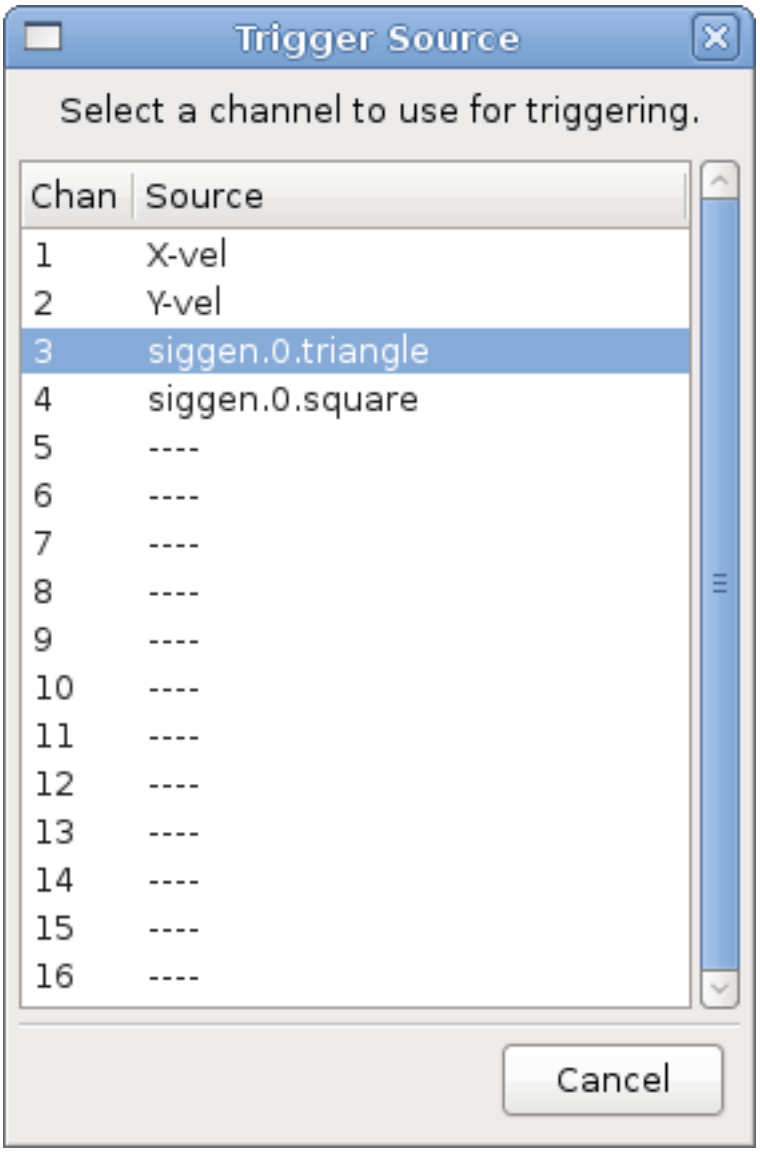


Figura 14.13: Diálogo de Fuente de Disparo

Después de configurar la fuente de activación, puede ajustar el nivel de activación y posición del gatillo usando los controles deslizantes en la casilla *Trigger* a lo largo del borde derecho. El nivel se puede ajustar desde la parte superior a la inferior de la pantalla, y se muestra debajo de los controles deslizantes. La posicion es la ubicación del punto de disparo dentro del registro general. Con el deslizador completamente hacia abajo, el punto de disparo está al final del registro, y halscope muestra lo que sucedió antes de ese punto. Cuando el control deslizante está arriba, el punto de disparo está al comienzo de la captura de muestras, mostrando lo que sucedió después de que se activó. El punto del gatillo es visible como una línea vertical en el cuadro de progreso sobre la pantalla. La polaridad del gatillo se puede cambiar haciendo clic en el botón justo debajo del nivel de disparo.

Ahora que hemos ajustado los controles verticales y el disparo, la pantalla del osciloscopio se parecerá a la siguiente figura.

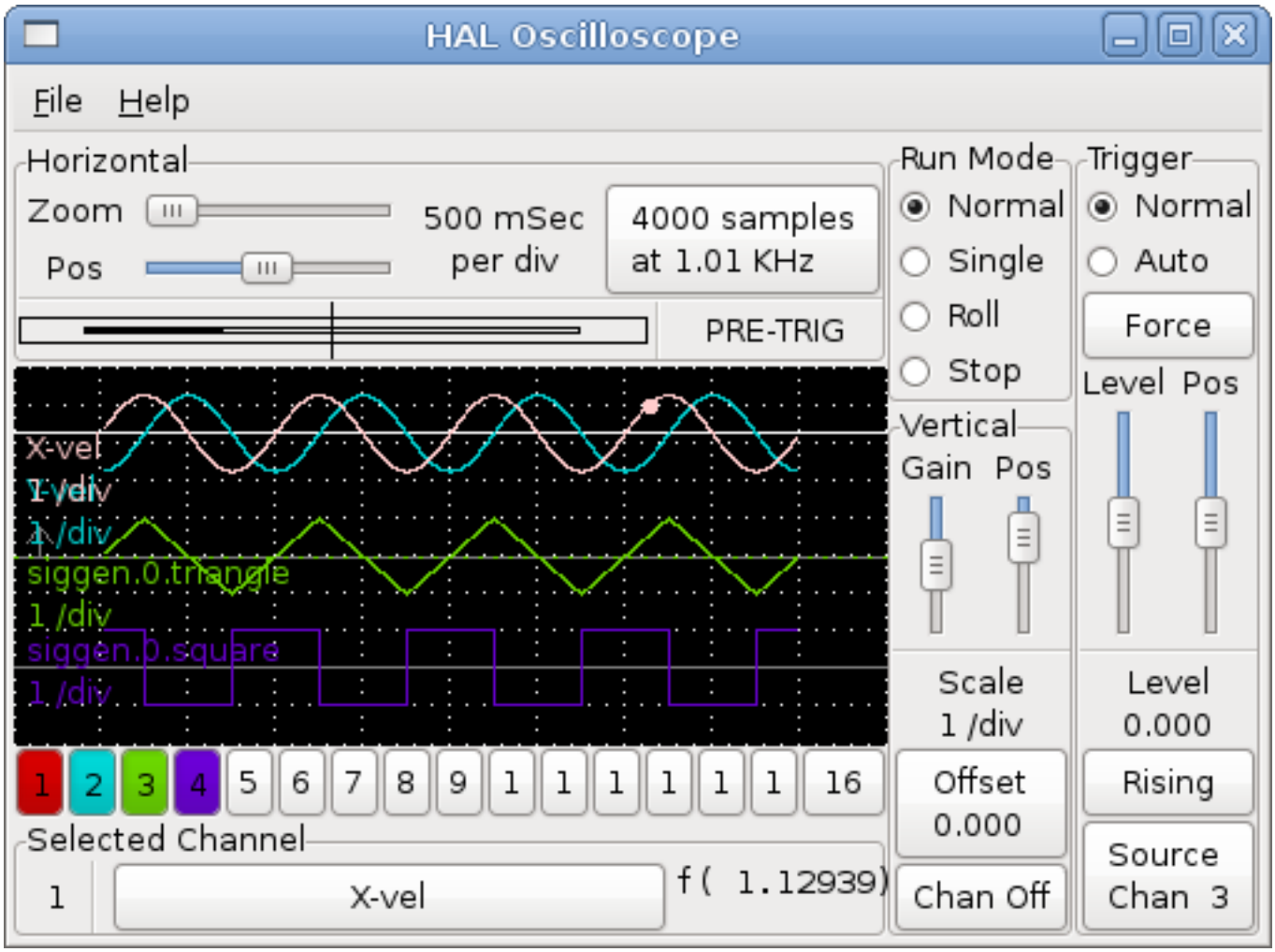


Figura 14.14: Formas de onda con Triggering

14.4.5.5. Ajustes horizontales

Para observar detenidamente parte de una forma de onda, puede usar el deslizador de zoom en la parte superior de la pantalla, para expandir las formas de onda horizontalmente, y el control deslizante de posición para determinar qué parte de la forma de onda ampliada sera visible. Sin embargo, a veces simplemente expandir las formas de onda no es suficiente y necesita aumentar la tasa de muestreo. Por ejemplo, nos gustaría para ver los pulsos de pasos reales que se generan en nuestro ejemplo. Dado que los pulsos de paso pueden tener solo 50 us. de largo, muestrear a 1 KHz no es lo suficientemente rápido para capturarlos. Para cambiar la frecuencia de muestreo, haga clic en el botón que indica el número de muestras y la frecuencia de muestreo, que mostrara el dialogo *Seleccionar Frecuencia de muestreo*. Para este ejemplo, haremos clic en el hilo *fast* de 50 us, que nos da una velocidad de muestra de aproximadamente 20KHz. Ahora en lugar de mostrar aproximadamente 4 segundos de datos, un registro es 4000 muestras a 20 kHz, o aproximadamente 0,20 segundos.

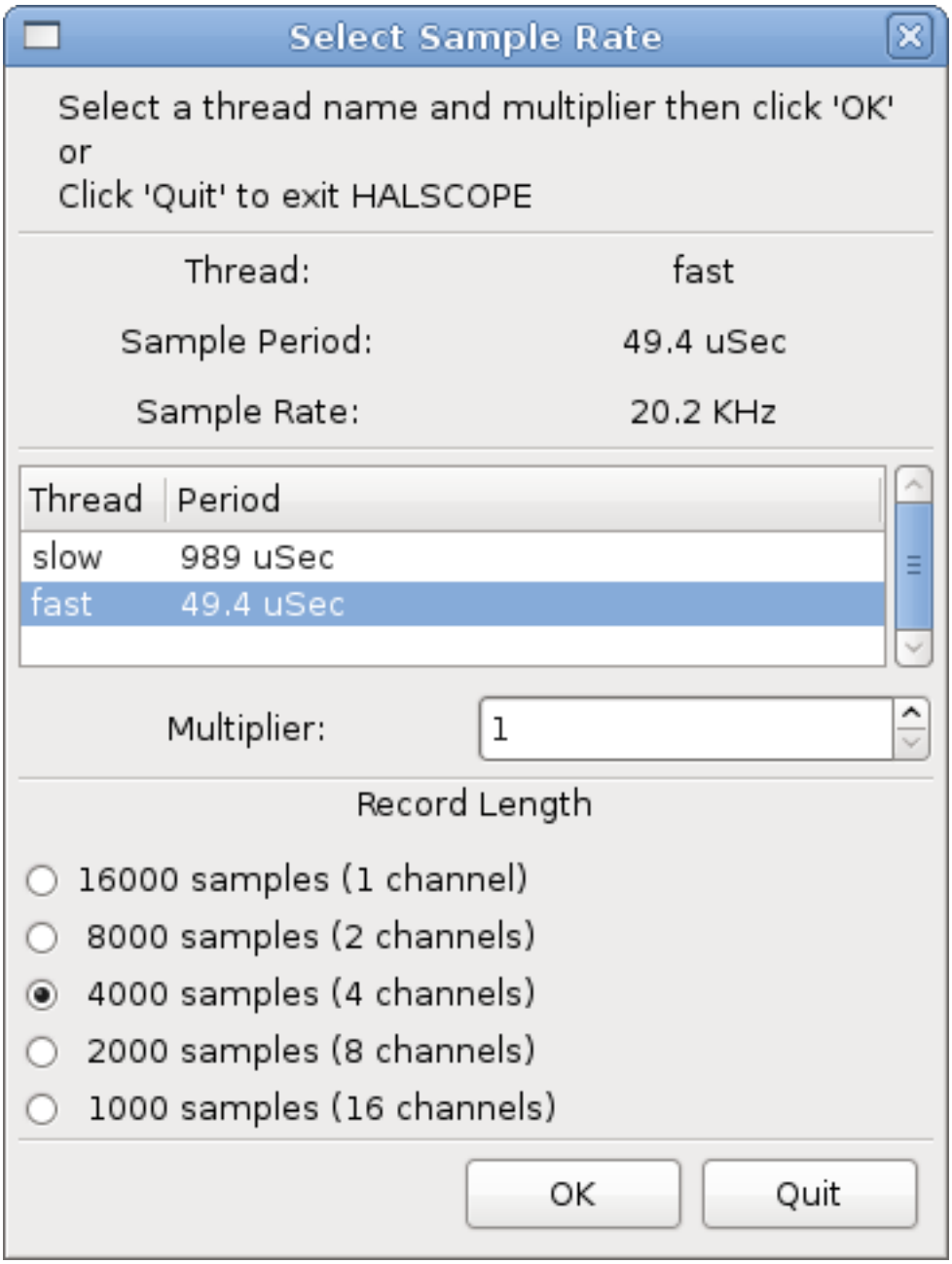


Figura 14.15: Diálogo de frecuencia de muestreo

14.4.5.6. Más canales

Ahora veamos los pulsos de paso. Halscope tiene 16 canales, pero para este ejemplo estamos usando solo 4 a la vez. Antes de seleccionar más canales, tenemos que apagar un par. Haga clic en el botón del canal 2, luego haz clic en el botón "Chan Off" en la parte inferior del cuadro "Vertical". A continuación, haga clic en el canal 3, apaguelo, y haga lo mismo para el canal 4. Aunque los canales están apagados, todavía recuerdan a que están conectados y, de hecho, vamos a seguir utilizando el canal 3 como fuente de disparo. Para agregar nuevos canales, seleccione el canal 5 y elija el pin *stepgen.0.dir*, luego el canal 6, y seleccione *stepgen.0.step*. Entonces haga clic en el modo de ejecución *Normal* para iniciar el osciloscopio y ajuste el zoom horizontal a 5 ms por división. Debería ver que los pulsos de paso disminuyen a medida que el comando de velocidad (canal 1) se acerca a cero, luego el pin de dirección cambia de estado y los pulsos de paso se aceleran nuevamente. Usted podría querer aumentar la ganancia en el canal 1 a alrededor de 20 ms por división para ver mejor el cambio en el comando de velocidad. El resultado debería verse como la siguiente figura.

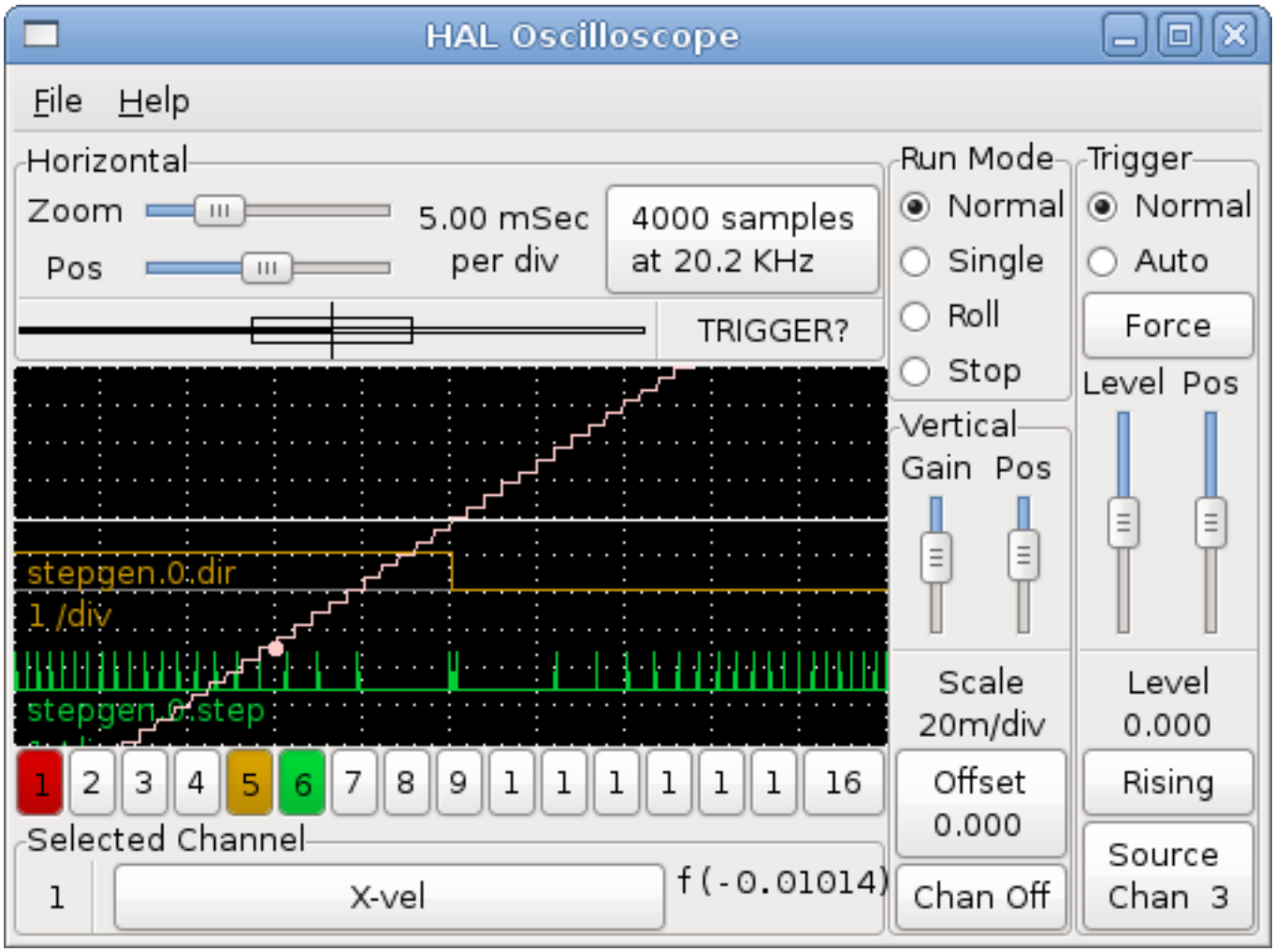


Figura 14.16: Pasos de paso

14.4.5.7. Más muestras

Si desea grabar más muestras a la vez, reinicie el tiempo real y cargue halscope con un argumento numérico que indique el número de muestras que quiere capturar

```
halcmd: loadusr halscope 80000
```

Si el componente *scope_rt* no estaba cargado, halscope lo carga y solicita 80000 muestras en total, de modo que al tomar muestras en 4 canales a la vez habrá 20000 muestras por canal. (Si *scope_rt* ya estaba cargado, el argumento numérico para halscope no tendrá ningún efecto).

=Referencia general

==Convenciones Generales de Nomenclatura

Unas convenciones de nombres consistentes harán que HAL sea mucho más fácil de usar. Por ejemplo, si cada controlador de encoder proporciona el mismo conjunto de pines y se los nombró de la misma manera, sería fácil cambiar de un tipo de controlador de encoder a otro. Desafortunadamente, al igual que muchos proyectos de código abierto, HAL es una combinación de cosas que fueron diseñadas, y cosas que simplemente evolucionaron. Como resultado, hay muchas inconsistencias. Esta sección intenta abordar ese problema definiendo algunas convenciones, pero probablemente pasará un tiempo antes de que todos los módulos se conviertan para seguirlas.

Halcmd y otras utilidades HAL de bajo nivel tratan los nombres HAL como entidades simples, sin estructura interna. Sin embargo, la mayoría de los módulos sí tienen alguna estructura implícita. Por ejemplo, una tarjeta proporciona varios bloques

funcionales, cada bloque puede tener varios canales, y cada canal tiene uno o más pines. Resulta asi en una estructura que se asemeja a un árbol de directorios. Aunque halcmd no reconoce estructuras de árbol, la elección adecuada de las convenciones de nomenclatura dejará agrupar elementos relacionados (ya que ordena los nombres). Además, las herramientas de más alto nivel se pueden diseñarse para reconocer dicha estructura, si los nombres proporcionan la información necesaria. Para hacer eso, todos los componentes HAL deberían seguir estas reglas:

- Los puntos (".") separan niveles de la jerarquia. Esto es análogo a la barra inclinada ("/") en un nombre de archivo.
- Los guiones ("-") separan palabras o campos en el mismo nivel de la jerarquia.
- Los componentes HAL no deben usar guiones bajos o "mezcla de casos".
- Use solo letras minúsculas y números en los nombres.

==Convenciones de nombres de controladores de hardware

===Nombres de pines/parámetros

Los controladores de hardware deben usar cinco campos (en tres niveles) para formar un pin o nombre del parámetro, de la siguiente manera:

`<nombre-dispositivo>.<dispositivo-num>.<tipo-io>.<chan-num>.<nombre-específico>`

Los campos individuales son:

<nombre-dispositivo>
El dispositivo con el que el controlador está diseñado trabajar. Esto es a menudo una placa de interfaz de algún tipo, pero hay otras posibilidades.

<dispositivo-num>
Es posible instalar más de una placa servo, puerto paralelo, u otro dispositivo de hardware en una computadora. El número de dispositivo identifica un dispositivo específico. Los números de dispositivo comienzan en 0 y se incrementan.

<tipo-io>
La mayoría de los dispositivos proporcionan más de un tipo de E/S. Incluso el simple puerto paralelo tiene entradas digitales y salidas digitales. Tarjetas mas complejas pueden tener entradas y salidas digitales, contadores de encoder, generadores pwm o de impulsos de pasos, convertidores analógico a digital y/o digital a analógico u otras capacidades únicas. El tipo de E/S se usa para identificar el tipo de E/S al que está asociado un pin o parámetro. Idealmente, los controladores que implementan el mismo tipo de E/S, incluso si son para dispositivos muy diferentes, deberían proporcionar un conjunto consistente de pines y parámetros, y un comportamiento idéntico. Por ejemplo, todas las entradas digitales debería comportarse de la misma manera desde el interior del HAL, independientemente del dispositivo.

<chan-num>
Prácticamente todos los dispositivos de E/S tienen múltiples canales, y el número de canal identifica a cada uno de ellos. Como los números de dispositivo, los números de canal comienzan en cero y se incrementan.[3] Si hay más de un dispositivo instalado, los números de canal en dispositivos adicionales comienzan desde cero. Si es posible tener un número de canal mayor que 9, los números de canal deben ser dos dígitos, con un cero a la izquierda en números menores a 10 para preservar la ordenacion. Algunos módulos tienen pines y/o parámetros que afectan a más de un canal. Por ejemplo, un generador de PWM podría tener cuatro canales con cuatro entradas independientes de "ciclo de trabajo", pero un solo parámetro "frecuencia" que controla los cuatro canales (debido a limitaciones hardware). El parámetro de frecuencia debe usar "0-3" como número de canal.

<nombre-específico>
Un canal de E/S individual puede tener solo un pin HAL asociado con el, pero la mayoría tiene más de uno. Por ejemplo, una entrada digital tiene dos pines, uno es el estado del pin físico, el otro es la misma cosa pero invertida. Eso le permite al configurador elegir entre entradas activas bajas y activas altas. Para la mayoría de los tipos-io, hay un conjunto estándar de pines y parámetros, (denominada "interfaz canónica") que el driver debería implementar. Las interfaces canónicas se describen en el capítulo Interfaces canonicos de dispositivos

[3] Una excepción a la regla "los números de los canales comienzan en cero" es el puerto paralelo. Sus pines HAL están numerados con el número de pin correspondiente en el conector DB-25. Esto es conveniente para el cableado, pero inconsistente con otros controladores. Existe cierto debate sobre si esto es un error o una característica.

EJEMPLOS

motenc.0.encoder.2.position

- Salida de posición del tercer canal (2) de encoder en la primera (0) tarjeta Motenc.

stg.0.din.03.in

- Estado de la cuarta entrada digital (03) en la primera (0) tarjeta Servo-to-Go.

ppmc.0.pwm.00-03.frequency

- Frecuencia de portadora utilizada para los canales PWM 0 a 3 (cuatro canales) en la primera (0) placa ppmc Pico Systems.

===Nombres de funciones

Los controladores de hardware generalmente solo tienen dos tipos de funciones HAL, unas que lee el hardware y actualiza los pines HAL, y otras que escriben en el hardware utilizando datos de pines HAL. Deben nombrarse de la siguiente manera:

`<device-name>-<device-num>.<io-type>-<chan-num-range>.read|write`

<device-name>
El mismo que se use para pines y parámetros.

<device-num>
El dispositivo específico al que accederá la función.

<io-type>
Opcional. Una función puede acceder a todas las E/S de una placa, o puede acceder solo a cierto tipo. Por ejemplo, puede haber distintas funciones para leer contadores de encoder y leer E/S digitales. Si tales funciones independientes existen, el campo <io-type> identifica el tipo de E/S a la que acceden. Si una sola función lee todas las E/S provistas por la tarjeta, <io-type> no se utiliza. [4]

<chan-num-range>
Opcional. Se usa solo si la E/S <io-type> se divide en grupos y se accede por diferentes funciones.

read|write
Indica si la función lee el hardware o escribe en él.

EJEMPLOS

motenc.0.encoder.read

- lee todos los codificadores en la primera placa motenc.

generic8255.0.din.09-15.read

- lee el segundo puerto de 8 bits en el primera placa genérica basada en 8255 de E/S digital.

ppmc.0.write

- escribe todas las salidas (generadores de pasos, pwm, DAC y digitales) en la primera placa Pico Systems ppmc.

14.5. Componentes principales

Vea también la página man *motion(9)*.

[4] Nota para los programadores de controladores: NO implemente por separado funciones para diferentes tipos de E/S a menos que sean interrupibles y puedan trabajar en hilos independientes. Si se interrumpe una lectura de encoder, se leen entradas digitales, y luego se reanudar la lectura, causará problemas. Implemente una función única que haga todo.

14.5.1. Motion (motmod)

Los siguientes pines y parámetros son creados por el módulo *motmod* en tiempo real. Este módulo proporciona una interfaz HAL para el planificador de movimiento de LinuxCNC. Básicamente, *motmod* toma una lista de puntos de referencia y genera un flujo correcto, limitado por las restricciones de posiciones de articulaciones, de posiciones de articulaciones que alimenta a los drivers de motor.

Opcionalmente, el número de E/S digitales se establece con *num_dio*. El número de E/S analógicas se establece con *num_aio*. El valor predeterminado es 4 en cada uno.

Los nombres de pines y parámetros, que comienzan con *axis.L* y *joint.N* son leídos y actualizados por la función motion-controller.

Motion se carga con el comando *motmod*. Antes de motion, debe ser cargado un modulo de cinematica.

```
loadrt motmod [base_period_nsec=period] [servo_period_nsec=period]
              [traj_period_nsec=period] [num_joints=[0-9] ([num_dio=1-64] [num_aio=1-16])
              ([unlock_joints_mask=0xnn])
```

- *base_period_nsec=period* - es el período de la tarea *Base* en nanosegundos. Este es el hilo más rápido en la máquina.

  nota
  En sistemas basados en servo, generalmente no hay razón para que *base_period_nsec* sea mas rapida que *servo_period_nsec*. En máquinas con generación por pasos de software, *base_period_nsec* determina la cantidad máxima de pasos por segundo. En ausencia de necesidad de pasos largos y requisitos de espacio, la tasa de pasos máxima absoluta es de un paso por *base_period_nsec*. Por lo tanto, para *base_period_nsec = 50000* (0,05 ms) se tendria una tasa de paso máxima absoluta de (1000/0,05) 20,000 pasos por segundo. 50,000 ns (50 us) es un valor bastante conservador. El menor valor utilizable está relacionado con el resultado de la prueba de latencia, el ttiemp de paso necesario, y la velocidad del procesador. Elegir un *base_period_nsec* que sea demasiado rapido puede conducir a un mensaje de "retraso en tiempo real inesperado", bloqueos o reinicios espontáneos.

- *servo_period_nsec = period* - Este es el período de la tarea *Servo* en nanosegundos. Este valor se redondeará a un múltiplo entero de *base_period_nsec*. Este período se usa incluso en sistemas basados en motores paso a paso.

  Esta es la velocidad a la que se calculan las nuevas posiciones de los motores, se verifica el error de seguimiento, se actualizan los valores de salida PID, y así sucesivamente. La mayoría de los sistemas no necesitarán cambiar este valor. Es la tasa de actualización del planificador de movimiento de bajo nivel.

- *traj_period_nsec = period* - Este es el periodo de la tarea *Planificador de trayectoria* en nanosegundos. Este valor se redondeará a un número entero múltiplo de *servo_period_nsec*. Excepto para máquinas con cinemática unusual (por ejemplo, hexápodos) no hay razón para que este valor sea mas alto que *servo_period_nsec*.

14.5.1.1. Opciones

Si la cantidad de E/S digital necesaria es mayor que el valor predeterminado (4), puede agregar hasta 64 E/S digitales usando la opción num_dio al cargar *motmod*

Si la cantidad de E/S analógica necesaria es mayor que el valor predeterminado de (4), puede agregar hasta 16 E/S analógicas usando la opción num_aio al cargar *motmod*

El parámetro *unlock_joints_mask* se usa para crear pines para una articulacion utilizada como indexador con bloqueo (típicamente rotativas). Los bits de máscara seleccionan la(s) articulación(es). El bit menos significativo de la máscara selecciona la articulacion 0 y sucesivamente. Ejemplo:

```
unlock_joints_mask = 0x38 (111000b)
```

selecciona articulaciones 3,4,5

14.5.1.2. Pines

Estos pines son creados por el módulo de tiempo real *motmod*:

- *motion.adaptive-feed* - (float, in) Cuando la alimentación adaptativa está habilitada con *M52 P1*, la velocidad ordenada se multiplica por este valor. Este efecto es multiplicativo con el valor de nivel NML de ajuste manual de alimentación y *motion.feed-hold*. A partir de la versión 2.8 de LinuxCNC, es posible utilizar un valor de alimentación adaptativo negativo para ejecutar la ruta del código G en reversa.
- *motion.analog-in-00* - (float, in) Estos pines (00, 01, 02, 03, o más si están configurados) son controlado por M66 (espera en entrada).
- *motion.analog-out-00* - (float, out) Estos pines (00, 01, 02, 03, o más si están configurados) son controlado por M67 o M68 (salida analogica sincronizada/inmediata)
- *motion.coord-error* - (bit, out) TRUE cuando el movimiento ha encontrado un error, como exceder un límite software.
- *motion.coord-mode* - (bit, out) TRUE cuando el movimiento está en *modo coordinado*, en oposición al *modo teleop*
- *motion.current-vel* - (float, out) La velocidad actual de la herramienta en unidades de usuario por segundo.
- *motion.digital-in-00* - (bit, in) Estos pines (00, 01, 02, 03 o más si están configurados) son controlado por *M62 a M65* (control de salida digital).
- *motion.digital-out-00* - (bit, out) Estos pines (00, 01, 02, 03 o más si están configurados) son controlado por *M62 a M65*.
- *motion.distance-to-go* - (float, out) La distancia restante en el movimiento actual.
- *motion.enable* - (bit, in) Si este bit se hace FALSE, el movimiento se detiene; la máquina queda colocada en el estado *máquina desconectada* y se muestra un mensaje para el operador. En movimiento normal, este bit es TRUE.
- *motion.feed-hold* - (bit, in) Cuando Feed Stop Control está habilitado (*M53 P1*), y este bit es TRUE, la velocidad de avance se establece en 0.
- *motion.feed-inhibit* - (bit, in) Cuando este bit es TRUE, la velocidad de avance se establece en 0. Esta accion se retrasará durante los movimientos de sincronización del husillo hasta el final del movimiento.
- *motion.in-position* - (bit, out) TRUE si la máquina está en posición.
- *motion.motion-enabled* - (bit, out) TRUE cuando la maquina está en estado *machine on*.
- *motion.motion-type* - (s32, out) Valores definidos en src/emc/nml_intf/motion_types.h
  - 0: Idle (sin movimiento)
  - 1: Traverse
  - 2: Linear feed
  - 3: Arc feed
  - 4: Tool change
  - 5: Probing
  - 6: Rotary axis indexing
- *motion.on-soft-limit* - (bit, out) TRUE cuando la máquina está en un límite software.
- *motion.probe-input* - (bit, in) *G38.n* (sondeo) usa el valor de este pin para determinar cuándo la sonda ha hecho contacto. TRUE para contacto de sonda cerrado (ha tocado), FALSO para contacto de sonda abierto.
- *motion.program-line* - (s32, out) La línea de programa actual durante la ejecución. Vale cero si no corre un programa o entre líneas mientras se avanza linea a linea.
- *motion.requested-vel* - (float, out) La velocidad solicitada actual en unidades de usuario por segundo. Este valor es la configuración de la palabra F del archivo de código G, posiblemente reducido para acomodar la velocidad y aceleración a los limites de la máquina. El valor en este pin no refleja el ajuste manual de la alimentación o cualquier otro ajuste.

- *motion.teleop-mode* - (bit, out) TRUE cuando el movimiento está en "modo teleop", en oposición a *modo coordinado*
- *motion.tooloffset.x* … *motion.tooloffset.w* - (float, out, uno por eje) muestra el desplazamiento de la herramienta en efecto; podría provenir de la tabla de herramientas (*G43* activo), o podría venir del gcode (*G43.1* activo)
- *spindle.0.at-speed* - (bit, in) El movimiento se detendrá hasta que este pin sea TRUE, bajo las siguientes condiciones:
  - antes del primer movimiento de alimentación, después de cada arranque de husillo. o cambio de velocidad.
  - antes del inicio de cada cadena de movimientos sincronizados con el husillo.
  - si está en modo CSS, en cada transición de velocidad rapida a velocidad de alimentacion.

  Esta entrada se puede usar para asegurar que el husillo esté a su velocidad antes de comenzar un corte, o que un husillo de torno en modo CSS se ha ralentizado después de un pase de refrentado grande a pequeño antes de comenzar el próximo pase en el diámetro grande. Muchos VFD tienen una salida *a velocidad*. De lo contrario, es fácil generar esta señal con el componente HAL *near* mediante la comparación de las velocidades solicitadas y reales del eje.
- *spindle.N.brake* - (bit, out) TRUE cuando se debe aplicar el freno del husillo
- *spindle.N.forward* - (bit, out) TRUE cuando el husillo debe girar en sentido normal.
- *spindle.N.index-enable* - (bit, I/O) Para un funcionamiento correcto de los movimientos sincronizados del eje, este pin debe estar conectado con el pin de habilitación de índice del encoder del husillo.
- *spindle.N-inhibit* - (bit, in) Cuando este bit es TRUE, la velocidad del husillo se establece en 0.
- *spindle.N.on* - (bit, out) TRUE cuando el husillo debe rotar.
- *spindle.N.reverse* - (bit, out) TRUE cuando el husillo debe girar en sentido contrario
- *spindle.N.revs* - (float, in) Para un funcionamiento correcto de los movimientos sincronizados del husillo, esta señal debe estar enganchada al pin de posición del encoder del husillo. La posición del encoder del husillo debe escalarse de manera que spindle-revs aumente en 1 por cada rotación del husillo en el sentido de las agujas del reloj (*M3*).
- *spindle.N.speed-in* - (float, in) Retroalimentación de la velocidad real del husillo en rotaciones por segundo. Esto es utilizado en movimientos de avance por revolución (*G95*). Si su controlador del encoder del husillo no tiene salida de velocidad, puede generar uno adecuado enviando la posición del husillo a través de un componente *ddt*. Si no tiene un encoder de husillo, puede hacer bucle con *spindle.N.speed-out-rps*.
- *spindle.N.speed-out* - (float, out) Velocidad ordenada del husillo en rotaciones por minuto. Positivo para giro horario (*M3*), negativo para giro antihorario (*M4*).
- *spindle.N.speed-out-abs* - (float, out) Velocidad ordenada del husillo en rotaciones por minuto. Siempre sera un número positivo.
- *spindle.N.speed-out-rps* - (float, out) Velocidad del husillo ordenada en rotaciones por segundo. Positivo para sentido horario (*M3*), negativo para sentido antihorario (*M4*).
- *spindle.N.speed-out-rps-abs* - (float, out) Velocidad del husillo ordenada en rotaciones por segundo. Siempre sera un número positivo.
- 'spindle.N.orient-angle` - (float, out) Orientación del husillo especificada por M19. Valor del parámetro de la palabra R de M19 más el valor del parámetro ini [RS274NGC]ORIENT_OFFSET.
- 'spindle.N.orient-mode` - (s32, out) Modo de rotación de husillo para M19. Modo predeterminado = 0 (el menor angulo).
- 'spindle.N.orient` - (bit, out) Indica el inicio del ciclo de orientación del husillo. Activado por M19. Desactivado por M3, M4 o M5. Si spindle-orient-fault no es cero mientras que spindle-orient es TRUE, el comando M19 falla con un mensaje de error.
- 'spindle.N.is-oriented` - (bit, in) Pin de confirmación de spindle-orient. Completa el ciclo de orientacion. Si spindle-orient era verdadero cuando spindle-is-oriented se activa, el pin spindle-orient se borra y el pin spindle-locked se activa. Además, se activa el pin del freno del husillo.
- 'spindle.N.orient-fault` - (s32, in) Entrada del código de fallo para el ciclo de orientacion. Cualquier valor distinto de cero provocará que el ciclo de orientacion se aborte.

- 'spindle.N.lock` - (bit, out) Pin de orientacion de husillo completada. Desactivado por M3, M4 o M5.

Uso del pin HAL de orientacion del husillo M19. Conceptualmente, el husillo está en uno de los siguientes modos:

- modo de rotación (predeterminado)
- modo de busqueda de orientación deseada
- modo de orientación completada.

Cuando se ejecuta un M19, el husillo cambia a *buscando el modo de orientación deseado*, y se activa el pin HAL `spindle.N.orient`. La posición objetivo deseada se especifica mediante los pines `spindle.N.orient-angle` y `spindle.N.orient-fwd`, segun los parámetros R y P de M19.

Se espera que la lógica de soporte HAL reaccione a `spindle.N.orient` moviendo el husillo a la posición deseada. Cuando esto se completa, se espera que la lógica HAL lo reconozca activando el pin `spindle.N.is-oriented`.

A continuación, motion reconoce esto desactivando el pin `spindle.N.orient` y activando el pin `spindle.N.locked` para indicar el modo *orientación completa*.

Si mientras que `spindle.N.orient` es verdadero, `spindle.N.is-oriented` no ha sido aun activado y el pin `spindle.N.orient-fault` tiene un valor diferente a cero, el comando M19 se cancela, se muestra un mensaje que incluye el código de fallo, y la cola de movimiento se vacía. El husillo vuelve a modo de rotación.

Además, cualquiera de los comandos M3, M4 o M5 cancela los modos de *busqueda de orientación deseada* o *orientación completa*. Esto queda indicado al desactivar los pins *spindle-orient* y *spindle-locked*.

El pin *spindle-orient-mode* refleja la palabra M19 P y debe ser interpretado de la siguiente manera:

- 0: girar en sentido horario o antihorario, el que obtenga el movimiento angular más pequeño.
- 1: girar siempre en el sentido horario.
- 2: girar siempre en sentido antihorario.

Se puede usar con el componente HAL `orient` que proporciona un PID basado en la posición del encoder del husillo, `spindle-orient-angle` y `spindle-orient-mode`.

14.5.1.3. Parámetros

Muchos de estos parámetros sirven como ayudas para la depuración, y están sujetos a cambio o eliminación en cualquier momento.

- *motion-command-handler.time* - (s32, RO)
- *motion-command-handler.tmax* - (s32, RW)
- *motion-controller.time* - (s32, RO)
- *motion-controller.tmax* - (s32, RW)
- *motion.debug-bit-0* - (bit, RO) Se usa con fines de depuración.
- *motion.debug-bit-1* - (bit, RO) Se usa con fines de depuración.
- *motion.debug-float-0* - (flotante, RO) Se usa con fines de depuración
- *motion.debug-float-1* - (flotante, RO) Se usa con fines de depuración
- *motion.debug-float-2* - (flotante, RO) Se usa con fines de depuración
- *motion.debug-float-3* - (flotante, RO) Se usa con fines de depuración

- *motion.debug-s32-0* - (s32, RO) Se usa con fines de depuración
- *motion.debug-s32-1* - (s32, RO) Se usa con fines de depuración
- *motion.servo.last-period* - (u32, RO) El número de ciclos de CPU entre las invocaciones del hilo servo. Normalmente este número, dividido por la velocidad de la CPU, da el tiempo en segundos, y se puede usar para determinar si el movimiento en tiempo real del controlador cumple con sus restricciones de tiempo
- *motion.servo.last-period-ns* - (flotar, RO)

14.5.1.4. Funciones

En general, estas funciones se agregan al hilo servo en el orden mostrado.

- *motion-command-handler* - Procesa comandos de movimiento provenientes del espacio de usuario
- *motion-controller* - Ejecuta el controlador de movimiento LinuxCNC

14.5.2. Pines y parámetros de ejes y articulaciones

Estos pines y parámetros son creados por el modulo *motmod* en tiempo real. [En las máquinas de "cinemática trivial", hay una correspondencia uno-a-uno entre articulaciones y ejes.] Son leídos y actualizados por la función *motion-controller*.

Consulte la página del manual *motion(9)* para obtener detalles sobre los pines y parámetros.

14.5.3. iocontrol

iocontrol - acepta comandos de E/S NML, interactúa con HAL en el espacio de usuario.

Las señales se activan y desactivan en el espacio de usuario. Si tiene requisitos de tiempo estrictos o simplemente necesita más E/S, considere usar el tiempo real I/O sincronizado proporcionado por motion en su lugar.

14.5.3.1. Pines

- *iocontrol.0.coolant-flood* - (bit, out) TRUE cuando se solicita refrigerante de inundación.
- *iocontrol.0.coolant-mist* - (bit, out) TRUE cuando se solicita refrigerante de niebla.
- *iocontrol.0.emc-enable-in* - (bit, in) Debe ser FALSE cuando exista una condición externa de E-Stop.
- *iocontrol.0.lube* - (bit, out) TRUE cuando se activa el lubricante.
- *iocontrol.0.lube_level* - (bit, in) Debe ser TRUE cuando el nivel de lubricante es correcto.
- *iocontrol.0.tool-change* - (bit, out) TRUE cuando se solicita un cambio de herramienta.
- *iocontrol.0.tool-changed* - (bit, in) Debe ser TRUE cuando se completa un cambio de herramienta.
- *iocontrol.0.tool-number* - (s32, fuera) El número de herramienta actual.
- *iocontrol.0.tool-prep-number* - (s32, out) El número de la siguiente herramienta, de la palabra T RS274NGC.
- *iocontrol.0.tool-prepare* - (bit, out) TRUE cuando se solicita preparación de una herramienta.
- *iocontrol.0.tool-prepared* - (bit, in) Debe ser TRUE cuando se completa una preparación de herramienta.
- *iocontrol.0.user-enable-out* - (bit, out) FALSE cuando existe una condición de parada de emergencia interna.
- *iocontrol.0.user-request-enable* - (bit, out) TRUE cuando el usuario ha solicitado que se borre el E-Stop.

14.5.4. Configuración ini

Muchos items de configuracion ini están disponibles como pines de entrada hal.

14.5.4.1. Pines

N se refiere a un número de articulacion, L se refiere a una letra de eje

- *ini.N.ferror* - (float, in) [JOINT_N]FERROR
- *ini.N.min_ferror* - (float, in) [JOINT_N]MIN_FERROR
- *ini.N.backlash* - (float, in) [JOINT_N]BACKLASH
- *ini.N.min_limit* - (float, in) [JOINT_N]MIN_LIMIT
- *ini.N.max_limit* - (float, in) [JOINT_N]MAX_LIMIT
- *ini.N.max_velocity* - (float, in) [JOINT_N]MAX_VELOCITY
- *ini.N.max_acceleration* - (float, in) [JOINT_N]MAX_ACCELERATION
- *ini.N.home* - (float, in) [JOINT_N]HOME
- *ini.N.home_offset* - (float, in) [JOINT_N]HOME_OFFSET
- *ini.N.home_offset* - (s32, in) [JOINT_N]HOME_SEQUENCE ??
- *ini.L.min_limit* - (float, in) [AXIS_L]MIN_LIMIT
- *ini.L.max_limit* - (float, in) [AXIS_L]MAX_LIMIT
- *ini.L.max_velocity* - (float, in) [AXIS_L]MAX_VELOCITY
- *ini.L.max_acceleration* - (float, in) [AXIS_L]MAX_ACCELERATION

nota
Los pines min_limit y max_limit por eje se respetan continuamente continua después de homing. Los pines ferror y min_ferror por eje se respetan cuando la máquina está encendida y no en posición Los pines max_velocity y max_acceleration por eje se muestrean cuando la máquina está encendida y motion_state es libre (homing o jog) pero no son muestreados cuando se está ejecutando un programa (modo automático) o en modo mdi. Por consiguiente, cambiar los valores de pin cuando un programa se está ejecutando no tendrá efecto hasta el programa se detiene y motion_state vuelve a estar libre.

- *ini.traj_arc_blend_enable* - (bit, in) [TRAJ]ARC_BLEND_ENABLE
- *ini.traj_arc_blend_fallback_enable* - (bit, in) [TRAJ]ARC_BLEND_FALLBACK_ENABLE
- *ini.traj_arc_blend_gap_cycles* - (float, in) [TRAJ]ARC_BLEND_GAP_CYCLES
- *ini.traj_arc_blend_optimization_depth* - (float, in) [TRAJ]ARC_BLEND_OPTIMIZATION_DEPTH
- *ini.traj_arc_blend_ramp_freq* - (float, in) [TRAJ]ARC_BLEND_RAMP_FREQ

nota
Los pines traj_arc_blend se muestrean continuamente pero cambiar los valores de pin mientras se ejecuta un programa puede no tener efecto inmediato debido a la cola de comandos.

- *ini.traj_default_acceleration* - (float, in) [TRAJ]DEFAULT_ACCELERATION
- *ini.traj_default_velocity* - (float, in) [TRAJ]DEFAULT_VELOCITY
- *ini.traj_max_acceleration* - (float, in) [TRAJ]MAX_ACCELERATION

14.6. Interfaces de dispositivos canónicos

14.6.1. Introducción

Las siguientes secciones muestran los pines, parámetros y funciones suministrados por "dispositivos canónicos". Todos los controladores de dispositivo HAL deben suministrar los mismos pines y parámetros, e implementar el mismo comportamiento.

Tenga en cuenta que solo los campos `<io-type>` y `<specific-name>` estan definido para un dispositivo canónico. Los campos `<device-name>`, `<device-num>`, y `<chan-num>` se establecen en función de las características del dispositivo real.

14.6.2. Entrada digital

La entrada digital canónica (campo tipo E/S:`digin`) es bastante simple.

14.6.2.1. Pines

- (bit) **in** — Estado de la entrada hardware.
- (bit) **in-not** — Estado invertido de la entrada.

14.6.2.2. Parámetros

- Ninguno

14.6.2.3. Funciones

- (funct) **read** — Lee el hardware y configura los pines HAL *in* y 'in-not`.

14.6.3. Salida digital

La salida digital canónica (campo de tipo E/S: `digout`) también es muy sencilla.

14.6.3.1. Pines

- (bit) * out * — Valor a escribir (posiblemente invertido) en la salida hardware.

14.6.3.2. Parámetros

- (bit) **invert** — Si es TRUE, **out** se invierte antes de escribir en el hardware.

14.6.3.3. Funciones

- (funct) **write** — Lee **out** e **invert** y establece la salida hardware en consecuencia.

14.6.4. Entrada Analógica

La entrada analógica canónica (tipo E/S: `adcin`) se espera que sea utilizada para convertidores analógico a digital que, por ejemplo, conviertan voltaje a un rango continuo de valores.

14.6.4.1. Pines

- (float) **value** — La lectura del hardware, escalada según los parámetros **scale** y **offset**. **Value** = ((lectura de entrada, en unidades dependientes del hardware) * **scale**) - **offset**

14.6.4.2. Parámetros

- (float) **scale** — El voltaje de entrada (o corriente) se multiplicará por **scale** antes de salir a **value**.
- (float) **offset** — Se restará de la entrada de voltaje (o corriente) hardware después de que se haya aplicado el multiplicador de escala.
- (float) **bit_weight** — El valor del bit menos significativo(LSB). Es efectivamente la granularidad de la lectura de entrada.
- (float) **hw_offset** — El valor presente en la entrada cuando se aplican 0 voltios al pin(es) de entrada.

14.6.4.3. Funciones

- (funct) **read** — Lee los valores de este canal de entrada analógica. Se puede usar para leer un canal individual, o puede hacer que se lean todos los canales

14.6.5. Salida analógica

La salida analógica canónica (tipo E/S: **adcout**). Esto esta pensado para cualquier tipo de hardware que pueda generar un rango de valores más o menos continuo. Los ejemplos son convertidores de digital a analógico o generadores PWM.

14.6.5.1. Pines

- (float) **value** — El valor que se escribirá. El valor real de salida para el hardware dependerá de la escala y los parámetros de offset.
- (bit) **enable** — Si es falso, entonces salida al hardware es 0, independientemente del pin **value**.

14.6.5.2. Parámetros

- (float) **offset** — Esto se agregará a **value** antes de la actualizacion del hardware.
- (float) **scale** — Se debe configurar de modo que una entrada de 1 en el pin **value** causará que en el pin de salida analógica se lea 1 voltio.
- (float) **high_limit** (opcional) — Cuando se calcula el valor para salida al hardware, si **value + offset** es mayor que **high_limit**, se usará **high_limit** .
- (float) **low_limit** (opcional) — Cuando se calcula el valor de salida para el hardware, si **value + offset** es menor que **low_limit**, se usará **low_limit**.
- (float) **bit_weight** (opcional) — El valor del bit menos significativo (LSB), en voltios (o mA, para salidas de corriente)
- (float) **hw_offset** (opcional) — El voltaje (o corriente) real que saldrá si se escribe 0 en el hardware.

14.6.5.3. Funciones

(funct) **write** — Hace que el valor calculado sea enviado al hardware. Si el pin enable es falso, la salida será 0, independientemente de **value**, **scale** y **offset**. El significado de "0" depende del hardware. Por ejemplo, un A/D bipolar de 12 bits puede necesitar escribir 0x1FF (escala intermedia) para obtener D/A 0 voltios desde el pin de hardware. Si el pin enable es verdadero, lee scale, offset y value y pone a la salida del ADC (**scale** * **value**) + **offset**.

=Herramientas HAL

==Halcmd

Halcmd es una herramienta de línea de comandos para manipular HAL. Hay una página del manual bastante completa para halcmd, que se instalará si ha instalado LinuxCNC desde una fuente o un paquete. La página de manual proporciona información de uso:

```
man halcmd
```

Si ha compilado LinuxCNC para "ejecutar en el lugar (RIP)", debe correr el script rip-environment para hacer que la página man esté disponible:

```
cd toplevel_directory_for_rip_build
. scripts/rip-environment
man halcmd
```

El capitulo Tutorial HAL es un buen tutorial para halcmd y tiene varios ejemplos de uso.

==Halmeter

Halmeter es un *voltímetro* para HAL. Le permite ver el valor actual de un pin, señal, o parámetro. Es bastante simple de usar. Comience escribiendo **halmeter** en una ventana de shell. Halmeter es una aplicación GUI. Aparecerá una ventana pequeña, con dos botones etiquetados *Seleccionar* y *Salir*. La salida es obvia - apaga el programa. Con Seleccionar aparece una ventana más grande, con tres pestañas. Una pestaña muestra todos los pines actualmente definidos en HAL, la siguiente enumera todas las señales, y la última pestaña lista todos los parámetros. Haga clic en una pestaña, luego haga clic en un pin/señal/parámetro. Entonces haga clic en *OK*. Las listas desaparecerán, y la ventana pequeña mostrará el nombre y el valor del elemento seleccionado. La pantalla se actualiza aproximadamente 10 veces por segundo. Si hace clic en "Aceptar" en lugar de *OK*, la ventana pequeña mostrará el nombre y el valor del elemento seleccionado, pero la ventana grande permanecerá en la pantalla. Esto es conveniente si desea ver una serie de elementos diferentes rápidamente.

Puede tener muchos halmeters funcionando al mismo tiempo, si desea controlar varios elementos. Si quiere lanzar un halmeter sin depender de una ventana de shell, escriba *halmeter &* para ejecutarlo en segundo plano. También puedes hacer que halmeter comience mostrando un elemento específico inmediatamente, agregando *pin|sig|par[am]<nombre>* a la línea de comando. Mostrará el pin, la señal o el parámetro <nombre> tan pronto como comienze. (Si no hay tal elemento, comienza normalmente). Y, finalmente, si especifica un elemento a mostrar, usted puede agregar *-s* antes de pin|sig|param para indicar a halmeter que use una pequeña ventana. El nombre del elemento se mostrará en la barra de título en lugar de debajo del valor, y no habrá botones. Útil cuando quiere muchos halmeter en una pequeña cantidad de espacio en la pantalla.

Consulte la sección Tutorial Halmeter para obtener más información.

Halmeter se puede cargar desde una terminal o desde Axis y es más rápido que Halshow en la visualización de valores. Halmeter tiene dos ventanas, una para seleccionar el pin, señal o parámetro para monitorear y una que muestra el valor. Pueden abrirse multiples Halmeter al mismo tiempo. Si utiliza una secuencia de comandos para abrir múltiples Halmeters, puede establecer la posición de cada uno con -g X Y en relación con la esquina superior izquierda de la pantalla. Por ejemplo:

```
loadusr halmeter pin hm2.0.stepgen.00.velocity-fb -g 0 500
```

Vea la página man para más opciones. Vea tambien la sección Halmeter.

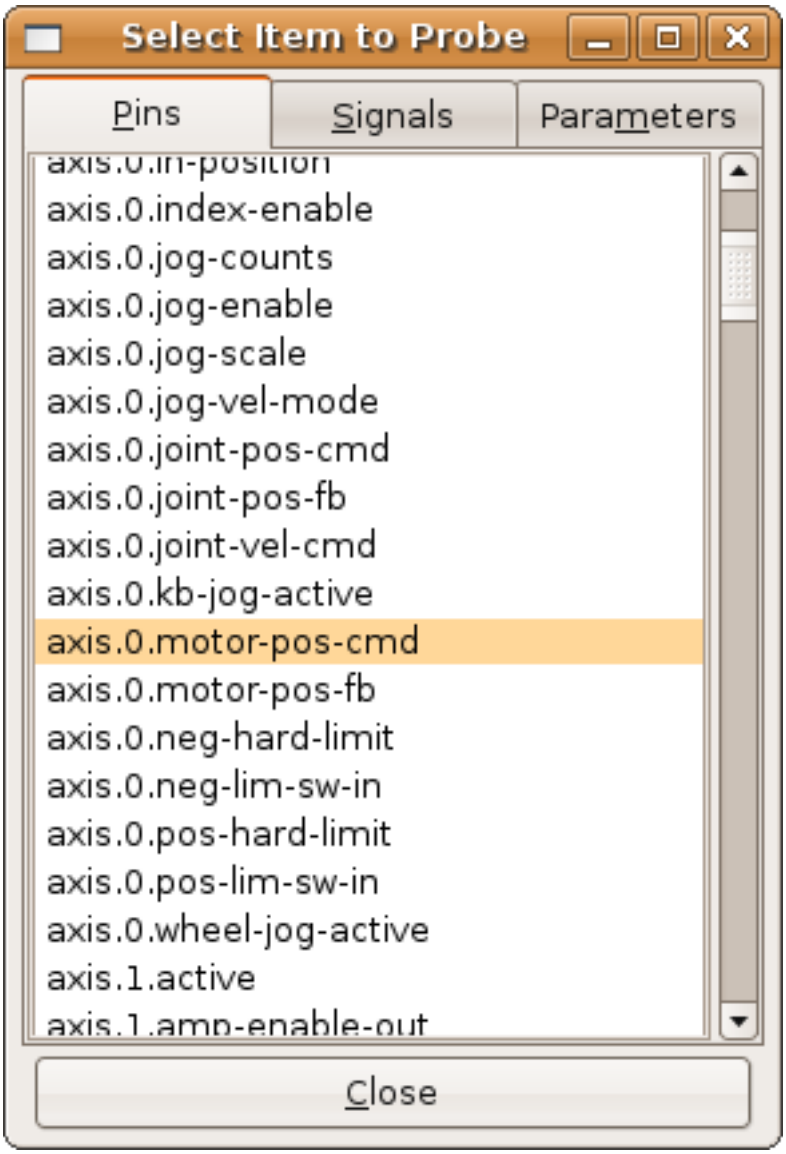
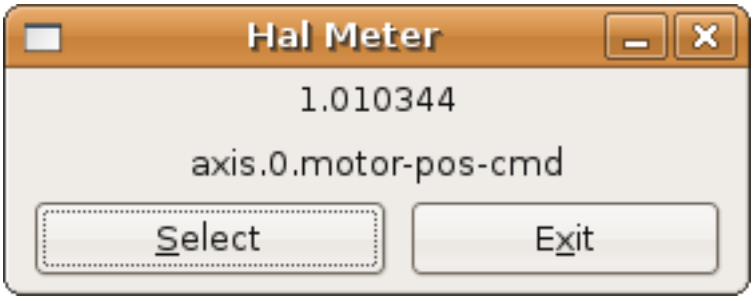


Figura 14.17: Halmeter



14.6.6. Halshow

Halshow (ver sección separada para descripción completa de uso) se puede iniciar desde la línea de comando para mostrar los detalles de los componentes seleccionados, pines, parámetros, señales, funciones e hilos de un HAL en ejecución. La pestaña WATCH proporciona una visualización continua de pines, parámetros y elementos de señal. El menú Archivo proporciona botones para 1) guardar los elementos observados en una lista de vigilancia y para cargar una lista de vigilancia existente. Los elementos de la lista de observación pueden también cargarse automáticamente al inicio. Para su uso, en línea de comandos:

```
Usage:
  halshow [Options] [watchfile]
  Options:
           --help    (this help)
           --fformat format_string_for_float
           --iformat format_string_for_int

Notes:
       Create watchfile in halshow using: 'File/Save Watch List'
       linuxcnc must be running for standalone usage
```

LinuxCNC debe estar ejecutanse para uso independiente de Halshow.

El archivo watchfile creado utilizando la opción de menú *Archivo/Guardar lista de observación* está formateado como una sola línea con tokens "pin+", "param+", "sig=+" seguido del nombre de pin, param o señal apropiado. Los pares token-nombre están separados por un carácter de espacio.

Ejemplo: pin+joint.0.pos-hard-limit pin+joint.1.pos-hard-limit sig+estop-loop

Un archivo watchfile creado utilizando la opción de menú *Archivo/Guardar lista de observación (multilínea)* está formateado con líneas separadas para cada elemento identificado con pares token-nombre como se describe arriba.

Ejemplo: pin+joint.0.pos-hard-limit pin+joint.1.pos-hard-limit sig+estop-loop

Al cargar un archivo watchfile con el elemento de menú *Archivo/cargar lista de observación*, los pares de nombres de tokens pueden aparecer como líneas simples o múltiples. Líneas en blanco y líneas que comienzan con un caracter # son ignoradas.

14.6.7. Halscope

Halscope es un "osciloscopio" para HAL. Le permite capturar el valor de pines, señales y parámetros como una función del tiempo. Las instrucciones de operación deben ubicarse aquí eventualmente. Por ahora, consulte la sección [sec:tutorial-halscope] en el capítulo tutorial, que explica los conceptos básicos.

El selector del menú Archivo de halscope proporciona botones para guardar una configuración o abrir una configuración previamente guardada. Cuando halscope termina, la última configuración se guarda en un archivo llamado autosave.halscope.

Los archivos de configuración también se pueden especificar al iniciar halscope desde la línea de comando. Ayuda de uso en linea de comandos (-h):

```
halscope -h
Uso:
  halscope [-h] [-i infile] [-o archivo de salida] [num_samples]
```

14.6.8. Sim Pin

sim_pin es una utilidad de línea de comandos para mostrar y actualizar cualquier cantidad de Pines, parámetros o señales editables. Uso:

```
  Uso:
         sim_pin [Opciones] nombre1 [nombre2 ...] &

  Opciones:
         --ayuda (este texto)
         --title title_string (título de la ventana, predeterminado: sim_pin)

  Nota: LinuxCNC (o una aplicación Hal independiente) debe estar ejecutándose
         Un elemento con nombre puede especificar un pin, param o señal
         El elemento debe ser escribible, por ejemplo:
            pin: IN o I/O (y no conectado a una señal con un escritor)
            param: RW
```

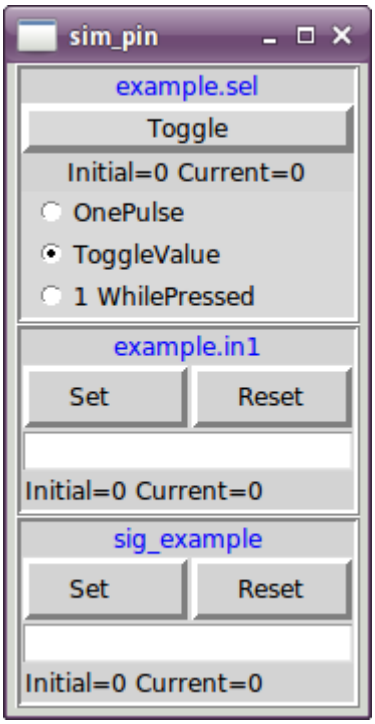
```
            señal: conectado a un pin de escritura

         Se admiten los tipos Hal de elementos bit, s32, u32 y float

         Cuando se especifica un elemento de bit, se crea un botón
         para administrar el item de una de las tres maneras especificadas
         por botones de radio:
             alternar: alternar el valor cuando se presiona el botón
             pulso: Pulse el elemento a 1 una vez cuando se presiona el botón
             mantener: se establece en 1 con el botón presionado
         El modo de botón de bit se puede especificar en la
         línea de comandos formateando el nombre del elemento:
             namei/mode=[toggle | pulse | hold]
         Si el modo comienza con una letra mayúscula,
         no se muestran los botones radio para seleccionar otros modos
```

Para obtener información completa, consulte la página man:

```
man sim_pin
```

Ejemplo (con LinuxCNC ejecutándose):

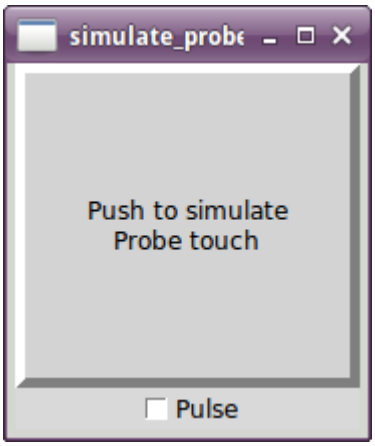
```
halcmd loadrt mux2 names=example; halcmd net sig_example example.in0
sim_pin example.sel example.in1 sig_example &
```



14.6.9. Sonda simulada

simulate_probe es una gui simple para simular la activación del pin motion.probe-input. Uso:

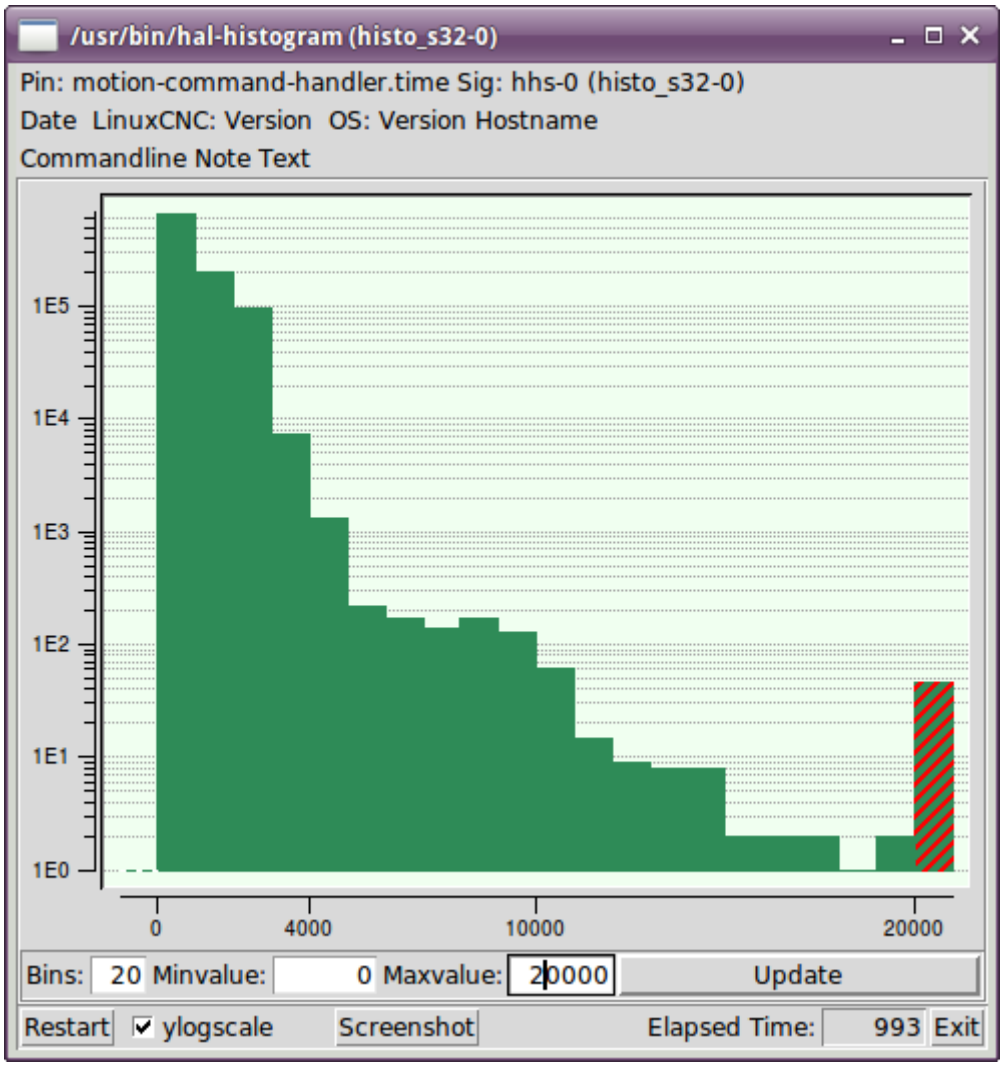
```
simulate_probe &
```



14.6.10. Hal Histogram

hal-histogram es una utilidad de línea de comandos para mostrar histogramas para pines hal.

```
Uso:
   hal-histograma --help | -?
o
   hal-histogram [Opciones] [pinname]

Opciones:
  --minvalue minvalue  (bin mínimo, por defecto: 0)
  --binsize binsize    (binsize, por defecto: 100)
  --nbins nbins        (número de contenedores, por defecto: 50)

  --logscale  0|1      (escala logaritmica del eje y, valor predeterminado: 1)
  --text      note     (visualización de texto, por defecto: "")
  --show               (muestra el conteo de nbins no mostradas, desactivado por defecto)
  --verbose            (progreso y depuración, desactivado por defecto)

Notas:
  1) LinuxCNC (u otra aplicación Hal) debe estar ejecutándose
  2) Si no se especifica ningún nombre pin, el valor predeterminado es: motion-command- ↩
      handler.time
  3) Esta aplicación se puede abrir para hasta 5 pines
  4) Los tipos float, s32, u32, bit son compatibles
  5) El pin debe estar asociado con un hilo que soporte punto flotante
     Para un hilo base, esto puede requerir el uso de:
     loadrt motmod ... base_thread_fp=1
```



14.6.11. Halreport

halreport es una utilidad de línea de comandos que genera un informe sobre conexiones Hal para una aplicación LinuxCNC (u otra hal) en ejecución. El informe muestra todas las conexiones de señal y desvela posibles problemas. Información incluida:

1. Descripción del sistema y versión del kernel.
2. Señales y todos los pines de salida, io y entrada conectados.
3. De cada pin, component_function, hilo y orden en addf.
4. Pines de componentes de espacio de usuario que tienen funciones no ordenadas.
5. Identificación de funciones desconocidas para componentes no manejados.
6. Señales sin salida.
7. Señales sin entradas.
8. Funciones sin addf.
9. Etiquetas de advertencia para componentes marcados como desaconsejados/obsoletos en los documentos.
10. Nombres reales para pines que usan nombres de alias.

El informe puede generarse desde la línea de comando y dirigirse a un archivo de salida (o stdout si no se especifica archivo de salida):

```
Uso:
  halreport -h | --help (this help)
o
  halreport [outfilename]
```

Para generar el informe para cada inicio de LinuxCNC, incluya halreport y un nombre de archivo de salida como una entrada de [APPLICATIONS]APP en el archivo ini. Ejemplo:

```
[APPLICATIONS]
APP = halreport /tmp/halreport.txt
```

El ordén addf de funciones puede ser importante para bucles servo donde la secuencia de las funciones calculadas en cada período servo es importante. Por lo general, el orden es: leer los pines de entrada, hacer las funciones del manejador de comandos y controlador de movimiento, realizar los cálculos pid y, finalmente, escribir los pines de salida.

Para cada señal en una ruta crítica, el orden addf del pin de salida debe ser numéricamente más bajo que el orden addf de los pines de entrada críticos a los que se conecta.

Para rutas de señal de rutinas que manejan entradas de interruptores, pines de espacio de usuario, etc., el ordenamiento addf a menudo no es crítico. Además, el momento de los cambios de valor de pin de espacio de usuario no se puede controlar o garantizar en los intervalos típicamente empleados para hilos hal.

Ejemplo de archivo de informe que muestran un bucle pid para hostmot2 stepgen operado en modo de velocidad en una máquina Trivkins con joint.0 correspondiente a la coordenada del eje X:

```
SIG:    pos-fb-0
  OUT:    h.00.position-fb                    hm2_7i92.0.read        servo-thread 001
          (=hm2_7i92.0.stepgen.00.position-fb)
    IN:     X_pid.feedback                    X_pid.do-pid-calcs     servo-thread 004
    IN:     joint.0.motor-pos-fb              motion-command-handler servo-thread 002
            ...................               motion-controller      servo-thread 003
...
SIG:    pos-cmd-0
  OUT:    joint.0.motor-pos-cmd               motion-command-handler servo-thread 002
          .....................               motion-controller      servo-thread 003
    IN:     X_pid.command                     X_pid.do-pid-calcs     servo-thread 004
...
SIG:    motor-cmd-0
  OUT:    X_pid.output                        X_pid.do-pid-calcs     servo-thread 004
    IN:     h.00.velocity-cmd                 hm2_7i92.0.write       servo-thread 008
            (=hm2_7i92.0.stepgen.00.velocity-cmd)
```

En el ejemplo anterior, el HALFILE usa alias halcmd para simplificar los nombres de los pines para una placa hostmot2 fpga con comandos como:

```
alias pin hm2_7i92.0.stepgen.00.position-fb h.00.position-fb
```

nota
La detección de funciones de componentes cuestionables puede ocurrir para 1) componentes no compatibles (obsoletos), 2) componentes creados por el usuario que usan múltiples funciones o nombres de funciones no convencionales, o 3) componentes de espacio de usuario creados por GUI que carecen de características de distinción como un prefijo basado en el nombre del programa gui. Las funciones cuestionables están etiquetadas con un signo de interrogación "?".

nota
Los pines de componentes que no pueden asociarse con una función en un hilo conocido informan la función como "Desconocida".

nota
halreport genera un informe de conexiones para una aplicación hal en ejecución para ayudar en el diseño y verificación de conexiones. No se muestran tipos y valores actuales de pines. Para esta información use aplicaciones como halshow, halmeter, halscope o el comando *show* disponible con el programa halcmd.

14.7. Halshow

El script halshow puede ayudarle a revisar HAL en ejecución. Este es un sistema muy especializado y debe conectarse con HAL en funcionamiento. No puede ejecutarse de forma independiente porque depende de la capacidad de HAL para informar de lo que sabe de sí mismo a través de la biblioteca de interfaz halcmd; esto se basa en el descubrimiento de sus propias capacidades. Cada vez que halshow se ejecuta con un LinuxCNC diferente, la configuración será diferente.

Como veremos, esta capacidad de HAL para documentarse es clave para hacer un sistema CNC efectivo.

14.7.1. Comenzando Halshow

Halshow se encuentra en el menú AXIS en Máquina > Mostrar Configuración HAL.

Halshow está en TkLinuxCNC bajo el menú Scripts > HAL Show.

Halshow se puede iniciar desde una línea de comandos del terminal y especificar formatos para elementos enteros y de punto flotante (pines o señales) e identificar un archivo guardado de lista de seguimiento para su utilización:

```
$ halshow --help
Uso:
  halshow [Opciones] [watchfile]
  Opciones:
           --help (esta ayuda)
           --fformat format_string_for_float
           --iformat format_string_for_int

Notas:
       Cree un archivo de seguimiento en halshow usando: 'Archivo > Guardar lista de  ↩
           seguimiento'
       linuxcnc debe estar ejecutándose para uso independiente
```

Ejemplo para limitar a cinco el número de decimales para flotantes y usar un archivo llamado my.halshow en el directorio actual:

```
$ halshow --fformat ".5f" ./my.halshow
```

14.7.2. Area de Arbol HAL

A la izquierda de su pantalla, como se muestra en la figura, hay una vista de árbol parecido al de los navegadores de archivos. A la derecha hay un cuaderno con las pestañas SHOW (ver) y WATCH (observar).

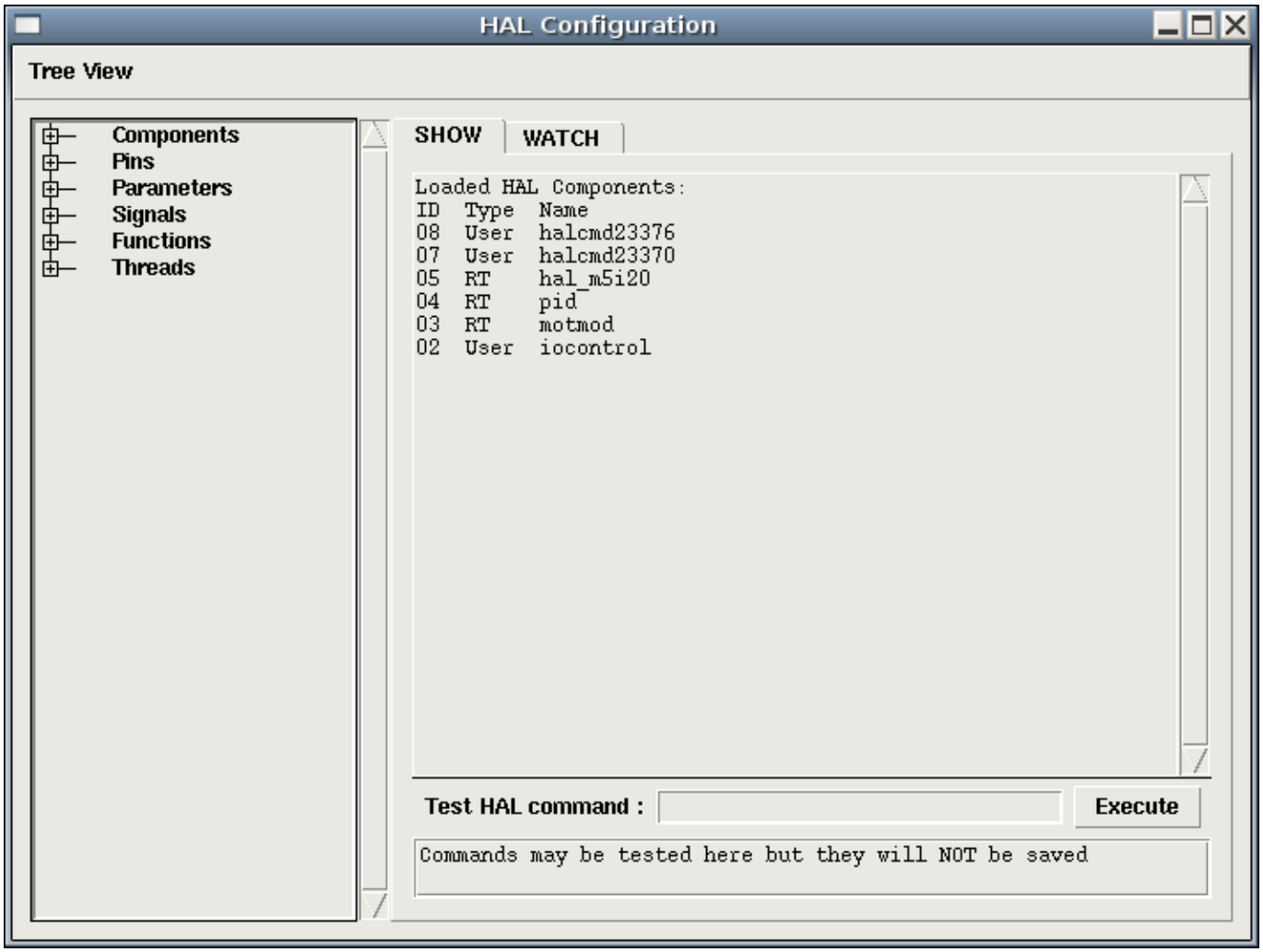


Figura 14.18: Ventana de Halshow

El árbol muestra todas las partes principales de HAL. Delante de cada una hay un pequeño signo más (+) o menos (-) en un recuadro. Al hacer clic en el signo (+) se expande el nodo del árbol para mostrar lo que está bajo de él. Si esa casilla muestra un signo menos (-), al hacer clic en él, contraerá esa sección del árbol.

También puede expandir o contraer la visualización del árbol usando el menu *Tree View* en el borde superior izquierdo de la pantalla. Bajo *Tree View* podrá encontrar: Expandir Árbol, Contraer árbol, Expandir pines, Expandir parámetros, Expandir Señales y Borrar Observacion (Erase Watch). (Tenga en cuenta que Erase Watch borra *todo* lo previamente establecido en WATCH; no puede borrar solo un item.)

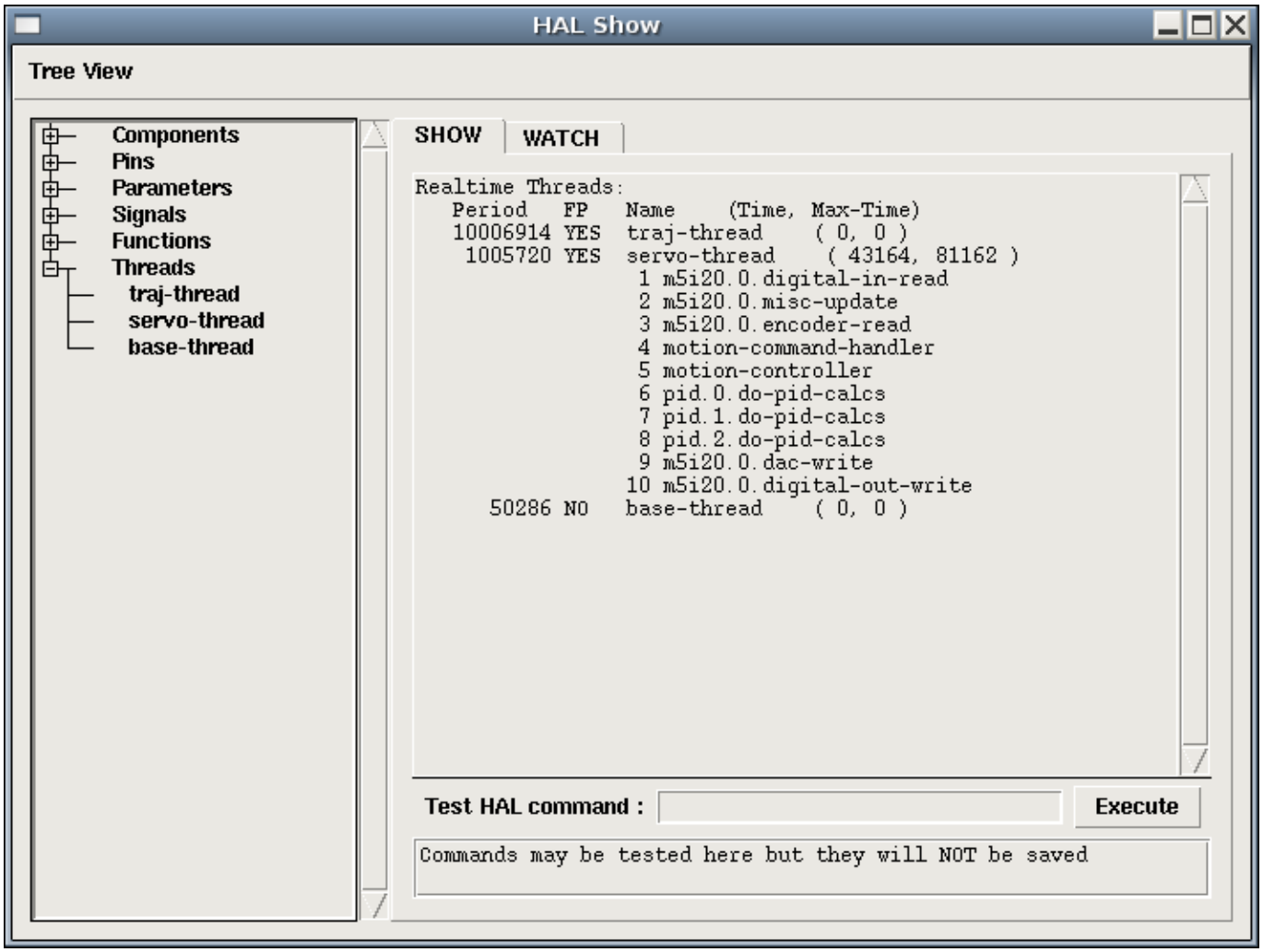


Figura 14.19: Pestaña "Show"

14.7.3. Area Show HAL

Al hacer clic en el nombre de un nodo, "Components" por ejemplo, mostrara (en la pestaña "SHOW") todos los contenidos HAL sobre ese nodo. La Figura Figura 14.18 muestra una lista exactamente como la veria si hace clic en el nombre de "Components" mientras ejecuta, por ejemplo, una tarjeta servo Mesa m5i20 estándar. La información en pantalla es exactamente como la que se muestran en las herramientas tradicionales de análisis de HAL basadas en texto. La ventaja aquí es que tenemos acceso con un clic del mouse, acceso que puede ser tan amplio o tan enfocado como se necesite.

Si echamos un vistazo más de cerca a la pantalla del árbol podemos ver que las seis partes principales de HAL pueden expandirse al menos en un nivel. Cuando estos niveles se expanden, puede enfocarse más lo que interesa haciendo clic en el nodo de árbol más a la derecha. Usted encontrará que hay algunos pines y parámetros HAL que muestran más de una respuesta. Esto se debe a la naturaleza de las rutinas de búsqueda en halcmd. Si busca un pin determinado, puede obtener dos, como en este caso:

```
Component Pins:
Owner  Type  Dir  Value  Name
06     bit    -W   TRUE  parport.0.pin-10-in
06     bit    -W  FALSE  parport.0.pin-10-in-not
```

El nombre del segundo pin contiene el nombre completo del primero.

Debajo del área de exhibición a la derecha hay un conjunto de widgets que permitirán interactuar con HAL. Los comandos que se

ingresan aquí y los efecto que tienen en el HAL en ejecución no se guardan. Seran persistentes mientras LinuxCNC permanezca activo, pero se borraran tan pronto como LinuxCNC se cierre.

El cuadro de entrada etiquetado "Comando de prueba HAL:" aceptará cualquiera de los comandos listados para halcmd. Éstos incluyen:

- loadrt, unloadrt (cargar/descargar módulo en tiempo real)
- loadusr, unloadusr (cargar/descargar componente de espacio de usuario)
- addf, delf (agregar/eliminar una función a/desde un hilo en tiempo real)
- net (crear una conexión entre dos o más elementos)
- setp (establecer parámetro o pin a un valor)

Este pequeño editor ingresará un comando cada vez que presione <enter> o presione el botón *Execute*. Cuando estos comandos no están formados correctamente, se mostrará un mensaje de error de halcmd. Si no está seguro de cómo configurar un comando adecuado, necesitará leer una vez más la documentación sobre halcmd y los módulos específicos que está utilizando.

Usemos este editor para agregar un módulo diferencial a un HAL y conéctarlo a la posición del eje para que podamos ver la tasa de cambio de posición, es decir, la aceleración. Primero necesitamos cargar un componente HAL llamado ddt; agréguelo al hilo del servo. Luego conéctelo al pin de posición de una articulación. Una vez hecho esto podemos encontrar la salida del diferenciador en halscope. Veamoslo.

```
loadrt ddt
```

Ahora mire el nodo de componentes y debería ver ddt en algún lugar.

```
Loaded HAL Components:
ID Type        Name
10 User halcmd29800
09 User halcmd29374
08   RT         ddt
06   RT hal_parport
05   RT    scope_rt
04   RT     stepgen
03   RT      motmod
02 User   iocontrol
```

Por supuesto, ahí está. Tenga en cuenta que su ID es 08. A continuación necesitamos averigüar qué funciones están disponibles con él, por lo que seleccionaremos el nodo funciones:

```
Exported Functions:
Owner  CodeAddr      Arg  FP Users Name
  08   E0B97630 E0DC7674 YES     0 ddt.0
  03   E0DEF83C 00000000 YES     1 motion-command-handler
  03   E0DF0BF3 00000000 YES     1 motion-controller
  06   E0B541FE E0DC75B8  NO     1 parport.0.read
  06   E0B54270 E0DC75B8  NO     1 parport.0.write
  06   E0B54309 E0DC75B8  NO     0 parport.read-all
  06   E0B5433A E0DC75B8  NO     0 parport.write-all
  05   E0AD712D 00000000  NO     0 scope.sample
  04   E0B618C1 E0DC7448 YES     1 stepgen.capture-position
  04   E0B612F5 E0DC7448  NO     1 stepgen.make-pulses
  04   E0B614AD E0DC7448 YES     1 stepgen.update-freq
```

Aquí buscamos el ID 08 y vemos una función llamada ddt.0. Deberíamos poder agregar ddt.0 al hilo del servo y hará sus cálculos cada vez que se actualice ese hilo. El comando *addf* usa tres argumentos como estos:

```
addf <functname> <nombre de hilo> [<posición>]
```

Ya conocemos la functname = ddt.0, así que obtengamos el nombre del hilo correcto expandiendo el nodo de hilos (Threads) en el árbol. Aquí vemos dos hilos; hilo servo e hilo base. La posición de ddt.0 en el hilo no es crítica. Agregemos la función ddt.0 al hilo servo:

```
addf ddt.0 servo-thread
```

Esto es solo para ver su valor, así que dejamos [<posición>] en blanco y la funcion queda en la última posición en el hilo. La siguiente figura muestra el estado de halshow después de que este comando ha sido emitido.

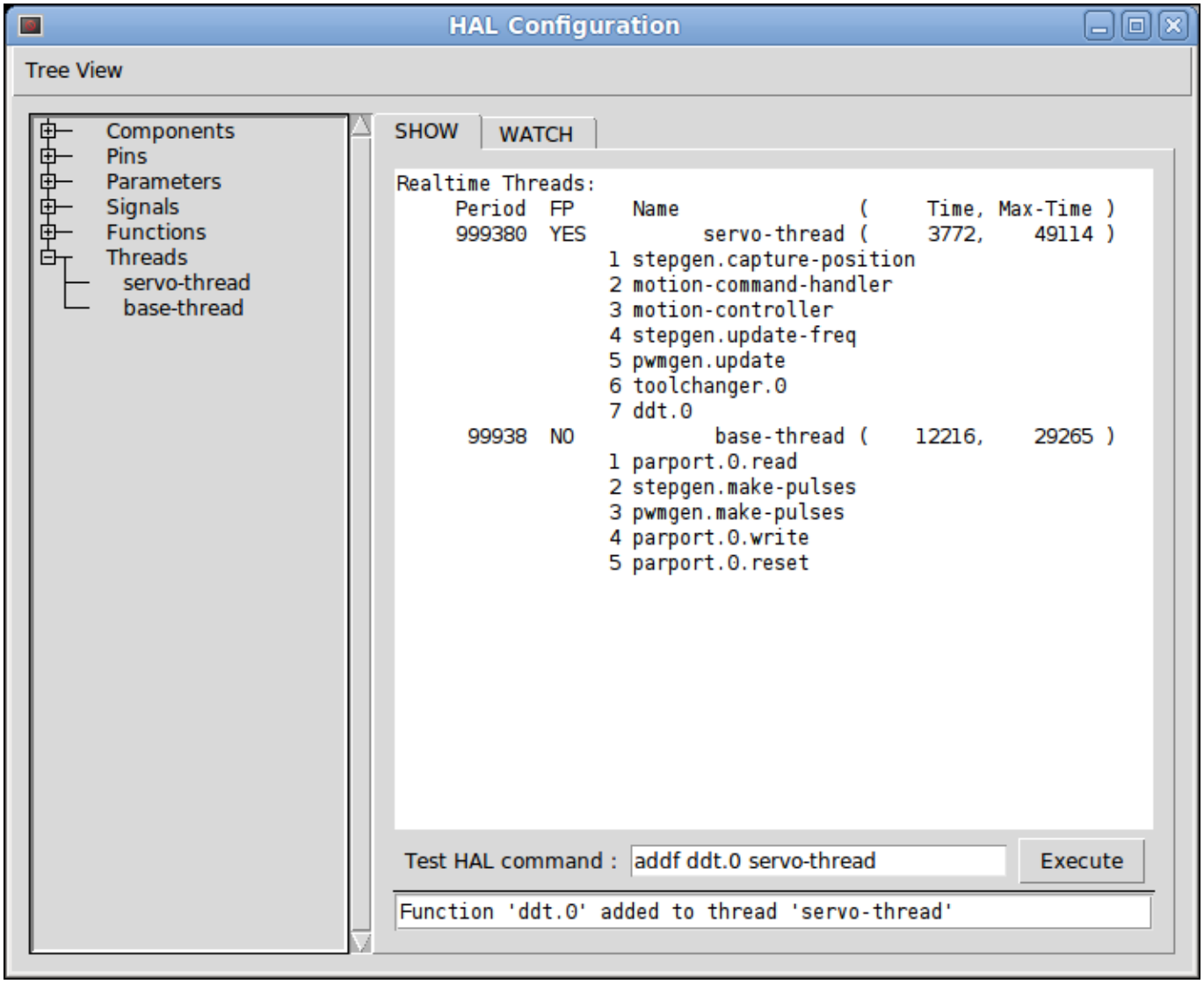


Figura 14.20: Comando Addf

A continuación necesitamos conectar ddt a algo. Pero ¿cómo sabemos qué pines están disponibles?. La respuesta es mirar debajo de los pines. Ahí encontraremos ddt y se vera esto:

```
Component Pins:
Owner Type  Dir Value       Name
08    float R-  0.00000e+00 ddt.0.in
08    float -W  0.00000e+00 ddt.0.out
```

Parece fácil de entender, pero ¿qué señal o pin queremos conectar a ddt?. Podría ser un pin de eje, un pin de stepgen o una señal. Vemos esto cuando miramos joint.0:

```
Component Pins:
Owner Type  Dir Value       Name
03    float -W  0.00000e+00 joint.0.motor-pos-cmd ==> Xpos-cmd
```

Así que parece que Xpos-cmd debería ser una buena señal para usar. De vuelta al editor, donde ingresamos el siguiente comando:

```
linksp Xpos-cmd ddt.0.in
```

Ahora, si observamos la señal Xpos-cmd usando el nodo del árbol, veremos lo que hemos hecho

```
Signals:
Type Value Name
float 0.00000e+00 Xpos-cmd
<== joint.0.motor-pos-cmd
==> ddt.0.in
==> stepgen.0.position-cmd
```

Vemos que esta señal proviene de joint.0.motor-pos-cmd y va a ddt.0.in y stepgen.0.position-cmd. Al conectar nuestro bloque a la señal hemos evitado cualquier complicación con el flujo normal de este comando de movimiento.

El area *Show* utiliza halcmd para descubrir lo que está sucediendo en un HAL en ejecucion. Da información completa sobre lo que ha descubierto. También se actualiza a medida que se emiten los comandos en el pequeño panel de edición para modificar esa HAL. Hay momentos en que se quiere que se muestren un conjunto diferente de cosas sin toda la información disponible en este area. Ahí es donde el Área *WATCH* de HAL es de valor.

14.7.4. Pestaña WATCH

Al hacer clic en la pestaña WATCH aparece un cuadro en blanco. Puede añadir señales y pines a este cuadro y ver sus valores. [5] Usted puede agregar señales o pines cuando se muestra la pestaña WATCH haciendo clic en sus nombres. La siguiente figura muestra este cuadro con varias señales de tipo "bit". Estas señales incluyen habilitación de salida (enable-out) para los primeros tres ejes y dos de las tres señales de "estop" de iocontrol. Observe que los ejes no están habilitados aunque las señales de parada indican que LinuxCNC no está en parada. Una mirada rápida a la gui de usuario muestra que la condición de LinuxCNC es ESTOP RESET (para TkLinuxCNC) u OFF (en Axis, esquina inferior izquierda). La habilitacion del amplificador no pasara a *verdadero* hasta que la máquina se haya encendido.

[5] La frecuencia de actualización de la pantalla es mucho más baja que Halmeter o Halscope. Si necesita buena resolucion del tiempo de las señales, esas herramientas son mucho más efectivas.

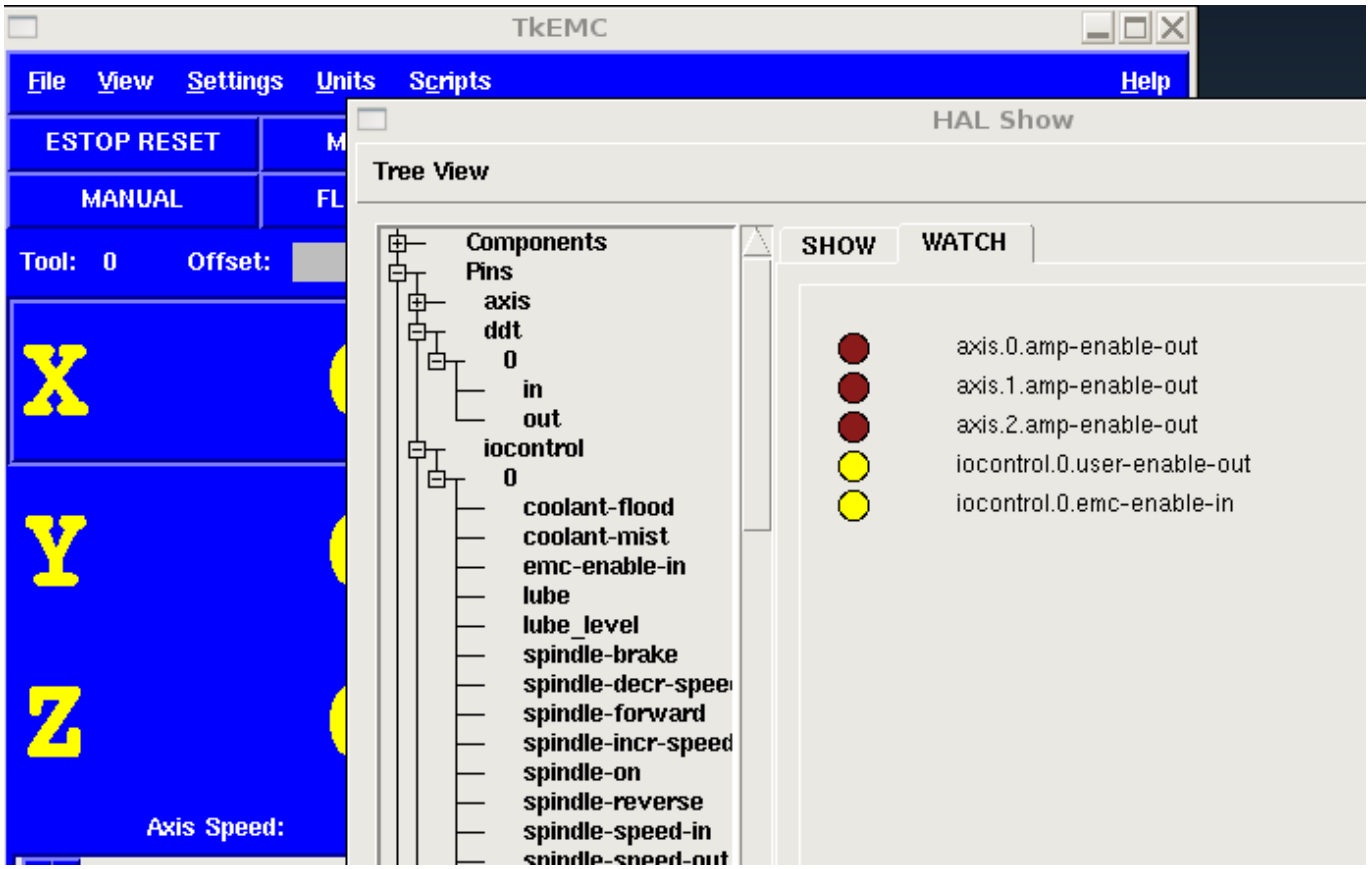


Figura 14.21: Watch Tab

WATCH muestra valores de tipo de bit (binarios) utilizando círculos de colores representando leds. Se muestran en color marrón oscuro cuando la señal de bit o el pin son *Falso*, y amarillo claro si es verdadero. Si selecciona un pin o señal que no es una señal de tipo bit (binario), WATCH mostrará un valor numérico.

WATCH permite probar rápidamente interruptores o ver el efecto de los cambios que realice en LinuxCNC mientras utiliza la interfaz gráfica. La frecuencia de actualización de WATCH es un poco lenta para ver los pulsos de paso, pero puede usarlo para eso si mueve un eje muy lentamente o en muy pequeños incrementos de distancia. Si ha usado IO_Show en LinuxCNC, la página de visualización en halshow puede configurarse para ver un parport como lo hizo con IO_Show.

14.8. Componentes HAL

14.8.1. Comandos y componentes de espacio de usuario

Todos los comandos en la siguiente lista tienen páginas de manual. Algunos tendrán descripciones ampliadas y otros, limitadas. Con estas listas se sabe qué componentes existen. Usar *man n nombre* para obtener información adicional. Para ver la información en la página de manual, por ejemplo de Axis, escriba en una ventana de terminal:

```
man axis (o tal vez 'man 1 axis' si su sistema lo requiere)
```

axis
: Interfaz gráfica de usuario AXIS para LinuxCNC (The Enhanced Machine Controller)

axis-remote
: Interface AXIS Remote.

comp
Construye, compila e instala componentes HAL de LinuxCNC.

gladevcp
Panel de control virtual para LinuxCNC basado en widgets Glade, Gtk y HAL.

gs2
Componente HAL de espacio de usuario para VDF Automation Direct GS2.

halcmd
Manipula HAL del controlador de máquina mejorado desde la línea de comandos.

hal_input
Controlar los pines HAL con cualquier dispositivo de entrada de Linux, incluidos los dispositivos USB HID.

halmeter
Observar pines, señales y parámetros de HAL.

halrun
Manipular HAL del controlador de máquina mejorado desde la línea de comandos.

halsampler
Muestreo de datos de HAL en tiempo real.

halstreamer
Transmitir archivos de datos a HAL en tiempo real.

halui
Observar los pines HAL y dar ordenes a LinuxCNC a través de NML.

io
Acepta comandos de E/S NML e interactúa con HAL en el espacio de usuario.

iocontrol
Acepta comandos de E/S NML e interactúa con HAL en el espacio de usuario.

linuxcnc
LinuxCNC (The Enhanced Machine Controller).

pyvcp
Panel de control virtual para LinuxCNC.

shuttle
controla los pines HAL con los dispositivos ShuttleXpress y ShuttlePRO fabricados por Contour Design.

14.8.2. Lista de componentes en tiempo real

Todos los comandos en la siguiente lista tienen páginas de manual. Algunos tendrán descripciones ampliadas y otros, limitadas. Con estas dos listas se sabe qué componentes existen. Usar *man n name* para obtener información adicional.

---

nota
Si el componente requiere un hilo de punto flotante, suele ser el hilo servo, más lento.

14.8.2.1. Componentes principales de LinuxCNC

motion
Acepta comandos de movimiento NML e interactúa con HAL en tiempo real.

axis
Acepta comandos de movimiento NML e interactúa con HAL en tiempo real.

classicladder
Software PLC en tiempo real basado en lógica de escalera. Consulte el capítulo ClassicLadder para obtener más información.

gladevcp
Muestra paneles de control virtuales construidos con GTK/Glade.

threads
Crea hilos HAL en tiempo real.

14.8.2.2. Componentes lógicos y enfocados a bits (bitwise)

and2
Puerta AND de dos entradas. Para devolver TRUE, ambas entradas deben ser ciertas.

not
Inversor.

or2
Puerta OR de dos entradas.

xor2
Puerta XOR (OR exclusivo) de dos entradas.

debounce
Filtra ruido en las entradas digitales.

edge
Detector de flanco.

flipflop
flip-flop tipo D.

oneshot
Generador de disparos de un pulso.

logic
Componente de función lógica general.

lut5
Función lógica de 5 entradas basada en tabla de consulta (look-up table).

match8
Detector de coincidencia binaria de 8 bits.

select8
Detector de coincidencia binaria de 8 bits.

14.8.2.3. Componentes aritméticos y de punto flotante

abs
Calcula el valor absoluto y el signo de la señal de entrada.

blend
Realiza interpolación lineal entre dos valores.

comp
Comparador de dos entradas con histéresis.

constant
Use un parámetro para establecer el valor de un pin.

sum2
Suma de dos entradas (cada una con una ganancia) y un desplazamiento.

counter
Cuenta los pulsos de entrada (obsoleto). Utilice el componente encoder.

updown
Cuenta hacia arriba o hacia abajo, con límites opcionales y comportamiento envolvente.

ddt
Calcula la derivada de la función de entrada.

deadzone
Devuelve el centro si está dentro del umbral.

hypot
Calculadora de hipotenusa de tres entradas (distancia euclidiana).

mult2
Producto de dos entradas.

mux16
Selecciona uno de entre dieciséis valores de entrada.

mux2
Selecciona uno entre dos valores de entrada.

mux4
Selecciona uno de entre cuatro valores de entrada.

mux8
Selecciona uno de entre ocho valores de entrada.

near
Determina si dos valores son aproximadamente iguales.

offset
Agrega un desplazamiento a una entrada y lo resta del valor de realimentación.

integ
Integrador.

invert
Calcula el inverso de la señal de entrada.

wcomp
Comparador de ventana.

weighted_sum
Convierte un grupo de bits a un entero.

biquad
Filtro Biquad IIR

lowpass
filtro de paso bajo

limit1
Limita la señal de salida para que caiga entre mín y máx. [6]

limit2
Limita la señal de salida para que caiga entre min y max. Limita la velocidad de giro a menos de maxv por segundo. [7]

limit3
Limita la señal de salida para que caiga entre min y max. Limita su velocidad de giro a menos de maxv por segundo. Limita su segunda derivada a menos de MaxA por segundo al cuadrado. [8]

maj3
Calcula la mayor de 3 entradas.

scale
Aplica una escala y un desplazamiento a su entrada.

14.8.2.4. Conversion de tipos

conv_bit_s32
Convierte un valor bit a s32.

conv_bit_u32
Convierte un valor bit a u32.

conv_float_s32
Convierte un valor float a s32.

conv_float_u32
Convierte un valor float a u32.

conv_s32_bit
Convierte un valor s32 a bit.

conv_s32_float
Convierte un valor s32 a float.

conv_s32_u32
Convierte un valor s32 a u32.

conv_u32_bit
Convierte un valor u32 a bit.

conv_u32_float
Convierte un valor u32 a float.

conv_u32_s32
Convierte un valor u32 a s32.

[6] Cuando la entrada es una posición, esto significa que la *posición* está limitada.
[7] Cuando la entrada es una posición, esto significa que *posición* y *velocidad* están limitadas.
[8] Cuando la entrada es una posición, esto significa que la *posición*, *velocidad*, y *aceleración* están limitadas.

14.8.2.5. Controladores de hardware

hm2_7i43
Controlador HAL para tarjetas EPP Anything Mesa Electronics 7i43 EPP con HostMot2.

hm2_pci
controlador HAL para Mesa Electronics 5i20, 5i22, 5i23, 4i65, 4i68 o cualquier placa de E/S, con firmware HostMot2.

hostmot2
controlador HAL para el firmware Mesa Electronics HostMot2.

mesa_7i65
Soporte para la tarjeta servo de ocho ejes Mesa 7i65.

pluto_servo
Controlador de hardware y firmware para la FPGA de puerto paralelo Pluto-P, para utilizar con servos.

pluto_step
Controlador de hardware y firmware para la FPGA de puerto paralelo Pluto-P, para utilizar con steppers.

thc
Control de la altura de antorcha utilizando una tarjeta Mesa THC.

serport
Controlador de hardware para los bits de E/S digitales del puerto serie 8250 y 16550.

14.8.2.6. Cinemática

kins
Definiciones de cinemática para LinuxCNC.

gantrykins
Un módulo de cinemática que mapea un eje a múltiples articulaciones.

genhexkins
Da seis grados de libertad en posición y orientación (XYZABC). La ubicación de los motores se define en tiempo de compilación.

genserkins
Cinemática que puede modelar un manipulador general de eslabones en serie con hasta 6 articulaciones angulares.

maxkins
Cinemática para una fresadora de 5 ejes llamada *max* con cabezal de inclinación (eje B) y rotativo horizontal montado sobre la mesa (eje C). Proporciona movimiento UVW en el sistema de coordenadas rotado. El archivo fuente, maxkins.c, puede ser un punto de inicio útil para otros sistemas de 5 ejes.

tripodkins
Las articulaciones representan la distancia del punto controlado desde tres ubicaciones predefinidas (los motores), dando tres grados de libertad en posición (XYZ).

trivkins
Las fresadoras y tornos estándar utilizan el módulo de cinemática trivial. Hay una correspondencia 1:1 entre articulaciones y ejes.

pumakins
Cinemática para robots estilo PUMA.

rotatekins
Los ejes X e Y se giran 45 grados en comparación con las articulaciones 0 y 1.

scarakins
Kinematics para robots tipo SCARA.

14.8.2.7. Control del motor

at_pid
Controlador proporcional/integral/derivativo con ajuste automático.

pid
Controlador proporcional/integral/derivativo.

pwmgen
Software de generación PWM/PDM.

encoder
Conteo por software de señales de encoder en cuadratura.

stepgen
Generación de pulsos de pasos de software.

14.8.2.8. BLDC y control de motores trifásicos

bldc_hall3
Controlador de motor BLDC bipolar de 3 cables, de conmutación trapezoidal, que utiliza sensores Hall.

clarke2
Versión de dos entradas de la transformada de Clarke.

clarke3
Transformada Clarke (3 fases a cartesiana) .

clarkeinv
Transformada Clarke inversa.

14.8.2.9. Otros componentes

charge_pump
Crea una onda cuadrada para la entrada de *bomba de carga* de algunas placas controladoras. La *bomba de carga* debe agregarse a la función hilo base. Cuando está habilitada, la salida está activada durante un período y desactivada durante otro. Para calcular la frecuencia de la salida 1/(período de tiempo en segundos x 2) = hz. Por ejemplo, si tiene un período base de 100.000 ns, o 0,0001 segundos, la fórmula sería 1/(0,0001 x 2) = 5000 hz o 5 Khz.

encoder_ratio
Engranaje electrónico para sincronizar dos ejes.

estop_latch
ESTOP latch.

feedcomp
Multiplica la entrada por la relación de la velocidad actual a la velocidad de alimentación.

gearchange
Seleccion de uno de dos rangos de velocidad.

ilowpass
Si bien puede encontrar otras aplicaciones, este componente se escribió para crear un movimiento más suave con un MPG.

En una máquina con alta aceleración, un jog corto puede comportarse casi como una funcion paso. Al poner el componente ilowpass entre l a salida de cuentas del codificador MPG y la entrada de jog-count del eje, se puede suavizar.

Elija la escala de forma conservadora para que durante una sesión nunca sea más de aproximadamente 2e9/escala pulsos vistos en el MPG. Elija la ganancia de acuerdo al nivel de suavizado deseado. Divida los valores de axis.N.jog-scale por escala.

joyhandle
Establece movimientos de joypad no lineales, bandas muertas y escalas.

knob2float
Convierte los conteos (probablemente de un mando codificador) a un valor de punto flotante.

minmax
Realiza un seguimiento de los valores mínimo y máximo de la entrada a las salidas.

sample_hold
Muestreo y retencion.

sampler
Muestrea datos HAL en tiempo real.

siggen
Generador de señal.

sim_encoder
codificador de cuadratura simulado.

sphereprobe
Sondeo de una semiesfera.

steptest
Utilizado por Stepconf para permitir la prueba de los valores de aceleración y velocidad de un eje.

streamer
Transmite archivos de datos HAL en tiempo real.

supply
Establece los pines de salida con valores de parámetros (en desuso).

threadtest
Componente para probar el comportamiento del hilo.

time
Temporizador acumulado de tiempo de ejecución que cuenta HH:MM:SS de entrada *activa*.

timedelay
El equivalente a un relé con retardo de tiempo.

timedelta
Componente que mide el comportamiento del tiempo de programación de subprocesos.

toggle2nist
Botón alternante para logica nist.

toggle
Push-on, push-off de pulsadores momentáneos.

tristate_bit
Coloca una señal en un pin de E/S solo cuando esté habilitado, similar a un buffer triestado en electrónica.

tristate_float
Coloca una señal en un pin de E/S solo cuando esté habilitado, similar a un buffer triestado en electrónica.

watchdog
Monitorea de una a treinta y dos entradas para un *latido*.

14.8.3. Llamadas API HAL

hal_add_funct_to_thread.3hal
hal_bit_t.3hal
hal_create_thread.3hal
hal_del_funct_from_thread.3hal
hal_exit.3hal
hal_export_funct.3hal
hal_float_t.3hal
hal_get_lock.3hal
hal_init.3hal
hal_link.3hal
hal_malloc.3hal
hal_param_bit_new.3hal
hal_param_bit_newf.3hal
hal_param_float_new.3hal
hal_param_float_newf.3hal
hal_param_new.3hal
hal_param_s32_new.3hal
hal_param_s32_newf.3hal
hal_param_u32_new.3hal
hal_param_u32_newf.3hal
hal_parport.3hal
hal_pin_bit_new.3hal
hal_pin_bit_newf.3hal
hal_pin_float_new.3hal
hal_pin_float_newf.3hal
hal_pin_new.3hal
hal_pin_s32_new.3hal
hal_pin_s32_newf.3hal
hal_pin_u32_new.3hal
hal_pin_u32_newf.3hal
hal_ready.3hal
hal_s32_t.3hal
hal_set_constructor.3hal
hal_set_lock.3hal
hal_signal_delete.3hal
hal_signal_new.3hal
hal_start_threads.3hal
hal_type_t.3hal
hal_u32_t.3hal
hal_unlink.3hal
intro.3hal
indocument.3hal

14.8.4. Llamadas RTAPI

EXPORT_FUNCTION.3rtapi
MODULE_AUTHOR.3rtapi
MODULE_DESCRIPTION.3rtapi
MODULE_LICENSE.3rtapi
RTAPI_MP_ARRAY_INT.3rtapi
RTAPI_MP_ARRAY_LONG.3rtapi
RTAPI_MP_ARRAY_STRING.3rtapi
RTAPI_MP_INT.3rtapi

```
RTAPI_MP_LONG.3rtapi
RTAPI_MP_STRING.3rtapi
intro.3rtapi
rtapi_app_exit.3rtapi
rtapi_app_main.3rtapi
rtapi_clock_set_period.3rtapi
rtapi_delay.3rtapi
rtapi_delay_max.3rtapi
rtapi_exit.3rtapi
rtapi_get_clocks.3rtapi
rtapi_get_msg_level.3rtapi
rtapi_get_time.3rtapi
rtapi_inb.3rtapi
rtapi_init.3rtapi
rtapi_module_param.3rtapi
RTAPI_MP_ARRAY_INT.3rtapi
RTAPI_MP_ARRAY_LONG.3rtapi
RTAPI_MP_ARRAY_STRING.3rtapi
RTAPI_MP_INT.3rtapi
RTAPI_MP_LONG.3rtapi
RTAPI_MP_STRING.3rtapi
rtapi_mutex.3rtapi
rtapi_outb.3rtapi
rtapi_print.3rtap
rtapi_prio.3rtapi
rtapi_prio_highest.3rtapi
rtapi_prio_lowest.3rtapi
rtapi_prio_next_higher.3rtapi
rtapi_prio_next_lower.3rtapi
rtapi_region.3rtapi
rtapi_release_region.3rtapi
rtapi_request_region.3rtapi
rtapi_set_msg_level.3rtapi
rtapi_shmem.3rtapi
rtapi_shmem_delete.3rtapi
rtapi_shmem_getptr.3rtapi
rtapi_shmem_new.3rtapi
rtapi_snprintf.3rtapi
rtapi_task_delete.3rtpi
rtapi_task_new.3rtapi
rtapi_task_pause.3rtapi
rtapi_task_resume.3rtapi
rtapi_task_start.3rtapi
rtapi_task_wait.3rtapi
```

14.9. Descripciones de Componentes HAL

14.9.1. Stepgen

Este componente proporciona generación de pulsos de paso, basada en software, en respuesta a comandos de posición o velocidad. En el modo posición tiene un bucle de posición preajustado, por lo que no se requiere sintonización PID. En modo velocidad, maneja un motor a la velocidad ordenada, manteniendo los límites de velocidad y aceleración. Es un componente en tiempo real solamente, y dependiendo de la velocidad de la CPU, etc, es capaz de velocidades de paso máximas de entre 10 kHz a 50kHz. El diagrama de bloques del generador de pulsos de pasos muestra un diagrama con tres bloques; cada uno es un generador de

pulsos de paso. El primer diagrama es para pasos tipo *0*, (paso y dirección). El segundo es para el tipo de paso *1* (arriba/abajo, o pseudo-PWM), y el tercero es para los tipos de paso 2 a 14 (varios patrones de pasos). Los dos primeros diagramas muestran el modo de control de posición y el tercero muestra el modo de velocidad. El modo de control y tipo de paso se configuran de forma independiente y se puede seleccionar cualquier combinación.

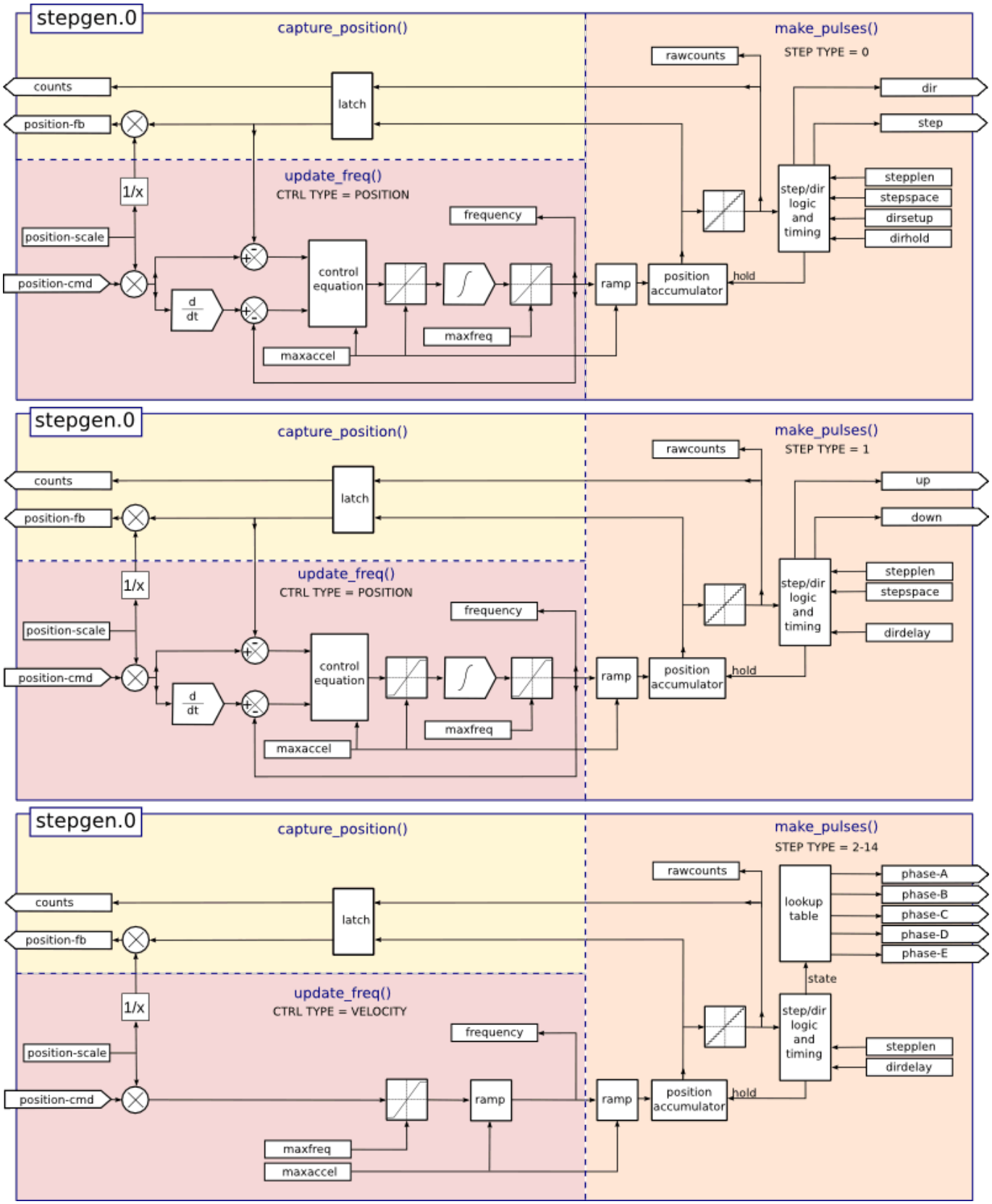


Figura 14.22: Diagramas de bloques del generador de pulsos, modo posición.

Instalación

```
halcmd: loadrt stepgen step_type=<type-array> [ctrl_type=<ctrl_array>]
```

<type-array> es una serie de enteros decimales separados por comas. Cada número carga un generador de pulsos de paso; el valor del número determina el tipo de stepping. *ctrl_type* es opcional; si se omite, todos los generadores de paso serán modo posición. *<ctrl_array>* es una serie de caracteres *p* o *v*, separados por comas, para especificar modo posición o modo velocidad.

Por ejemplo:

```
halcmd: loadrt stepgen step_type=0,0,2 ctrl_type=p,p,v
```

instalará tres generadores de pasos. Los dos primeros usan el tipo de paso *0* (paso y dirección) y se ejecutan en modo posición. El tercero usa el tipo de paso *2* (cuadratura) y se ejecuta en modo velocidad. El valor predeterminado para *<type-array>* es *0,0,0* que instalará tres generadores de tipo *0* (paso/dir). El maximo número de generadores de pasos es 8 (definido en MAX_CHAN en stepgen.c). Cada generador es independiente, pero todos son actualizados por la(s) misma(s) función(es) al mismo tiempo. En las siguientes descripciones, *<chan>* es el número de un generador específico. El primer generador es el número 0.

Eliminacion

```
halcmd: unloadrt stepgen
```

Pines Cada generador de pulsos de paso tendrá varios de estos pines, dependiendo del tipo de paso y el tipo de control seleccionado:

- *(float) stepgen.<chan>.position-cmd* - Posición deseada del motor, en unidades de posición (para modo posición solamente).
- *(float) stepgen.<chan>.velocity-cmd* - Velocidad deseada del motor, en unidades de posición por segundo (para modo velocidad solamente).
- *(s32) stepgen.<chan>.counts* - Posición de retroalimentación en conteos, actualizado por *capture_position()*.
- *(float) stepgen.<chan>.position-fb* - Posición de retroalimentación en unidades de posición, actualizadas por *capture_position()*.
- *(bit) stepgen.<chan>.enable* - Habilita la salida de pasos. Cuando es falso, no se generan pasos
- *(bit) stepgen.<chan>.step* - Salida de pulso de pasos (pasos tipo 0 solamente).
- *(bit) stepgen.<chan>.dir* - Salida de pulso de dirección (pasos tipo 0 solamente).
- *(bit) stepgen.<chan>.up* - Salida UP pseudo-PWM (pasos tipo 1 solamente).
- *(bit) stepgen.<chan>.down* - Salida DOWN pseudo-PWM (paso tipo 1 solamente).
- *(bit) stepgen.<chan>.phase-A* - Salida de fase A (tipos de paso 2-14 solamente).
- *(bit) stepgen.<chan>.phase-B* - Salida de fase B (tipos de paso 2-14 solamente).
- *(bit) stepgen.<chan>.phase-C* - Salida de fase C (tipos de paso 3-14 solamente).
- *(bit) stepgen.<chan>.phase-D* - Salida de fase D (tipos de paso 5-14 solamente).
- *(bit) stepgen.<chan>.phase-E* - Salida de fase E (tipos de pasos 11-14 solamente).

PARAMETROS

- *(float) stepgen.<chan>.position-scale* - Pasos por unidad de posición. Este parámetro se usa tanto por la salida como por la retroalimentación.
- *(float) stepgen.<chan>.maxvel* - Velocidad máxima, en unidades de posición por segundo. Si es 0.0, no tiene efecto.
- *(float) stepgen.<chan>.maxaccel* - Tasa máxima de aceleración/desaceleración, en unidades de posicion por segundo al cuadrado. Si es 0.0, no tiene efecto.
- *(float) stepgen.<chan>.frequency* - Frecuencia actual de pasos, en pasos por segundo.

- *(float) stepgen.<chan>.steplen* - Duración de un pulso de paso (pasos tipo 0 y 1) o tiempo mínimo en un estado dado (tipos de pasos 2-14), en nanosegundos.
- *(float) stepgen.<chan>.stepspace* - Espacio mínimo entre dos pulsos de paso (tipos de pasos 0 y 1 solamente), en nanosegundos. Establecer a 0 para habilitar la función stepgen *doublefreq*. Para usar *doublefreq* la funcion reset parport debe estar habilitada.
- *(float) stepgen.<chan>.dirsetup* - Tiempo mínimo desde un cambio de dirección hasta el comienzo del siguiente pulso de paso (pasos tipo 0 solamente), en nanosegundos.
- *(float) stepgen.<chan>.dirhold* - Tiempo mínimo desde el final de un pulso de paso hasta un cambio de dirección (pasos tipo 0 solamente), en nanosegundos.
- *(float) stepgen.<chan>.dirdelay* - Minimo tiempo entre cualquier paso a un paso en la dirección opuesta (paso tipos 1-14 solamente), en nanosegundos.
- *(s32) stepgen.<chan>.rawcounts* - conteo de retroalimentacion sin procesar, actualizado por *make_pulses()*.

En el modo posición, los valores de *maxvel* y *maxaccel* son utilizados por el bucle de posición interno para evitar la generación de trenes de pulso de paso que el motor no pueda seguir. Cuando se establecen valores apropiados para el motor, incluso un gran cambio instantáneo en la posición ordenada da como resultado un movimiento trapezoidal suave hacia la nueva ubicación. El algoritmo funciona midiendo el error de posición y el error de velocidad, y calculando una aceleración que intentara reducir ambos a cero al mismo tiempo. Para más detalles, incluido el contenido del cuadro *ecuación de control*, consulte el código.

En el modo velocidad, *maxvel* es un límite simple que se aplica a la velocidad ordenada, y *maxaccel* se usa para aumentar la frecuencia real si la velocidad ordenada cambia bruscamente. Como en el modo de posición, unos valores apropiados para estos parámetros aseguran que el motor pueda seguir el tren de pulsos generado

Pasos tipo 0 El tipo de pasos 0 es el tipo *paso y dirección* estándar. Cuando se configura para pasos tipo 0, hay cuatro parámetros adicionales que determinan el tiempo exacto de las señales de paso y dirección. En la siguiente figura se muestra el significado de estos parámetros. Los parámetros están en nanosegundos, pero se redondearán a un número entero múltiplo del período del hilo que llama a la funcion de stepgen *make_pulses()*. Por ejemplo, si se llama a *make_pulses()* cada 16 us, y la longitud Steplen es 20000 ns, entonces los pulsos de paso son 2us x 16us = 32us de largo. El valor predeterminado para los cuatro parámetros es 1 ns, pero el redondeo automático tiene efecto la primera vez que el código se ejecuta. Como un paso requiere *steplen* ns en alto y *stepspace* ns en bajo, la frecuencia máxima es 1.000.000.000 dividido por *(steplen + stepspace)*. Si *maxfreq* se establece más alto que ese límite, se bajara automáticamente, y si es cero, permanecerá en cero, pero la frecuencia de salida podra ser aún limitada

Al usar el controlador de puerto paralelo, la frecuencia de pasos se puede duplicar usando la función parport reset junto con la configuración *doublefreq* de stepgen.

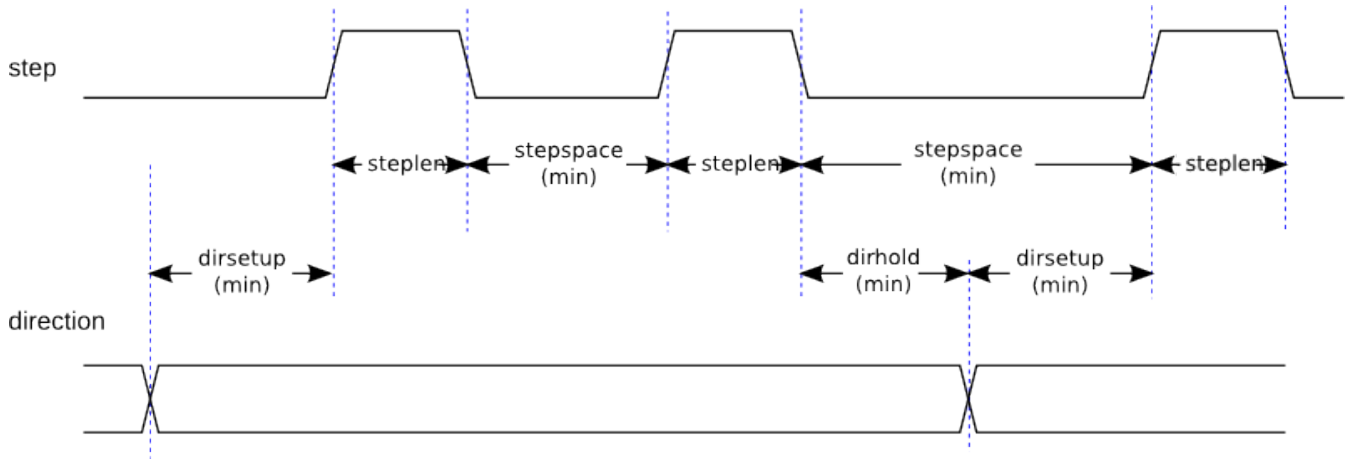


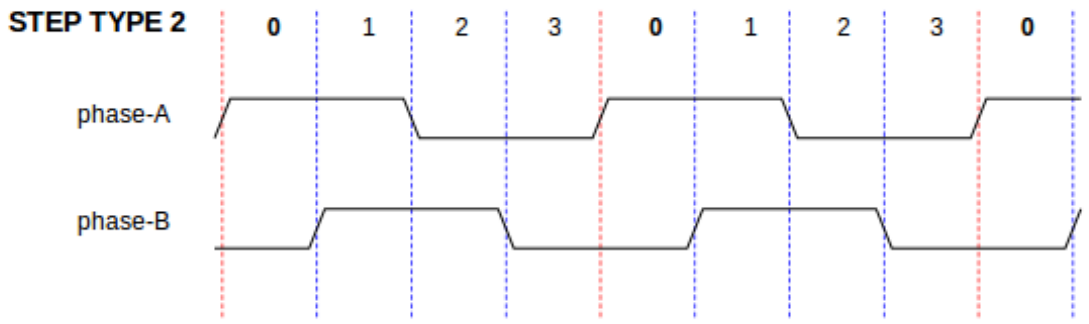
Figura 14.23: Temporizado de Paso y Dirección

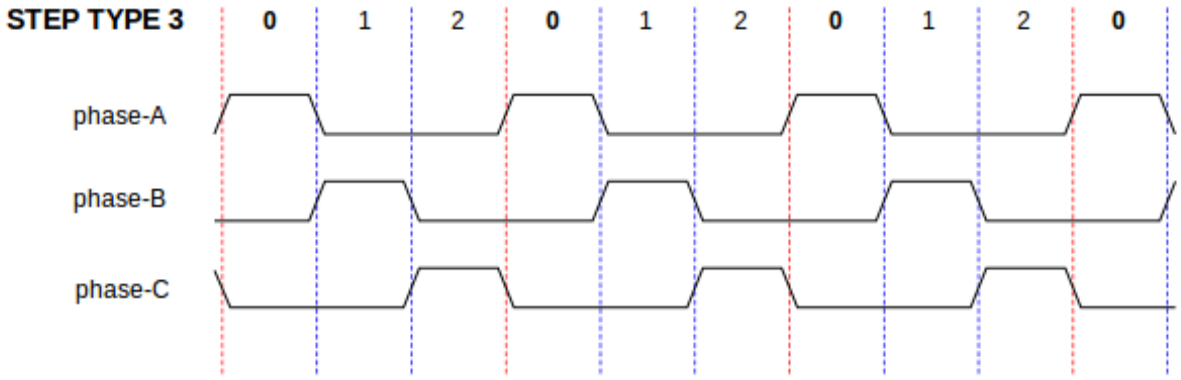
Pasos Tipo 1 El tipo de pasos 1 tiene dos salidas, *subir* y *bajar*. Los pulsos aparecen en una u otra, dependiendo de la dirección del movimiento deseada. Cada pulso es *steplen* ns de largo, y los pulsos están separados por al menos *stepspace* ns. La frecuencia máxima es la misma que para el tipo de pasos 0. Si se establece *maxfreq* más alto que ese límite, dicha frecuencia se reducirá. Si *maxfreq* es cero, seguirá siendo cero, pero la frecuencia de salida seguirá siendo limitada.

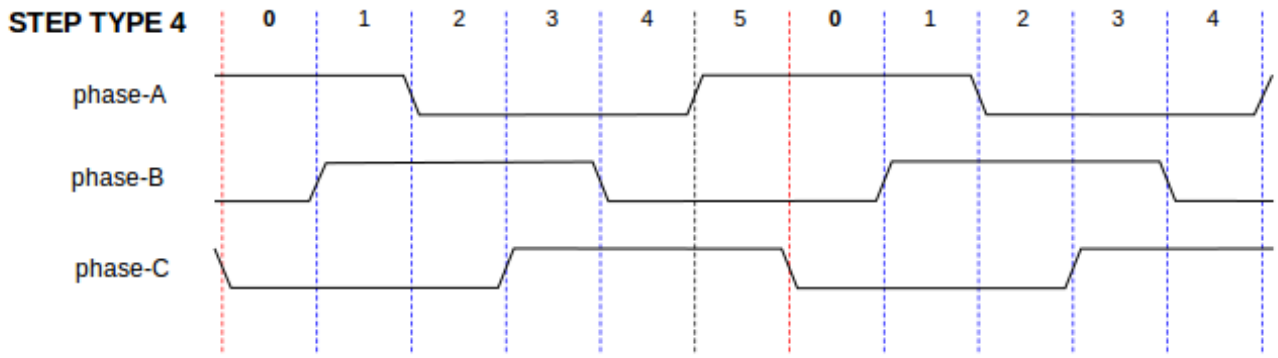
aviso
No use la función *reset parport* con los tipos de paso 2 - 14. Pueden producirse resultados inesperados.

Paso tipo 2 - 14 Los tipos de pasos 2 a 14 están basados en estados y tienen de dos a cinco salidas. En cada paso, un contador de estado se incrementa o disminuye. Se muestran patrones de salida de dos-tres fases, cuatro fases y cinco fases como una función del contador de estado. La frecuencia máxima es 1.000.000.000 divididos por *steplen*, y como en los otros modos, *maxfreq* se reducirá si está por encima del límite.

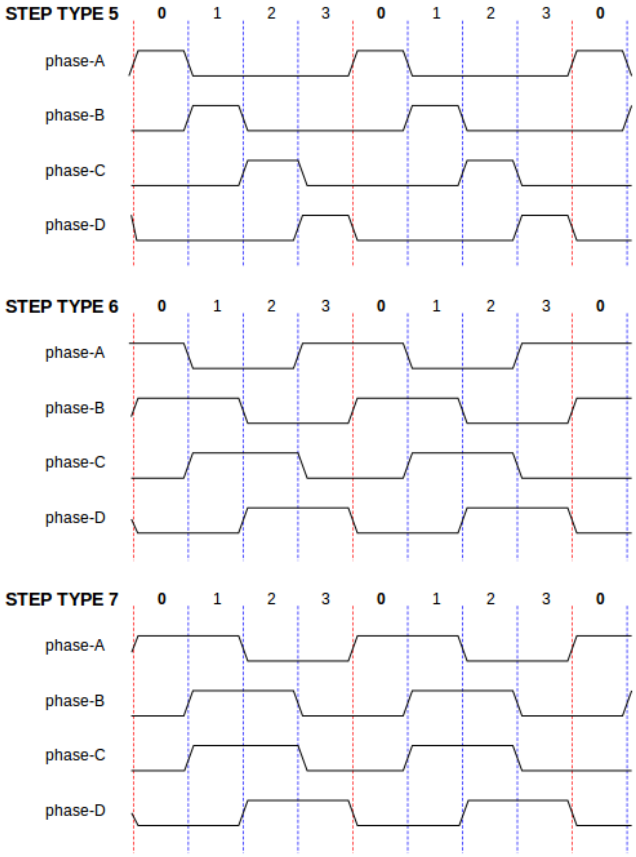
Tipos de pasos de dos y tres fases

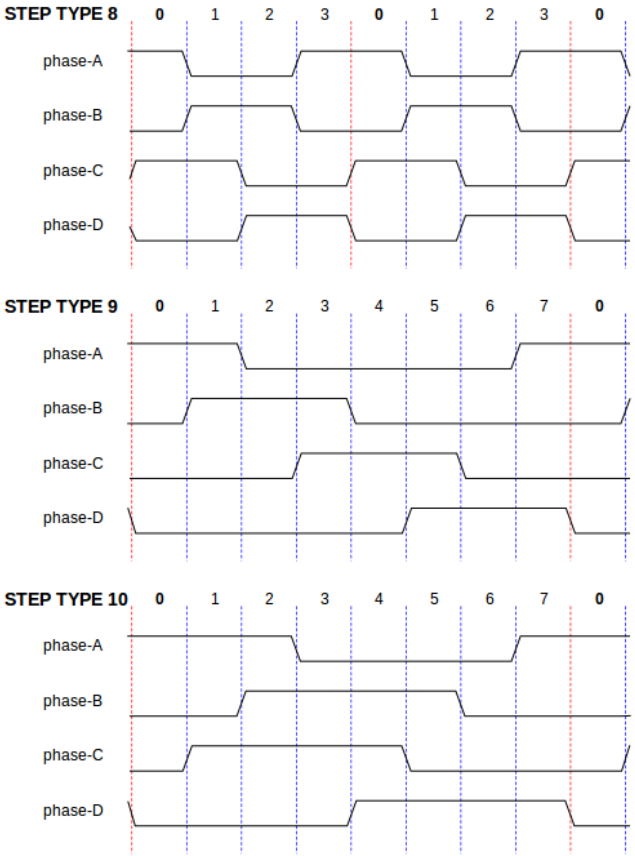




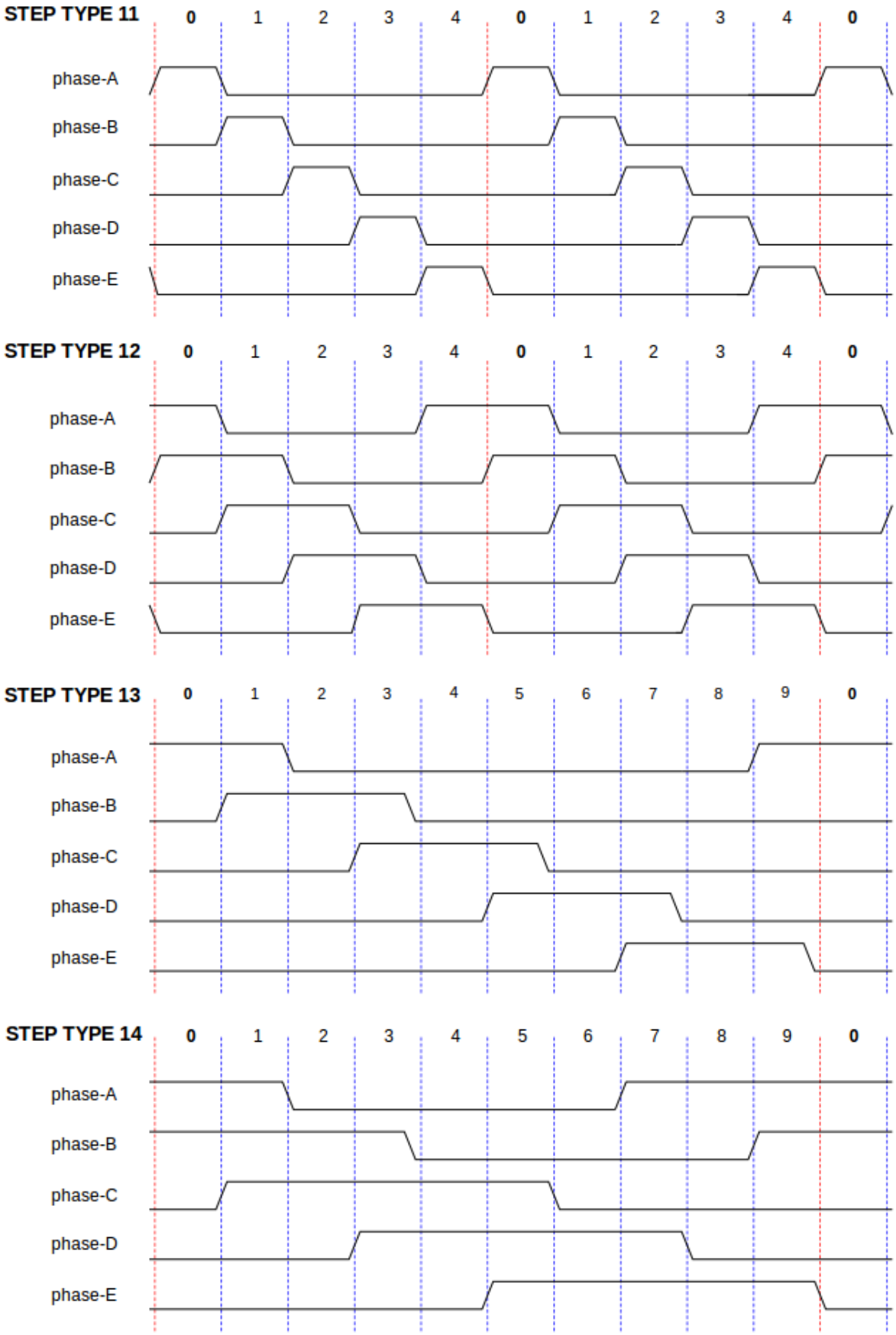


Tipos de pasos de cuatro fases





Fases de paso de cinco fases


STEP TYPE 11
0 1 2 3 4 0 1 2 3 4 0
phase-A
phase-B
phase-C
phase-D
phase-E
STEP TYPE 12
0 1 2 3 4 0 1 2 3 4 0
phase-A
phase-B
phase-C
phase-D
phase-E
STEP TYPE 13
0 1 2 3 4 5 6 7 8 9 0
phase-A
phase-B
phase-C
phase-D
phase-E
STEP TYPE 14
0 1 2 3 4 5 6 7 8 9 0
phase-A
phase-B
phase-C
phase-D
phase-E

Funciones El componente exporta tres funciones. Cada función actúa en todos los generadores de impulsos de pasos. No esta soportado ejecutar diferentes generadores en diferentes hilos.

- *(funct) stepgen.make-pulses* - Función de alta velocidad para generar y contar pulsos (sin punto flotante).
- *(funct) stepgen.update-freq* - Función de baja velocidad para conversión de posición a velocidad, escala y limitación.
- *(funct) stepgen.capture-position* - Función de baja velocidad para retroalimentación, actualizaciones de latches y escalado de posición.

La función de alta velocidad *stepgen.make-pulses* debe ejecutarse en un hilo rápido, de 10 a 50 us dependiendo de las capacidades de la computadora. El período de ese hilo determina la frecuencia de paso máxima, ya que *steplen*, *stepspace*, *dirsetup*, *dirhold* y *dirdelay* se redondean a un múltiplo entero del periodo del hilo en nanosegundos. Las otras dos funciones se pueden llamar a una tasa mucho más baja.

14.9.2. PWMgen

Este componente proporciona generación de PWM (modulacion de ancho de pulso) basada en software y formas de onda PDM (Modulación de Densidad de Pulso). Es un componente en tiempo real solamente, y dependiendo de la velocidad de la CPU, etc, es capaz de frecuencias PWM de unos pocos cientos de hercios con una resolución bastante buena, a quizás 10KHz con resolución limitada.

Instalación

```
loadrt pwmgen output_type=<config-array>
```

<config-array> es una serie de enteros decimales separados por comas. Cada número hace que se cargue un único generador PWM; el valor del número determina el tipo de salida. El siguiente ejemplo instalará tres generadores PWM. No hay un valor predeterminado; si no se especifica *<config-array>*, no se instalarán generadores PWM. La cantidad máxima de generadores de frecuencia es 8 (definido en MAX_CHAN en pwmgen.c). Cada generador es independiente, pero todos son actualizados por la(s) misma(s) función(es) al mismo tiempo. En las siguientes descripciones, *<chan>* es el número de un generador específico. El primer generador es el número 0.

Ejemplo

```
loadrt pwmgen output_type=0,1,2
```

Eliminacion

```
unloadrt pwmgen
```

Tipos de salida El generador PWM admite tres diferentes *tipos de salida*.

- *Tipo de salida 0* - un solo pin de salida PWM. Solo se aceptan comandos positivos; los valores negativos se tratan como cero (y se verán afectados por el parámetro *min-dc* si no son cero).
- *Tipo de salida 1* - pines PWM/PDM y de dirección. Entradas positivas y negativas saldran como PWM positivo y negativo. El pin de dirección es FALSO para comandos positivos, y VERDADERO para comandos negativos. Si su control necesita PWM positivo para CW y CCW, use el componente abs para convertir su señal PWM a un valor positivo cuando se ingrese una entrada negativa.
- *Tipo de salida 2* - Pines UP y DOWN. Para comandos positivos, la señal PWM aparece en la salida up, y la salida down permanece en FALSO. Para comandos negativos , la señal PWM aparece en la salida down, y la salida up sera FALSO. El tipo de salida 2 es el adecuado para conducir la mayoría de puentes H.

Pines Cada generador de PWM tendrá los siguientes pines:

- *(float) pwmgen.<chan>.value* - Valor de comando, en unidades arbitrarias. Será escalado por el parámetro *scale* (ver a continuación).

- *(bit) pwmgen.<chan>.enable* - Activa o desactiva las salidas del generador PWM.

Cada generador PWM también tendrá algunos de estos pines, dependiendo del tipo de salida seleccionado:

- *(bit) pwmgen.<chan>.pwm* - Salida PWM (o PDM), (tipos de salida 0 y 1 solo).
- *(bit) pwmgen.<chan>.dir* - Salida de dirección (salida tipo 1 solamente).
- *(bit) pwmgen.<chan>.up* - Salida PWM/PDM para un valor de entrada positivo (salida tipo 2 solamente).
- *(bit) pwmgen.<chan>.down* - Salida PWM/PDM para un valor de entrada negativa (salida tipo 2 solamente).

PARÁMETROS

- *(float) pwmgen.<chan>.scale* - Factor de escala para convertir *value* desde unidades arbitrarias hasta valores de ciclo de trabajo. Por ejemplo, si la escala está configurada en 4000 y el valor de entrada pasado a pwmgen.<chan>.value es 4000, se tomara el 100% duty-cycle (siempre activado). Si el valor es 2000, entonces será un 50%
- *(float) pwmgen.<chan>.pwm-freq* - Frecuencia PWM deseada, en Hz. Si es 0.0, genera PDM en lugar de PWM. Si se establece más alto que los límites internos, la próxima llamada de *update_freq()* lo establecerá en el límite interno. Si no es cero, y *dither* es falso, la próxima llamada de *update_freq()* lo configurará en el múltiplo entero más cercano del período de la función *make_pulses ()*.
- *(bit) pwmgen.<chan>.dither-pwm* - Si es verdadero, permite el tramado para alcanzar una frecuencia promedio PWM o ciclos de trabajo que no se pueden obtener con PWM puro. Si es falso, tanto la frecuencia PWM y el ciclo de trabajo se redondearán a valores que pueden ser logrados exactamente.
- *(float) pwmgen.<chan>.min-dc* - Ciclo de trabajo mínimo, entre 0.0 y 1.0 (ciclo de trabajo irá a cero cuando está deshabilitado, independientemente de esta configuración).
- *(float) pwmgen.<chan>.max-dc* - Ciclo de trabajo máximo, entre 0.0 y 1.0.
- *(float) pwmgen.<chan>.curr-dc* - Ciclo de trabajo actual - después de toda limitación y redondeo (solo lectura).

Funciones El componente exporta dos funciones. Cada función actúa en todos los generadores PWM. No esta soportado ejecutar diferentes generadores en diferentes hilos

- *(funct) pwmgen.make-pulses* - Función de alta velocidad para generar formas de onda PWM (sin punto flotante). La función de alta velocidad *pwmgen.make-pulses* debe ejecutarse en el hilo base (más rápido), de 10 a 50 us dependiendo de la capacidades de la computadora. El período de ese hilo determina la máxima frecuencia de la portadora PWM, así como la resolución de las señales PWM o PDM. Si el hilo base es 50000 ns, entonces cada 50us el módulo decide si es el momento de cambiar el estado de la salida. Con un ciclo de trabajo del 50% y frecuencia PWM de 25Hz esto significa que la salida cambia de estado cada (1/25)segundos/50uS*50% = 400 iteraciones. Esto también significa que tiene 800 valores posible de ciclo de trabajo (sin dithering)
- *(funct) pwmgen.update* - Función de baja velocidad para escalar y limitar el valor y manejar otros parámetros Esta es la función del módulo que contiene alguna matemática más complicada para calcular en cuántos periodos base el resultado debe ser alto, y en cuántos debe ser bajo.

14.9.3. Encoder

Este componente proporciona un conteo, basado en software, de señales de encoders de cuadratura. Es un componente en tiempo real, y dependiendo de la velocidad de la CPU, la latencia, etc., es capaz de obtener tasas de conteo máximas desde 10 kHz hasta quizás unos 50kHz.

El hilo base debe tener una velocidad doble a la de de conteo para permitir ruido y variaciónes de timing. Por ejemplo, si tiene un codificador de 100 impulsos por revolución en el husillo y sus RPM máximas es 3000, el hilo base máximo debe ser de 25 us.

Un encoder de 100 impulsos por revolución tendrá 400 conteos. La velocidad del eje de 3000 RPM = 50 RPS (revoluciones por segundo). 400*50 = 20,000 cuentas por segundo o 50 us entre cuentas.

El diagrama de bloques del encoder contador es un diagrama de un canal de encoder.

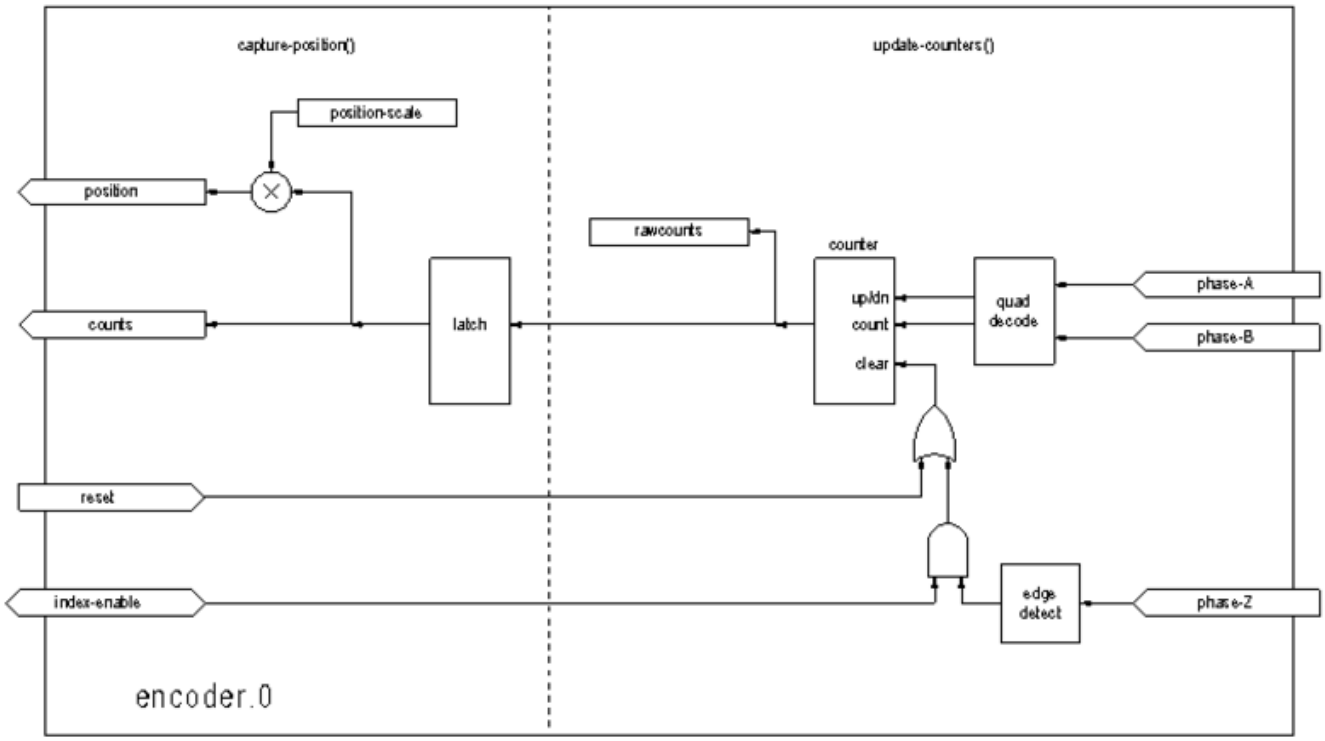


Figura 14.24: Diagrama de bloque de contador codificador

Instalación

```
halcmd: loadrt encoder [num_chan=<counters>]
```

<counters> es la cantidad de contadores de encoder que desea instalar. Si no se especifica *numchan*, serán instalados tres contadores. El maximo número de contadores es 8 (definido en MAX_CHAN en encoder.c). Cada contador es independiente, pero todos son actualizados por la(s) misma(s) función(es) al mismo tiempo. En las siguientes descripciones, *<chan>* es el número de un contador específico. El primer contador es el número 0.

Eliminacion

```
halcmd: unloadrt encoder
```

PINES

- *encoder.<chan>.counter-mode* (bit, I/O) (predeterminado: FALSE) - Habilita el modo contador. Cuando es VERDADERO, el contador cuenta cada flanco ascendente de la entrada de la fase A, ignorando el valor en la fase B. Esto es útil para contar la salida de un solo canal del sensor (sin cuadratura). Cuando es FALSO, cuenta en modo cuadratura.

- *encoder.<chan>.counts* (s32, salida) - Posición, en conteos del codificador.
- *encoder.<chan>.counts-latched* (s32, salida) - No se usa en este momento.
- *encoder.<chan>.index-enable* (bit, I/O) - Cuando es VERDADERO, *counts* y *position* se resetean a cero en el siguiente flanco ascendente de la Fase Z. Al mismo tiempo, *index-enable* se resetea a cero para indicar que el flanco ascendente ha aparecido. El pin es bidireccional. Si *index-enable* es FALSO, el canal de la fase Z del codificador será ignorado, y el contador contará normalmente. El controlador del codificador nunca pondra *index-enable* en VERDADERO, pero otros componentes puede hacerlo.
- *encoder.<chan>.latch-falling* (bit, entrada) (predeterminado: TRUE) - No utilizado en este momento.
- *encoder.<chan>.latch-input* (bit, entrada) (predeterminado: TRUE) - No utilizado en este momento.
- *encoder.<chan>.latch-rising* (bit, entrada) - No se usa en este momento.
- *encoder.<chan>.min-speed-estimate* (float, entrada) - Determina la magnitud de velocidad verdadera mínima a la cual la velocidad se estimará como distinta de cero, y *position-interpolated* es interpolada. Las unidades de *min-speed-estimate* son las mismas que unidades de *velocity*. El factor de escala, en cuentas por unidad de longitud. Ajustar este parámetro demasiado bajo hará que tome mucho tiempo el que la velocidad pase a 0 después de que los impulsos del encoder han dejado de llegar.
- *encoder.<chan>.phase-A* (bit, entrada) - Fase A de la señal del codificador en cuadratura.
- *encoder.<chan>.phase-B* (bit, entrada) - Fase B de la señal del codificador en cuadratura.
- *encoder.<chan>.phase-Z* (bit, entrada) - Fase Z (pulso de índice) de la señal del codificador en cuadratura.
- *encoder.<chan>.position* (float, salida) - Posición en unidades escaladas (ver *position-scale*).
- *encoder.<chan>.position-interpolated* (float, salida) - Posición en unidades escaladas, interpoladas entre cuenta del codificador. La *posición interpolada* intenta interpolar entre cuentas del codificador, basandose en la mayor velocidad medida reciente. Solo válido cuando la velocidad es aproximadamente constante y superior a *min-speed-estimate*. No lo use para control de posición, ya que su valor es incorrecto a velocidades bajas, durante las inversiones de dirección y durante los cambios de velocidad. Sin embargo, permite usar un codificador de pocos impulsos (incluido un impulso por revolución) para roscado en el torno, y puede tener otros usos también.
- *encoder.<chan>.position-latched (float, salida)* - No utilizado en este momento.
- *encoder.<chan>.position-scale (float, I/O)* - Factor de escala, en cuentas por unidad de longitud. Por ejemplo, si la escala de posición es 500, entonces 1000 conteos del codificador se reportarán como una posición de 2.0 unidades.
- *encoder.<chan>.rawcounts (s32, entrada)* - Conteo sin procesar, determinado por la funcion *update-counters*. Este valor es actualizado con más frecuencia que las cuentas y la posición. Tampoco se ve afectado por un reinicio o pulso de índice.
- *encoder.<chan>.reset* (bit, entrada) - Cuando es VERDADERO, fuerza *count* y *position* a cero de inmediato.
- *encoder.<chan>.velocity* (float, salida) - Velocidad en unidades escaladas por segundo. Se utiliza un algoritmo que reduce en gran medida la cuantificación de ruido comparado con una simple diferenciacion de la salida de *position*. Cuando la magnitud de la verdadera velocidad está por debajo de la estimación de velocidad mínima, la salida de velocidad es 0.
- *encoder.<chan>.x4-mode (bit, I/O) (predeterminado: TRUE)* - Habilita el modo 4x. Cuando es VERDADERO, el contador cuenta cada borde de la forma de onda en cuadratura (cuatro cuentas por ciclo completo). Cuando es falso, solo cuenta una vez por ciclo completo. Cuando es FALSO, este parámetro es ignorado. El modo 1x es útil para algunos jogwheels.

PARÁMETROS

- *encoder.<chan>.capture-position.time (s32, RO)*
- *encoder.<chan>.capture-position.tmax (s32, RW)*
- *encoder.<chan>.update-counters.time (s32, RO)*
- *encoder.<chan>.update-counter.tmax (s32, RW)*

Funciones El componente exporta dos funciones. Cada función actúa en todos los contadores de encoder - no esta soportado ejecutar diferentes contadores en diferentes hilos.

- *(funct) encoder.update-counters* - Función de alta velocidad para contar pulsos (sin punto flotante).
- *(funct) encoder.capture-position* - Función de baja velocidad para actualizar latches y escala de posición.

14.9.4. PID

Este componente proporciona control Proporcional/Integral/Derivativo en bucles. Es un componente en tiempo real solamente. Para simplificar, esta discusión asume que estamos hablando de bucles de posición, sin embargo este componente se puede utilizar para implementar otros ciclos de retroalimentación como velocidad, altura de una antorcha, temperatura, etc. El diagrama de bloque de lazo PID es un diagrama de bloques de un solo bucle PID.

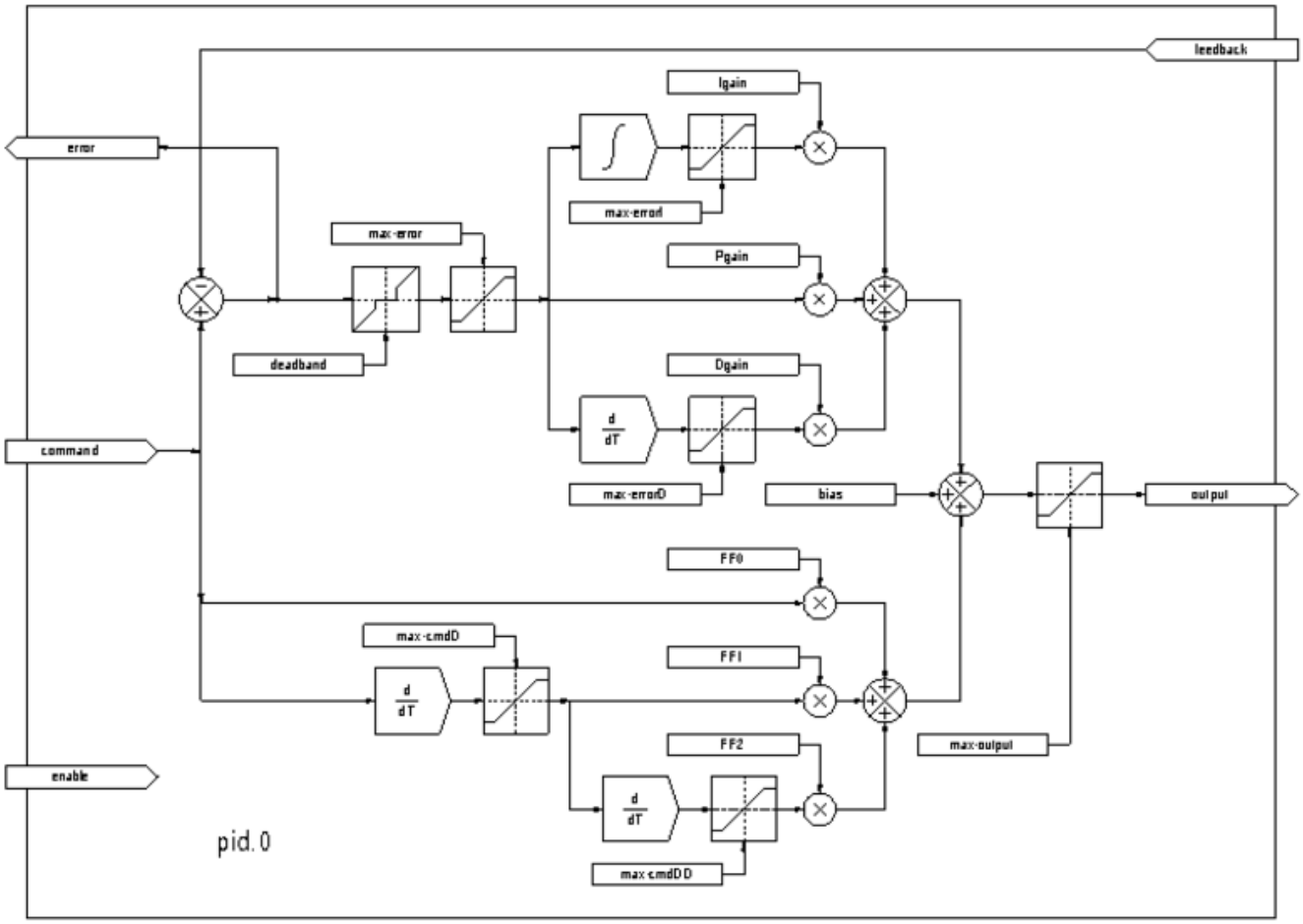


Figura 14.25: PID Diagrama de bloque de bucle

Instalación

```
halcmd: loadrt pid [num_chan=<loops>] [debug=1]
```

<loops> es la cantidad de bucles PID que desea instalar. Si *numchan* no se especifica, se instalará un solo bucle. El maximo número de bucles es 16 (definido por MAX_CHAN en pid.c). Cada bucle es completamente independiente. En las siguientes descripciones, *<loopnum>* es el numero de bucle de un bucle específico. El primer bucle es el número 0.

Si se especifica *debug=1*, el componente exportará algunos pines que pueden ser útiles durante la depuración y el ajuste. Por defecto, los pines extra no se exportan, para ahorrar espacio en la memoria compartida y evitar abarrotar la lista de pines.

Eliminacion

```
halcmd: unloadrt pid
```

Pines Los tres pines más importantes son

- *(float) pid.<loopnum>.command* - La posición deseada, que sera la comandada por otro componente del sistema.
- *(float) pid.<loopnum>.feedback* - La posición actual, que sera la medida por un dispositivo de retroalimentación tal como un encoder.
- *(float) pid.<loopnum>.output* - Comando de velocidad que intenta pasar de la posición actual a la posición deseada.

Para un bucle de posición, *command* y *feedback* están en unidades de posición. Para un eje lineal, esto podría ser pulgadas, mm, metros o lo que sea pertinente. Del mismo modo, para un eje angular, podría ser grados, radianes, etc. Las unidades del pin *output* representan el cambio necesario para hacer que la retroalimentación coincida con el comando. Como tal, para un bucle de posición, *output* es una velocidad, en pulgadas/seg, mm/seg, grados/seg, etc. Las unidades de tiempo son siempre en segundos, y las unidades de velocidad coinciden con las unidades de posición. Si el comando y la retroalimentación están en metros, la salida está en metros por segundo.

Cada bucle tiene dos pines que se utilizan para monitorizar o controlar la operación general del componente.

- *(float) pid.<loopnum>.error* - Igual a *.command* menos *.feedback*.
- *(bit) pid. <loopnum>.enable* - Un bit que habilita el bucle. Si *.enable* es falso, todos los integradores se resetean y la salida es obligada a cero. Si *.enable* es verdadero, el ciclo funciona normalmente.

Pines usados para informar la saturación. La saturación ocurre cuando la salida de el bloque PID está en su límite máximo o mínimo.

- *(bit) pid.<loopnum>.saturated* - Verdadero cuando la salida está saturada.
- *(float) pid.<loopnum>.saturated_s* - El tiempo que la salida ha sido saturada.
- *(s32) pid.<loopnum>.saturated_count* - La hora en que la salida ha sido saturada.

Las ganancias PID, los límites y otras características *ajustables* del ciclo estan disponible como pines para que puedan ajustarse dinámicamente para obtener más posibilidades de ajuste avanzadas.

- *(float) pid.<loopnum>.Pgain* - Ganancia proporcional
- *(float) pid.<loopnum>.Igain* - Ganancia integral
- *(float) pid.<loopnum>.Dgain* - Ganancia derivativa
- *(float) pid.<loopnum>.bias* - Offset constante en la salida
- *(float) pid.<loopnum>.FF0* - Zeroth Order feedforward - salida proporcional al comando (posición).
- *(float) pid.<loopnum>.FF1* - First order feedforward - salida proporcional a la derivada del comando (velocidad).
- *(float) pid.<loopnum>.FF2* - Second order feedforward - salida proporcional a la 2da derivada del comando (aceleración).
- *(float) pid.<loopnum>.deadband* - Cantidad de error que se ignorará

- *(float) pid.<loopnum>.maxerror* - Límite de error
- *(float) pid.<loopnum>.maxerrorI* - Limite en error integrador
- *(float) pid.<loopnum>.maxerrorD* - Límite de error derivativo
- *(float) pid.<loopnum>.maxcmdD* - Límite de derivada del comando
- *(float) pid.<loopnum>.maxcmdDD* - Límite de 2ª derivada del comando
- *(float) pid.<loopnum>.maxoutput* - Límite en el valor de salida

Si se especificó *debug=1* cuando el componente se instaló, se exportarán cuatro pines adicionales:

- *(float) pid.<loopnum>.errorI* - Integral de error.
- *(float) pid.<loopnum>.errorD* - Derivada del error.
- *(float) pid.<loopnum>.commandD* - Derivada del comando.
- *(float) pid.<loopnum>.commandDD* - 2da derivada del comando.

Funciones El componente exporta una función para cada lazo PID. Esta función realiza todos los cálculos necesarios para el ciclo. Como cada ciclo tiene su propia función, los bucles individuales se pueden incluir en diferentes hilos y ejecutar a diferentes velocidades.

- *(funct) pid.<loopnum>.do_pid_calcs* - Realiza todos los cálculos para un solo bucle PID.

Si quieres entender el algoritmo exacto utilizado para calcular la salida del bucle PID, consulte la figura Diagrama de bloques de bucle PID, los comentarios al comienzo de *emc2/src/hal/components/pid.c*, y por supuesto, el código en sí. Los cálculos del bucle están en la funcion C *calc_pid()*.

14.9.5. Encoder simulado

El encoder simulado es exactamente eso. Produce pulsos en cuadratura con un pulso de índice, a una velocidad controlada por un pin HAL. Principalmente útil para pruebas.

Instalación

```
halcmd: loadrt sim-encoder num_chan=<número>
```

<número> es la cantidad de codificadores que quiere simular. Si no se especifica, se instalará un solo encoder simulado. El número máximo es 8 (definido por MAX_CHAN en sim_encoder.c).

Eliminacion

```
halcmd: unloadrt sim-encoder
```

PINES

- *(float) sim-encoder.<chan-num>.speed* - El comando de velocidad para el eje simulado.
- *(bit) sim-encoder.<chan-num>.phase-A* - Salida de cuadratura.
- *(bit) sim-encoder.<chan-num>.phase-B* - Salida de cuadratura.
- *(bit) sim-encoder.<chan-num>.phase-Z* - Salida de impulsos de índice.

Cuando *.speed* es positivo, *.phase-A* precede a *.phase-B*.

PARÁMETROS

- *(u32) Sim-encoder.<chan-num>.ppr* - Impulsos por revolución.
- *(float) Sim-encoder.<chan-num>.scale* - Factor de escala para *speed*. El valor predeterminado es 1.0, lo que significa que *speed* está en revoluciones por segundo. Cambie a 60 para RPM, a 360 para grados por segundo, a 6.283185 para radianes por segundo, etc.

Tenga en cuenta que los pulsos por revolución no son lo mismo que los recuentos por revolución. Un pulso es un ciclo de cuadratura completo. En la mayoría de encoder, los contadores contarán cuatro veces durante un ciclo completo.

Funciones El componente exporta dos funciones. Cada función afecta a todos los encoder simulados

- *(funct) sim-encoder.make-pulses* - Función de alta velocidad para generar impulsos de cuadratura (sin punto flotante).
- *(funct) sim-encoder.update-speed* - Función de baja velocidad para leer *speed*, escalar y configurar *make-pulses*.

14.9.6. Debounce

Debounce es un componente en tiempo real que puede filtrar los rebotes creados por interruptores de contactos mecánicos. También puede ser útil en otras aplicaciones donde los pulsos cortos deben ser rechazados.

Instalación

```
halcmd: loadrt debounce cfg=<config-string>
```

<config-string> es una serie de enteros decimales separados por comas. Cada número instala un grupo de filtros antirrebote idénticos, el número determina cuántos filtros están en el grupo

Por ejemplo:

```
halcmd: loadrt debounce cfg=1,4,2
```

instalará tres grupos de filtros. El grupo 0 contiene un filtro, el grupo 1 contiene cuatro y el grupo 2 contiene dos filtros. El valor por defecto valor para *<config-string>* es "*1*" que instalará un solo grupo que contiene un solo filtro. El número máximo de grupos es 8 (definido por MAX_GROUPS en debounce.c). La cantidad máxima de filtros en un grupo está limitada solo por el espacio de memoria compartida. Cada grupo es completamente independiente. Todos los filtros en un solo grupo son idénticos, y todos son actualizados por la misma funcion al mismo tiempo. En las siguientes descripciones, *<G>* es el número de grupo y *<F>* es el número de filtro dentro del grupo. El primer filtro es el grupo 0, filtro 0.

Eliminacion

```
halcmd: unloadrt debounce
```

Pines Cada filtro individual tiene dos pines.

- *(bit) debounce.<G>.<F>.in* - Entrada del filtro *<F>* en el grupo *<G>*.
- *(bit) debounce.<G>.<F>.out* - Salida del filtro *<F>* en el grupo *<G>*.

Parámetros Cada grupo de filtros tiene un parámetro [9].

- *(s32) debounce.<G>.delay* - Retraso del filtro para todos los filtros del grupo *<G>*.

El retraso del filtro está en unidades de período de hilos. El retraso mínimo es cero. La salida de un filtro de retardo cero sigue exactamente su entrada: no filtra nada. A medida que aumenta "delay", son rechazados rebotes mas largos. Si *delay* es 4, todos los rebotes menores que o igual a cuatro períodos de hilo serán rechazados.

Funciones Cada grupo de filtros tiene una función que actualiza todos los filtros en ese grupo *simultáneamente*. Diferentes grupos de filtros pueden ser actualizado a partir de diferentes hilos en diferentes períodos.

- *(funct) debounce.<G>* - Actualiza todos los filtros en el grupo *<G>*.

[9] Cada filtro individual también tiene una variable de estado interna. Hay un switch en tiempo de compilacion que puede exportar esa variable como parámetro. Esta está destinado a pruebas, y simplemente desperdicia memoria compartida en condiciones normales.

14.9.7. Siggen

Siggen es un componente en tiempo real que genera ondas cuadradas, triángulares y sinusoidales. Se usa principalmente para pruebas.

Instalación

```
halcmd: loadrt siggen [num_chan=<chans>]
```

<chans> es la cantidad de generadores de señal que desea instalar. Si no se especifica *numchan*, se instalará un generador de señal. El maximo número de generadores es 16 (como se define en MAX_CHAN en siggen.c). Cada generador es completamente independiente. En las siguientes descripciones, *<chan>* es el número de un generador de señal específico (los números comenzar en 0).

Eliminacion

```
halcmd: unloadrt siggen
```

Pines Cada generador tiene cinco pines de salida.

- *(float) siggen.<chan>.sine* - Salida de onda sinusoidal.
- *(float) siggen.<chan>.cosine* - Salida de coseno.
- *(float) siggen.<chan>.sawtooth* - Salida de diente de sierra.
- *(float) siggen.<chan>.triangle* - Salida de onda triangular.
- *(float) siggen.<chan>.square* - Salida de onda cuadrada.

Las cinco salidas tienen la misma frecuencia, amplitud y offset.

Además de los pines de salida, hay tres pines de control:

- *(float) siggen.<chan>.frequency* - Establece la frecuencia en hercios, el valor predeterminado es 1 Hz.
- *(float) siggen.<chan>.amplitude* - Establece la amplitud máxima de formas de onda de salida, por defecto es 1.
- *(float) siggen.<chan>.offset* - Establece el desplazamiento DC de la salida de formas de onda, por defecto es 0.

Por ejemplo, si *siggen.0.amplitude* es 1.0 y *siggen.0.offset* es 0.0, las salidas oscilarán de -1.0 a /+1.0. Si *siggen.0.amplitude* es 2.5 y *siggen.0.offset* es 10.0, entonces las salidas oscilarán desde 7.5 a 12.5.

Parámetros Ninguno. [10]

FUNCIONES

- *(funct) siggen.<chan>.update* - Calcula nuevos valores para las cinco salidas.

14.9.8. lut5

El componente lut5 es un componente lógico de 5 entradas basado en una tabla de búsqueda.

- *lut5* no requiere un hilo de punto flotante.

Instalación y uso

[10] Antes de la versión 2.1, frecuencia, amplitud y desplazamiento fueron parámetros. Se cambiaron a pines para permitir el control por otros componentes

```
loadrt lut5 [count=N|names=name1[,name2...]]
addf lut5.N servo-thread | base-thread
setp lut5.N.function 0xN
```

Función de computacion Para calcular el número hexadecimal para la función comenzando desde la parte superior ponga un 1 o 0 para indicar si esa fila sería verdadera o falsa. A continuación, escriba cada número en la columna de salida comenzando desde arriba y escribiéndolos desde la derecha a la izquierda. Este será el número binario. Usando una calculadora, ingrese el número binario y luego conviértalo en hexadecimal; ese será el valor para la función.

Cuadro 14.1: Tabla de búsqueda

Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Output
0	0	0	0	0	
0	0	0	0	1	
0	0	0	1	0	
0	0	0	1	1	
0	0	1	0	0	
0	0	1	0	1	
0	0	1	1	0	
0	0	1	1	1	
0	1	0	0	0	
0	1	0	0	1	
0	1	0	1	0	
0	1	0	1	1	
0	1	1	0	0	
0	1	1	0	1	
0	1	1	1	0	
0	1	1	1	1	
1	0	0	0	0	
1	0	0	0	1	
1	0	0	1	0	
1	0	0	1	1	
1	0	1	0	0	
1	0	1	0	1	
1	0	1	1	0	
1	0	1	1	1	
1	1	0	0	0	
1	1	0	0	1	
1	1	0	1	0	
1	1	0	1	1	
1	1	1	0	0	
1	1	1	0	1	
1	1	1	1	0	
1	1	1	1	1	

Ejemplo de dos entradas En la siguiente tabla, hemos seleccionado el estado de salida para cada línea que deseamos sea verdad

Cuadro 14.2: Tabla de búsqueda

Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Output
0	0	0	0	0	0
0	0	0	0	1	1
0	0	0	1	0	0
0	0	0	1	1	1

Mirando la columna de salida de nuestro ejemplo queremos que la salida esté en ON cuando Bit 0 o (Bit 0 y Bit1) están activados y nada más. El número binario es *b1010* (gira la salida 90 grados CW). Ingrese este número en la calculadora luego cambie la pantalla a hexadecimal y el número necesario para la función es *0xa*. El prefijo hexadecimal es *0x*.

14.10. Ejemplos HAL

Todos estos ejemplos suponen que se comienza con una configuración basada en stepconf que tiene dos hilos *base-thread* y *servo-thread*. El asistente stepconf creará un archivo custom.hal vacío y un archivo custom_postgui.hal. El archivo custom.hal se cargará después del archivo HAL de configuración y el archivo custom_postgui.hal se cargan después de que la GUI ha sido cargada.

14.10.1. Conexión de dos salidas

Para conectar dos salidas a una entrada, puede usar el componente or2. or2 funciona así; si cualquiera de las entradas a or2 está activada, entonces la salida or2 está activada. Si ninguna entrada a or2 está activada, la salida or2 está desactivada.

Por ejemplo, tener dos botones pyvcp ambos conectados a un led.

El archivo .xml

```
<pyvcp>
  <button>
    <halpin>"button-1"</halpin>
    <text>"Button 1"</text>
  </button>

  <button>
    <halpin>"button-2"</halpin>
    <text>"Button 2"</text>
  </button>

  <led>
    <halpin>"led-1"</halpin>
    <size>50</size>
    <on_color>"green"</on_color>
    <off_color>"red"</off_color>
  </led>
</pyvcp>
```

El archivo postgui.hal

```
loadrt or2
addf or2.0 servo-thread
net button-1 or2.0.in0 <= pyvcp.button-1
net button-2 or2.0.in1 <= pyvcp.button-2
net led-1 pyvcp.led-1 <= or2.0.out
```

Cuando ejecuta este ejemplo creado con el Asistente Stepconf en una simulacion Axis, puede abrir un terminal y ver los pines creados con *loadrt o2* escribiendo *halcmd show pin or2* en el terminal.

```
halcmd show pin or2
Component Pins:
Owner   Type  Dir         Value  Name
    22  bit   IN          FALSE  or2.0.in0 <== button-1
    22  bit   IN          FALSE  or2.0.in1 <== button-2
    22  bit   OUT         FALSE  or2.0.out ==> led-1
```

Puede ver en la salida del comando hal *show pin or2* que el pin *button-1* esta conectado al pin *or2.0.in0* y por la flecha de dirección puede ver que el botón es una salida y *or2.0.in0* es una entrada. La salida de or2 va a la entrada del led.

14.10.2. Cambio manual de herramientas

En este ejemplo, se supone que está usando su propia configuración y desea agregar la ventana HAL Manual Toolchange. El cambio manual de herramientas HAL es principalmente útil si tiene herramientas preestablecidas y almacena los offsets en la

tabla de herramientas. Si necesita hacer touch off para cada cambio de herramienta, entonces es mejor dividir su código g. Para usar la ventana HAL Manual Toolchange básicamente tiene que cargar el componente hal_manualtoolchange y luego enviar el *cambio de herramienta* de iocontrol a el "cambio" de hal_manualtoolchange y devolver el cambio de hal_manualtool *cambiada* a iocontrol *herramienta cambiada*. Se proporciona un pin para una entrada externa para indicar que el cambio de herramienta está completo.

Este es un ejemplo de cambio de herramienta manual *con* el componente HAL Manual Toolchange:

```
loadusr -W hal_manualtoolchange
net tool-change iocontrol.0.tool-change => hal_manualtoolchange.change
net tool-changed iocontrol.0.tool-changed <= hal_manualtoolchange.changed
net external-tool-changed hal_manualtoolchange.change_button <= parport.0.pin-12-in
net tool-number iocontrol.0.tool-prep-number => hal_manualtoolchange.number
net tool-prepare-loopback iocontrol.0.tool-prepare => iocontrol.0.tool-prepared
```

Este es un ejemplo de cambio de herramienta manual *sin* el componente HAL Manual Toolchange:

```
net tool-number <= iocontrol.0.tool-prep-number
net tool-change-loopback iocontrol.0.tool.-change => iocontrol.0.tool-changed
net tool-prepare-loopback iocontrol.0.tool-prepare => iocontrol.0.tool-prepared
```

14.10.3. Cálculo de la velocidad

Este ejemplo usa *ddt*, *mult2* y *abs* para calcular la velocidad de un solo eje. Para obtener más información sobre los componentes en tiempo real, consulte las páginas de manual o la sección Componentes en tiempo real (Sección 14.8.2).

Lo primero es verificar su configuración para asegurarse de que no usa ninguno de los componentes en tiempo real. Puedes hacer esto abriendo la ventana de configuración de HAL y buscando los componentes en la sección de pines. Si encuentra alguno, busque el archivo .hal que están siendo cargado y aumente los count para ajustar las instancia al valor correcto. Agregue lo siguiente a su archivo custom.hal.

Carga de componentes de tiempo real.

```
loadrt ddt count=1
loadrt mult2 count=1
loadrt abs count=1
```

Agregue las funciones a un hilo para que se actualicen.

```
addf ddt.0 servo-thread
addf mult2.0 servo-thread
addf abs.0 servo-thread
```

Haga las conexiones.

```
setp mult2.in1 60
net xpos-cmd ddt.0.in
net X-IPS mult2.0.in0 <= ddt.0.out
net X-ABS abs.0.in <= mult2.0.out
net X-IPM abs.0.out
```

En esta última sección, se ha configurado mult2.0.in1 a 60 para convertir pulgadas por segundo, que obtenemos de ddt.0.out, a pulgadas por minuto .

La señal xpos-cmd envía la posición ordenada a ddt.0.in. El componente ddt calcula la derivada del cambio de la entrada, es decir, la velocidad.

ddt2.0.out se multiplica por 60 para dar IPM.

mult2.0.out se envía a abs para obtener el valor absoluto.

La siguiente figura muestra el resultado cuando el eje X se mueve a 15 IPM en la dirección negativa. Tenga en cuenta que podemos obtener el valor absoluto desde el pin abs.0.out o la señal X-IPM.

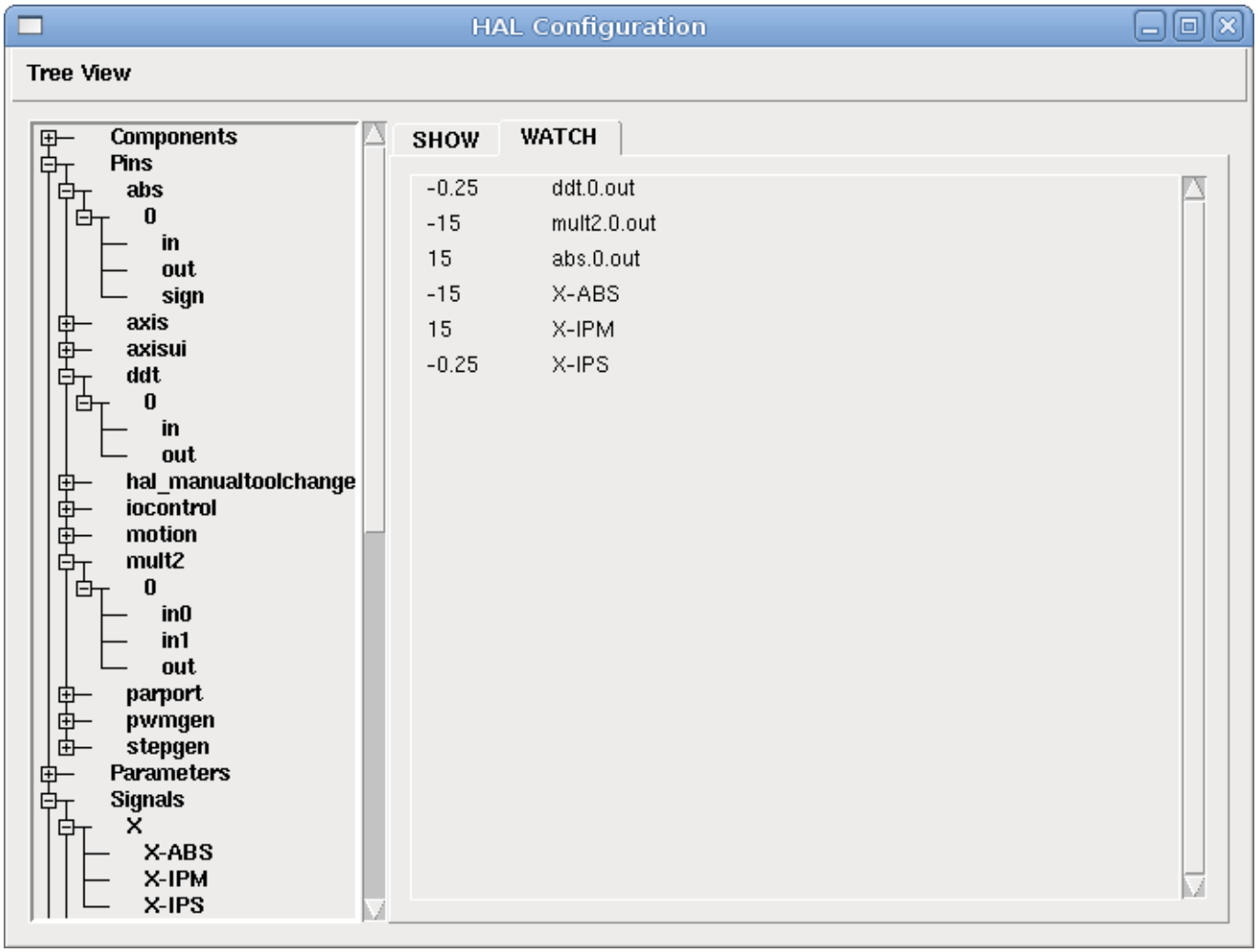


Figura 14.26: Ejemplo cálculo de velocidad

14.10.4. Arranque suave

Este ejemplo muestra cómo los componentes HAL *lowpass*, *limit2* o *limit3* puede usarse para limitar la rapidez con que cambia una señal.

En este ejemplo, tenemos un servomotor que impulsa un husillo de torno. Si solo usamos las velocidades de husillo ordenadas, el servo intentará ir desde la velocidad actual hasta la velocidad ordenada lo más rápido posible. Esto podría causar un problema o dañar la unidad. Para reducir la velocidad de cambio podemos envar spindle.N.speed-out a través de un limitador antes del PID, con lo que el valor del comando PID cambiará a nuevos valores más lentamente.

Los tres componentes integrados que limitan una señal son:

- *limit2* limita el rango y la primera derivada de una señal.
- *limit3* limita el rango, primera y segunda derivada de una señal.
- *lowpass* utiliza un promedio móvil ponderado exponencialmente para rastrear una señal de entrada.

Para encontrar más información sobre estos componentes HAL, consulte las páginas del manual.

Coloque lo siguiente en un archivo de texto llamado softstart.hal. Si no está familiarizado con Linux, coloque el archivo en su directorio de usuario.

```
loadrt threads period1=1000000 name1=thread
loadrt siggen
loadrt lowpass
loadrt limit2
loadrt limit3
net square siggen.0.square => lowpass.0.in limit2.0.in limit3.0.in
net lowpass <= lowpass.0.out
net limit2 <= limit2.0.out
net limit3 <= limit3.0.out
setp siggen.0.frequency .1
setp lowpass.0.gain .01
setp limit2.0.maxv 2
setp limit3.0.maxv 2
setp limit3.0.maxa 10
addf siggen.0.update thread
addf lowpass.0 thread
addf limit2.0 thread
addf limit3.0 thread
start
loadusr halscope
```

Abra una ventana de terminal y ejecute el archivo con el siguiente comando.

```
halrun -I softstart.hal
```

Cuando el osciloscopio HAL se inicie por primera vez, haga clic en *Aceptar* para aceptar el hilo predeterminado

A continuación, debe agregar las señales a los canales. Haga clic en el canal 1 y seleccione *cuadrado* en la pestaña Señales. Repita para los canales 2-4 y agregue lowpass, limit2 y limit3.

A continuación, para configurar una señal de disparo, haga clic en el botón Source None y seleccione cuadrado. El botón cambiará a Source Chan 1.

Luego haga clic en Single en el cuadro de botones de opción de Modo de ejecución. Esto comenzara a correr y cuando termine verá las trazas en el osciloscopio.

Para separar las señales para que pueda verlas mejor, haga clic en un canal y luego use el control deslizante Pos en el cuadro Vertical para establecer las posiciones.

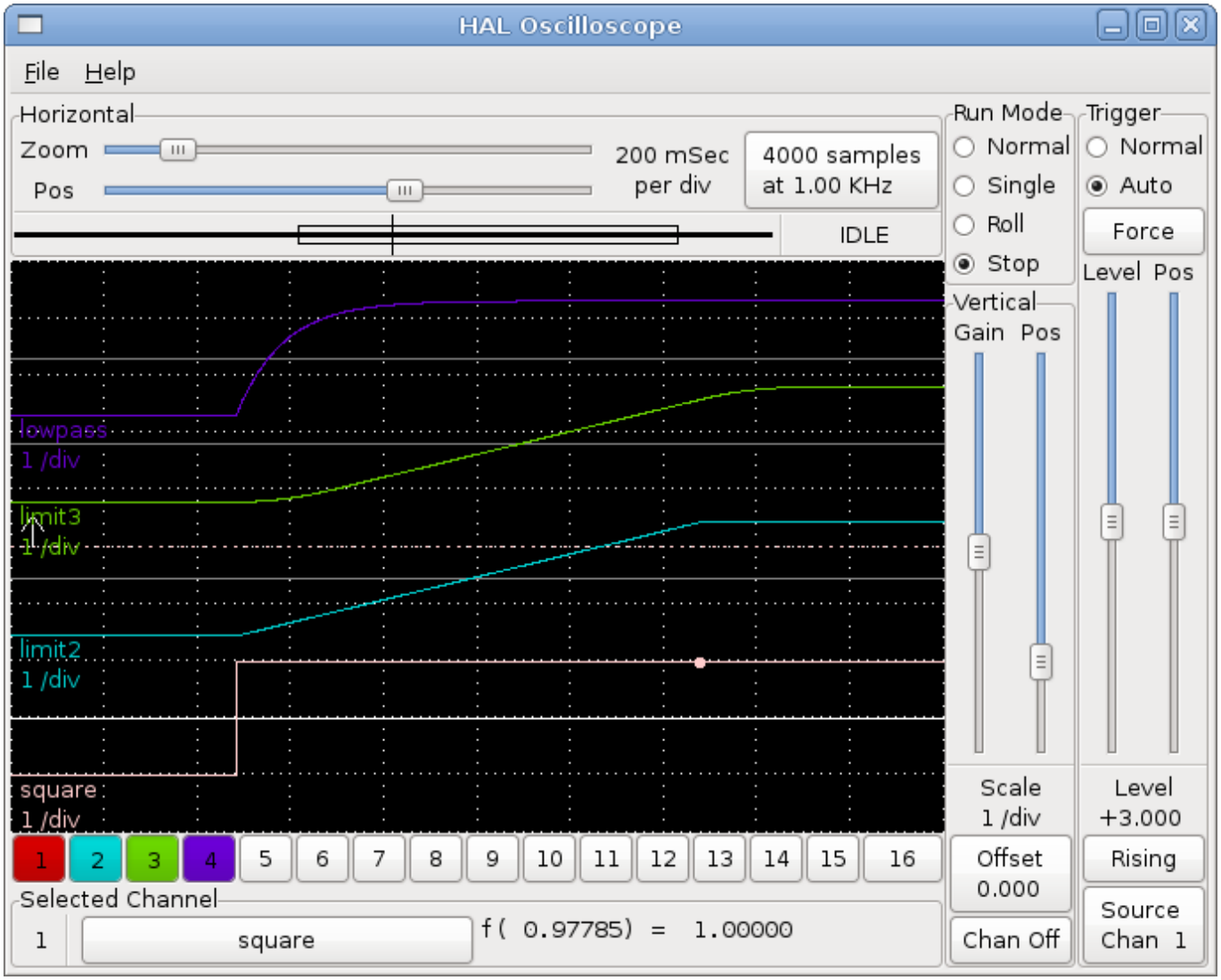


Figura 14.27: Arranque suave

To see the effect of changing the set point values of any of the components you can change them in the terminal window. To see what different gain settings do for lowpass just type the following in the terminal window and try different settings. Para ver el efecto de cambiar los valores del punto de ajuste de cualquiera de los componentes, puede cambiarlos en la ventana de terminal. Para ver qué hacen diferentes configuraciones de ganancia para lowpass simplemente escriba lo siguiente en la ventana de terminal y pruebe diferentes configuraciones.

```
setp lowpass.0.gain *.01
```

Después de cambiar una configuración, vuelva a ejecutar el osciloscopio para ver el cambio.

Cuando haya terminado, escriba *exit* en la ventana de terminal para cerrar halrun y halscope. No cierre la ventana del terminal con halrun corriendo, ya que podría dejar algunas cosas en la memoria que podrían interferir con la carga de LinuxCNC.

Para obtener más información sobre Halscope, consulte el manual de HAL.

14.10.5. Stand Alone HAL

In some cases you might want to run a GladeVCP screen with just HAL. For example say you had a stepper driven device that all you need is to run a stepper motor. A simple *Start/Stop* interface is all you need for your application so no need to load up and configure a full blown CNC application.

In the following example we have created a simple GladeVCP panel with one

14.10.6. HAL independiente

Basic Syntax En algunos casos, es posible que desee ejecutar una pantalla GladeVCP solo con HAL. Por ejemplo, digamos que tiene un dispositivo controlado por pasos que todo lo que necesita es ejecutar un motor paso a paso. Todo lo que necesita para su interfaz es "Iniciar/Parar", por lo que no es necesario cargar y configurar una aplicación CNC completa.

En el siguiente ejemplo, hemos creado un panel GladeVCP simple.

Sintaxis Básica

```
# cargar la GUI winder.glade y nombrarla winder
loadusr -Wn winder gladevcp -c winder -u handler.py winder.glade

# cargar componentes de tiempo real
loadrt threads name1=fast period1=50000 fp1=0 name2=slow period2=1000000
loadrt stepgen step_type=0 ctrl_type=v
loadrt hal_parport cfg="0x378 out"

# agregar funciones a hilos
addf stepgen.make-pulses fast
addf stepgen.update-freq slow
addf stepgen.capture-position slow
addf parport.0.read fast
addf parport.0.write fast

# hacer conexiones hal
net winder-step parport.0.pin-02-out <= stepgen.0.step
net winder-dir parport.0.pin-03-out <= stepgen.0.dir
net run-stepgen stepgen.0.enable <= winder.start_button



# iniciar los hilos
start

# comentar las siguientes líneas durante las pruebas y utilizar el interactivo
# option halrun -I -f start.hal to be able to show pins etc.

# espere hasta que la GUI gladevcp llamada winder termine
waitusr winder

# detener hilos HAL
stop

# descarguetodos los componentes HAL antes de salir
unloadrt all
```

14.11. El Generador de Componentes HAL

14.11.1. Introducción

Escribir un componente HAL puede ser un proceso tedioso; la mayor parte consiste en su configuración llamando a las funciones *rtapi_* y *hal_* y la comprobación de errores asociada. *halcompile* escribirá todo ese código automáticamente.

Cuando se usa *halcompile*, la compilación de un componente HAL es mucho más fácil si el componente es parte del árbol fuente de LinuxCNC, aunque tambien puede hacerse fuera de él.

Por ejemplo, cuando se codifica en C, un componente simple como "ddt" tiene alrededor de 80 líneas de código. El componente equivalente cuando se escribe usando el preprocesador *halcompile* es mucho mas corto:

Ejemplo de componente simple:

```
component ddt "Calcular la derivada de la función de entrada";
pin in float in;
pin out float out;
variable double old;
function _;
license "GPL"; // indica GPL v2 o posterior
;;
float tmp = in;
out = (tmp - old) / fperiod;
old = tmp;
```

14.11.2. Instalación

Si está trabajando con una versión instalada de LinuxCNC, necesitará instalar los paquetes de desarrollo.

Un método es usar la siguiente línea en una terminal.

Instalación del paquete de desarrollo:

```
sudo apt-get install linuxcnc-dev
o
sudo apt install linuxcnc-uspace-dev
```

Otro método es usar el administrador de paquetes Synaptic, desde el menú, para instalar linuxcnc-dev o linuxcnc-uspace-dev.

14.11.3. Usando un componente

Los componentes deben cargarse y agregarse a un hilo antes de poder usarlos. Ejemplo:

```
  loadrt threads name1=servo-thread period1=1000000
  loadrt ddt
  addf ddt.0 servo-thread
```

Se puede encontrar más información sobre loadrt y addf en la Referencia Básica de Hal .

Para probar su componente puede seguir los ejemplos en el tutorial de HAL.

14.11.4. Definiciones

- *componente* - Un componente es un módulo de tiempo real, único, que se carga con *halcmd loadrt*. Un componente se especifica con un archivo *.comp*. El nombre del componente y el nombre de archivo deben coincidir.
- *instancia* - Un componente puede tener cero o más instancias. Cada instancia de un componente se crea igual (todas tienen los mismos pines, parámetros, funciones y datos) pero se comportan independientemente cuando sus pines, parámetros y datos tienen diferentes valores.

  N.T. Las instancias de componentes pueden ser numeradas o con nombre.
- *singleton*: es posible que un componente sea un "singleton", en cuyo caso se crea una unica instancia. Rara vez tiene sentido escribir un componente *singleton* a menos que, literalmente, solo pueda haber un único objeto de ese tipo en el sistema (por ejemplo, un componente cuyo propósito es proporcionar un pin con la hora actual de UNIX, o un controlador de hardware para el altavoz interno de PC)

14.11.5. Creación de instancias

Para un componente singleton, la instancia única se crea cuando se carga el componente.

Para un no-singleton, el parámetro *count* del módulo determina cuantas instancias numeradas se crean. Si no se especifica *count*, el parámetro del módulo *names* determina cuántas instancias con nombre se crean. Si no se especifica ni *count* ni *names*, solo se crea una unica instancia numerada.

14.11.6. Parámetros implícitos

Las funciones pasan implícitamente el parámetro *period* que es el tiempo en nanosegundos del último período para ejecutar el componente. Las funciones que usan punto flotante también puede referirse a *fperiod*, que es el tiempo en coma flotante en segundos, o (period*1e-9). Esto puede ser útil en componentes que necesitan información de tiempo.

14.11.7. Sintaxis

Un archivo *.comp* consiste en varias declaraciones, seguidas por *";;"* en una línea propia, seguidas de código C que implementa las funciones del módulo.

Las declaraciones incluyen:

- *component HALNAME (DOC);*
- *pin PINDIRECTION TYPE HALNAME ([SIZE]\|[MAXSIZE: CONDSIZE]) (if CONDITION) (= STARTVALUE) (DOC) ;*
- *param PARAMDIRECTION TYPE HALNAME ([SIZE]\|[MAXSIZE: CONDSIZE]) (if CONDITION) (= STARTVALUE) (DOC) ;*
- *function HALNAME (fp | nofp) (DOC);*
- *option OPT (VALUE);*
- *variable CTYPE STARREDNAME ([SIZE]);*
- *description DOC;*
- *notes DOC;*
- *see_also DOC;*
- *license LICENSE;*
- *author AUTHOR;*
- *include HEADERFILE;*

Los paréntesis indican elementos opcionales. Una barra vertical indica alternativas. Las palabras en *MAYUSCULAS* indican texto variable, de la siguiente manera:

- *NAME* - Un identificador C estándar
- *STARREDNAME* - Un identificador C con cero o más * antes de él. Esta sintaxis puede ser utilizada para declarar variables de instancia que son punteros. Tenga en cuenta que debido a la gramática, es posible que no haya espacios en blanco entre el * y el nombre de la variable.
- *HALNAME* - Un identificador extendido. Cuando se usa para crear un identificador HAL, cualquier guión bajo se reemplaza con guiones, y cualquier guión o punto al final se elimina, por lo que "this_name_" se convertirá en "this-name", y si el nombre es "_", también se elimina un punto al final, de modo que "function _" da un nombre de función HAL como "component.<num>" en lugar de "component.<num>."

  Si está presente, el prefijo *hal_* se elimina del comienzo del nombre del componente al crear pines, parámetros y funciones.

  En el identificador HAL para un pin o parámetro, # denota un elemento de matriz, y debe usarse junto con una declaración *[SIZE]*. Las marcas hash (#) se reemplazan con un relleno de números 0 con la misma longitud que el número de caracteres #.

  Cuando HALNAME se usa para crear un identificador de C, se aplican los siguientes cambios:

1. Cualquier carácter "#" y cualquier carácter ".", "_" o "-" inmediatamente delante, se eliminan.
2. Cualquier "." restante y los caracteres "-" se reemplazan por "_".
3. Los caracteres repetidos "_" se cambian a un solo carácter "_".

Se conserva un "_" final, de modo que los identificadores HAL que de otro modo colisionarian con nombres reservados o palabras clave (por ejemplo, *min*) pueden ser utilizados.

HALNAME	Identificador C	Identificador HAL
x_y_z	x_y_z	x-y-z
x-y.z	x_y_z	x-y.z
x_y_z_	x_y_z_	x-y-z
x.##.y	x_y(MM)	x.MM.z
x.##	x(MM)	x.MM

- *if CONDITION* - Una expresión que involucra la variable *personality* , que no es cero cuando se debe crear el pin o parámetro
- *SIZE* - Un número que da el tamaño de una matriz. Los elementos de la matriz están numerados de 0 a (*SIZE*-1).
- *MAXSIZE : CONDSIZE* - Un número que da el tamaño máximo de la matriz seguido de una expresión que implica la variable *personality* y que siempre se evalúa a menos de *MAXSIZE*. Cuando la matriz se crea, su tamaño será *CONDSIZE*.
- *DOC* - Una cadena que documenta el elemento. La cadena puede ser "double quoted" de estilo C, como:

```
"Selecciona el flanco deseado: VERDADERO significa descendente, FALSO significa ascendente ↩
    "
```

o una cadena "triple quoted" al estilo Python, que puede incluir nuevas líneas incorporadas y caracteres de comillas, como:

```
"""El efecto de este parámetro, también conocido como "the orb of zot",
requerirá al menos dos párrafos para explicarlo.

Espero que estos párrafos te hayan permitido entender "zot"
mejor."""
```

o una cadena puede estar precedida por el carácter literal *r* (r-strings), en cuyo caso, la cadena se interpreta como una cadena en bruto (raw) de Python.

La cadena de documentación está en formato "groff -man". Para más información sobre este formato de marcado, ver *groff_man(7)*. Recuerde que *halcompile* interpreta escapes de barra invertida en cadenas. Por ejemplo para configurar la fuente en cursiva para la palabra *ejemplo*, escriba:

```
"\\fIejemplo\\fB"
```

En este caso, las r-strings son particularmente útiles, porque las barras invertidas en una cadena de caracteres no necesita duplicarse:

```
r"\fIexample\fB"
```

- *TYPE* - Uno de los tipos HAL; *bit*, *signed*, *unsigned* o *float*. Los viejos nombres *s32* y *u32* también se pueden usar, pero se prefiere *signed* y *unsigned*.
- *PINDIRECTION* - Uno de los siguientes; *in*, *out* o *io*. Un componente establece un valor para un pin *out*, lee un valor de un pin *in*, y puede leer o establecer el valor de un pin *io*.
- *PARAMDIRECTION* - Uno de los siguientes: *r* o *rw*. Un componente establece un valor para un parámetro *r*, y puede leer o establecer el valor de un parámetro *rw*.
- *STARTVALUE*: especifica el valor inicial de un pin o parámetro. Si no se especifica, el valor predeterminado es *0* o *FALSE*, según el tipo de objeto.
- *HEADERFILE* - El nombre de un archivo de encabezado, ya sea entre comillas dobles (`include "myfile.h";`) o en corchetes angulares (`include <systemfile.h>;`). El archivo de encabezado se incluirá (usando #include) en la parte superior del archivo, antes de las declaraciones de pines y parámetros.

14.11.7.1. Funciones HAL

- *fp* - Indica que la función realiza cálculos de coma flotante.
- *nofp*: indica que solo realiza cálculos enteros. Si no se especifica ninguno, se asume *fp*. Ni *halcompile* ni gcc pueden detectar el uso de cálculos de punto flotante en funciones etiquetadas como *nofp*, pero el uso de tales operaciones dan como resultado un comportamiento indefinido.

14.11.7.2. Opciones

Las opciones definidas actualmente son:

- *option singleton yes* - (valor predeterminado: no) No crear el parámetro de módulo *count*, y siempre crear una sola instancia. Con *singleton*, los elementos se denominan *nombre-componente.nombre-elemento* y sin *singleton*, los elementos, para las instancias numeradas, se nombran *nombre-component.<num>.nombre-elemento*.
- *option default_count number* - (valor predeterminado: 1) Normalmente, el parámetro de módulo *count* se establece de manera predeterminada en 1. Si se especifica, *count* cambiará a este valor por defecto.
- *option count_function yes* - (valor predeterminado: no) Normalmente, el número de instancias a crear se especifica en el parámetro del módulo *count*; si se especifica *count_function*, se usa en su lugar el valor devuelto por la función *int get_count(void)* , y el parámetro del módulo *count* no está definido.
- *opción rtapi_app no* - (predeterminado: yes) Normalmente, las funciones *rtapi_app_main()* y *rtapi_app_exit()* son definidas automáticamente. Con *option rtapi_app no*, no lo son, y debe ser previstas en el código C. Use los siguientes prototipos:
  ```
  int rtapi_app_main(void);
  void rtapi_app_exit(void);
  ```
  Al implementar su propio *rtapi_app_main()*, llame a la función *int export(char *prefix, long extra_arg)* para registrar los pines, parámetros y funciones para *prefix*.
- *option data TYPE* - (predeterminado: ninguno) **obsoleto**. Si se especifica, cada instancia del componente tendrá asociado un bloque de datos de tipo *TYPE* (que puede ser un tipo simple como *float* o el nombre de un tipo creado con *typedef*). En los componentes nuevos, se debe usar *variable* en su lugar.
- *option extra_setup yes* - (valor predeterminado: no) Si se especifica, llama a la función definida por *EXTRA_SETUP* en cada instancia. Si usa *rtapi_app_main* definido automáticamente, *extra_arg* es el número de esta instancia.
- *option extra_cleanup yes* - (valor predeterminado: no) Si se especifica, llama a la función definida por *EXTRA_CLEANUP* desde *rtapi_app_exit* definido automáticamente, o si se detecta un error en *rtapi_app_main* definido automáticamente.
- *option userspace yes* - (valor predeterminado: no) Si se especifica, este archivo describe un componente de espacio de usuario (es decir, no en tiempo real), en lugar de uno regular (es decir, en tiempo real). Un componente de espacio de usuario puede no tener funciones definidas por la directiva *function*. En cambio, después de que todas las instancias se construyan, se llama a la función C *void user_mainloop(void);*. Cuando esta función retorna, el componente sale. Normalmente, *user_mainloop()* usará *FOR_ALL_INSTS()* para realizar la acción de actualización para cada instancia, luego se detiene un tiempo corto. Otra acción común en *user_mainloop()* puede ser llamar al bucle del controlador de eventos de un toolkit de GUI.
- *option userinit yes* - (valor predeterminado: no) Esta opción se ignora si la opción *userspace* (ver arriba) está configurada en *no*. Si se especifica *userinit*, la función *userinit(argc, argv)* se llama antes que *rtapi_app_main()* (y por lo tanto antes de la llamada a *hal_init()*). Esta función puede procesar los argumentos de la línea de comando o tomar otras acciones. Su tipo de retorno es *void*; puede llamar a *exit()* si desea terminar en lugar de crear un componente HAL (por ejemplo, porque los argumentos de línea de comando no eran válidos).
- *option extra_link_args "... "* - (predeterminado: "") Esta opción se ignora si la opción *userspace* (ver arriba) está configurada en *no*. Al vincular un componente de espacio de usuario, se insertan los argumentos dados en la línea de enlace. Tenga en cuenta que debido a que la compilación tiene lugar en un directorio temporal, "-L" se refiere al directorio temporal y no al directorio donde el archivo fuente .comp reside.

Si el VALOR de una opción no está especificado, entonces es equivalente a especificar *option ... yes*. El resultado de asignar un valor inapropiado a una opción no está definido. El resultado de usar cualquier otra opción no está definido.

14.11.7.3. Licencia y autoría

- *LICENSE*: especifica la licencia del módulo para la documentación y para la declaración del módulo MODULE_LICENSE (). Por ejemplo, para especificar que la licencia del módulo es GPL v2 o posterior,

  ```
  license "GPL"; // indica GPL v2 o posterior
  ```

  Para obtener información adicional sobre el significado de MODULE_LICENSE() e identificadores de licencia adicionales, consulte <*linux/module.h*>. o la página de manual *rtapi_module_param(3)*

  Esta declaración es obligatoria.

- *AUTHOR*: especifica el autor del módulo para la documentación.

14.11.7.4. Almacenamiento de datos por instancia

- *variable CTYPE STARREDNAME;*
- *variable CTYPE STARREDNAME[SIZE];*
- *variable CTYPE STARREDNAME = DEFAULT;*
- *variable CTYPE STARREDNAME[SIZE] = DEFAULT;*

  Declare una variable *STARREDNAME* por instancia, de tipo *CTYPE*, opcionalmente como una matriz de *SIZE* elementos, y opcionalmente con un valor *DEFAULT* predeterminado. Los elementos sin *DEFAULT* se inicializan con todos los bits a cero. *CTYPE* es una palabra de tipo de C, como *float*, *u32*, *s32*, *int*, etc. El acceso a variables de matriz usa corchetes.

  Si una variable debe ser de tipo puntero, no debe haber espacio entre el "*" y el nombre de la variable. Por lo tanto, lo siguiente es aceptable:

  ```
  variable int *ejemplo;
  ```

  pero los siguientes no lo son:

  ```
  variable int* badexample;
  variable int * badexample;
  ```

14.11.7.5. Comentarios

En la sección de declaración, son compatibles comentarios de una línea de estilo C++ (// . . .) y comentarios multilínea estilo C (/* . . . */).

14.11.8. Restricciones

Aunque HAL permite que un pin, un parámetro y una función tengan el mismo nombre, *halcompile* no.

Nombres de variables y funciones que no se pueden usar o que pueden causar problemas incluyen:

- Cualquier cosa que comience con _comp.
- *comp_id*
- *fperiod*
- *rtapi_app_main*
- *rtapi_app_exit*
- *extra_setup*
- *extra_cleanup*

14.11.9. Macros de Conveniencia

En función de los elementos en la sección de declaración, *halcompile* crea una estructura C llamada `struct __comp_state`. Sin embargo, en lugar de referirse a los miembros de esta estructura (por ejemplo, `*(inst->name)`), generalmente serán referidos usando las macros que siguen. Los detalles de `struct __comp_state` y estas macros pueden cambiar de una versión de *halcompile* a la siguiente.

- *FUNCTION(name)* - Use esta macro para comenzar la definición de una función en tiempo real que fue declarada previamente con *function NAME*. La función incluye un parámetro *period* que es el número entero de nanosegundos entre llamadas a la función.
- *EXTRA_SETUP()* - Use esta macro para comenzar la definición de la función llamada a realizar una configuración adicional de esta instancia. Devuelve un *errno* negativo de Unix para indicar fallo (por ejemplo, *return -EBUSY* al no reservar un puerto de E/S) o 0 para indicar éxito.
- *EXTRA_CLEANUP()* - Use esta macro para comenzar la definición de la función destinada a realizar una configuracion adicional del componente. Tenga en cuenta que esta función debe limpiar todas las instancias del componente, no solo una. Las macros "pin_name", "parameter_name" y "data" no se pueden usar aquí.
- *pin_name* o *parameter_name* - Para cada pin *pin_name* o parametro *parameter_name* hay una macro que permite que el nombre se use solo para referirse al pin o parámetro. Cuando *pin_name* o *parameter_name* es una matriz, la macro es de la forma *pin_name(idx)* o *param_name(idx)* donde *idx* es el índice en la matriz. Cuando la matriz es de tamaño variable, solo es legal referirse a elementos hasta su *condsize*.

  Cuando el elemento es condicional, solo es legal referirse a él cuando su *condition* se evaluó a un valor distinto de cero.
- *variable_name* - Para cada variable *variable_name* hay una macro que permite usar el nombre que se utiliza para referirse a la variable. Cuando *variable_name* es una matriz, se usa el subíndice estilo C normal: *variable_name[idx]*
- *data* - Si se especifica "option data", esta macro permite el acceso a los datos de la instancia.
- *fperiod*: el número de segundos en, coma flotante, entre las llamadas a esta funcion de tiempo real.
- *FOR_ALL_INSTS() {...}* - Para componentes de espacio de usuario. Esta macro itera sobre todas las instancias definidas. Dentro del cuerpo del lazo, las macros *pin_name*, *parameter_name* y *data* funcionan como lo harian en funciones en tiempo real.

14.11.10. Componentes con una sola función

Si un componente tiene solo una función y la cadena "FUNCTION" no no aparecerá en ningún lugar después de *;;*, el resto sera tomado como el cuerpo de una función única del componente. Ver Simple Comp como ejemplo.

14.11.11. Personalidad del componente

Si un componente tiene pines o parámetros con una "if condition" o "[maxsize : condsize]", se llama componente con *personalidad*. La *personalidad* de cada instancia se especifica cuando el módulo está cargado. *Personality* se puede usar para crear pines solo cuando sea necesario. Por ejemplo, la personalidad se usa en el componente *logic*, para permitir un número variable de pines de entrada a cada puerta lógica y la selección de cualquiera de las funciones lógicas booleanas básicas *and*, *or*, y *xor* o combinacion de ellas.

El número predeterminado de elementos de *personalidad* permitidos es una configuración de tiempo de compilación (64). El valor predeterminado se aplica a numerosos componentes incluidos en la distribución que se crean utilizando halcompile.

Para alterar el número permitido de elementos de personalidad para componentes creados por el usuario, use la opción *--personality* con halcompile. Por ejemplo, para permitir hasta 128 "personality":

```
[sudo] halcompile --personality=128 --install ...
```

Cuando se usan componentes con personalidad, el uso normal es especificar un elemento de personalidad para **cada** instancia de componente especificada. Ejemplo para 3 instancias del componente logic:

```
loadrt logic names=and4,or3,nand5, personality=0x104,0x203,0x805
```

nota
Si una línea loadrt especifica más instancias que personalidades, a las instancias con personalidades no especificadas se les asigna una personalidad de 0. Si el número solicitado de instancias excede el número de personalidades permitidas, las personalidades se asignan mediante indexación módulo del número de personalidades permitidas. Se imprime un mensaje que denota tales asignaciones.

14.11.12. Compilando

Coloque el archivo *.comp* en el directorio fuente *linuxcnc/src/hal/components* y vuelva a ejecutar *make*. Los archivos *.comp* son automáticamente detectados por el sistema de compilación.

Si un archivo *.comp* es un controlador de hardware, puede colocarse en *linuxcnc/src/hal/drivers* y se compilará a menos que LinuxCNC sea configurado como simulador de espacio de usuario.

14.11.13. Compilación de componentes en tiempo real fuera del árbol fuente

halcompile puede procesar, compilar e instalar un componente en tiempo real en un solo paso, colocando el modulo producido *rtexample.ko* en el directorio de módulos en tiempo real de LinuxCNC:

```
halcompile --install rtexample.comp
```

O bien, puede procesar y compilar en un solo paso, dejando el modulo *example.ko* (o *example.so* para el simulador) en el directorio actual:

```
halcompile --compile example.comp
```

O simplemente puede procesar, dejando *example.c* en el directorio actual:

```
halcompile rtexample.comp
```

halcompile también puede compilar e instalar un componente escrito en C, usando las opciones *--install* y *--compile* que se muestran arriba:

```
halcompile --install rtexample2.c
```

la documentación en formato man también se puede crear a partir de la información en la sección de declaración:

```
halcompile --document rtexample.comp
```

La página de manual resultante, *ejemplo.9* se puede ver con

```
man ./ejemplo.9
```

o copiandola a una ubicación estándar de páginas de manual.

14.11.14. Compilación de componentes de espacio de usuario fuera del árbol de fuentes

halcompile puede procesar, compilar, instalar y documentar componentes de espacio de usuario:

```
halcompile usrexample.comp
halcompile --compile usrexample.comp
halcompile --install usrexample.comp
halcompile --document usrexample.comp
```

Esto solo funciona con archivos *.comp*, no con archivos *.c*.

14.11.15. Ejemplos

14.11.15.1. constant

La declaración "function _" crea funciones llamadas "constant.0" , etc. El nombre del archivo debe coincidir con el nombre del componente.

```
component constant;
pin out float out;
param r float value = 1.0;
function _;
license "GPL"; // indica GPL v2 o posterior
;;
FUNCTION(_) { out = value; }
```

14.11.15.2. sincos

Este componente calcula el seno y el coseno de un ángulo en radianes. Tiene diferentes capacidades que las salidas "seno" y "coseno" de siggen, porque la entrada es un ángulo, en lugar de correr libremente basado en un parámetro de "frecuencia".

Los pines se declaran con los nombres *sin_* y *cos_* en el código fuente para que no interfieran con las funciones *sin()* y *cos()*. Los pines HAL, sin embargo, se llaman *sincos.<num>.sin*.

```
component sincos;
pin out float sin_;
pin out float cos_;
pin in float theta;
function _;
license "GPL"; // indica GPL v2 o posterior
;;
#include <rtapi_math.h>
FUNCTION(_) { sin_ = sin(theta); cos_ = cos(theta); }
```

14.11.15.3. out8

Este componente es un controlador para una tarjeta *ficticia* llamada "out8", que tiene 8 pines de salida digital que son tratado como un solo valor de 8 bits. Puede haber un número variable de tales tarjetas en el sistema, y pueden estar en varias direcciones. El pin es llamado *out_* porque *out* es un identificador utilizado en *<asm/io.h>*. Eso ilustra el uso de *EXTRA_SETUP* y *EXTRA_CLEANUP* para solicitar una región de E / S y luego liberarla en caso de error o cuando el módulo sea descargado

```
component out8;
pin out unsigned out_ "Valor de salida, solo se usan los 8 bits mas bajos";
param r unsigned ioaddr;

function _;

option count_function;
option extra_setup;
option extra_cleanup;
option constructable no;

license "GPL"; // indica GPL v2 o posterior
;;
#include <asm/io.h>

#define MAX 8
int io[MAX] = {0,};
RTAPI_MP_ARRAY_INT(io, MAX, "direcciones de E/S de las tarjetas out8");
```

```
int get_count(void) {
    int i = 0;
    for(i=0; i<MAX && io[i]; i++) { /* Nada */ }
    return i;
}

EXTRA_SETUP() {
    if(!rtapi_request_region(io[extra_arg], 1, "out8")) {
        // establecer este puerto de E/S a 0 para que EXTRA_CLEANUP no libere
        // puertos IO que nunca fueron solicitados
        io [extra_arg] = 0;
        io[extra_arg] = 0;
        return -EBUSY;
    }
    ioaddr = io[extra_arg];
    return 0; }

EXTRA_CLEANUP() {
    int i;
    for(i=0; i < MAX && io[i]; i++) {
        rtapi_release_region(io[i], 1);
    }
}

FUNCTION(_) { outb(out_, ioaddr); }
```

14.11.15.4. hal_loop

```
component hal_loop;
pin out float example;
```

Este fragmento de un componente ilustra el uso del prefijo *hal_* en un nombre de componente. *loop* es el nombre de un módulo de kernel Linux estándar, por lo que otro componente de nombre *loop* podría no cargarse correctamente si el módulo *loop* de Linux también estaba presente en el sistema.

Cuando se carga, *halcmd show comp* mostrará un componente llamado *hal_loop*. Sin embargo, el pin que se muestra con *halcmd show pin* será *loop.0.example*, no *hal-loop.0.example*.

14.11.15.5. arraydemo

Este componente en tiempo real ilustra el uso de matrices de tamaño fijo:

```
component arraydemo "4-bit Shift register";
pin in bit in;
pin out bit out-# [4];
function _ nofp;
license "GPL"; // indica GPL v2 o posterior
;;
int i;
for(i=3; i>0; i--) out(i) = out(i-1);
out(0) = in;
```

14.11.15.6. rand

Este componente de espacio de usuario cambia el valor en su pin de salida a un nuevo valor aleatorio en el rango (0,1) aproximadamente una vez cada 1 ms.

```
component rand;
option userspace;

pin out float out;
license "GPL"; // indica GPL v2 o posterior
;;
#include <unistd.h>

void user_mainloop(void) {
    while(1) {
        usleep(1000);
        FOR_ALL_INSTS() out = drand48();
    }
}
```

14.11.15.7. logic

Este componente en tiempo real muestra cómo usar la "personalidad" para crear matrices de tamaño variable y pines opcionales.

```
component logic "componente LinuxCNC HAL que proporciona funciones lógicas experimentales";
pin in bit in-##[16 : personality & 0xff];
pin out bit and if personality & 0x100;
pin out bit or if personality & 0x200;
pin out bit xor if personality & 0x400;
function _ nofp;
description """
Componente 'función lógica' general experimental. Puede realizar 'and', 'or'
y 'xor' de hasta 16 entradas. Determine el valor apropiado para 'personalidad'
añadiendo:
.IP \\(bu 4
El número de pines de entrada, generalmente de 2 a 16
.IP \\(bu
256 (0x100) si se desea la salida 'and'
.IP \\(bu
512 (0x200) si se desea la salida 'or'
.IP \\(bu
1024 (0x400) si se desea la salida 'xor' (or exclusivo)""";
licencia "GPL"; // indica GPL v2 o posterior
;;
FUNCTION(_) {
    int i, a=1, o=0, x=0;
    for(i=0; i < (personality & 0xff); i++) {
        if(in(i)) { o = 1; x = !x; }
        else { a = 0; }
    }
    if(personality & 0x100) and = a;
    if(personality & 0x200) or = o;
    if(personality & 0x400) xor = x;
}
```

Una línea típica para cargar este componente podría ser

```
loadrt logic count=3 personality=0x102,0x305,0x503
```

que crea los siguientes pines:

- Una puerta AND de 2 entradas: logic.0.and, logic.0.in-00, logic.0.in-01
- Puertas AND y OR de 5 entradas: logic.1.and, logic.1.or, logic.1.in-00, logic.1.in-01, logic.1.in-02, logic.1.in-03, logic.1.in-04,
- Puertas AND y XOR de 3 entradas: logic.2.and, logic.2.xor, logic.2.in-00, logic.2.in-01, logic.2.in-02

14.11.15.8. funciones generales

Este ejemplo muestra cómo llamar a funciones desde la función principal.
También muestra cómo pasar la referencia de los pines HAL a esas funciones.

```
component example;
pin in s32 in;
pin out bit out1;
pin out bit out2;

function _;
license "GPL";
;;

// pin general establece función verdadera
void set(hal_bit_t *p){
    *p = 1;
}

// pin general establece función falsa
void unset(hal_bit_t *p){
    *p = 0;
}

//función principal
FUNCTION(_) {
    if (in < 0){
        set(&out1);
        unset(&out2);
    }else if (in >0){
        unset(&out2);
        set(&out2);
    }else{
        unset(&out1);
        unset(&out2);
    }

}
```

Este componente utiliza dos funciones generales para manipular un pin bit HAL al que se hace referencia.

14.11.16. Uso de línea de comando

La página de manual de halcompile proporciona detalles para invocar halcompile.

```
$ hombre halcompile
```

Un breve resumen del uso de halcompile está dado por:

```
$ halcompile --help
```

14.12. Crear Componentes Python de Espacio de Usuario

14.12.1. Uso Básico

Un componente de espacio de usuario comienza creando sus pines y parámetros. Luego entra en un bucle que manejara periódicamente todas las salidas segun las entradas. El siguiente componente copia el valor visto en su pin de entrada (*passthrough.in*)

a su pin de salida (*passthrough.out*) aproximadamente una vez por segundo.

```
#!/usr/bin/env python
import hal, time
h = hal.component("passthrough")
h.newpin("in", hal.HAL_FLOAT, hal.HAL_IN)
h.newpin("out", hal.HAL_FLOAT, hal.HAL_OUT)
h.ready()
try:
    while 1:
        time.sleep(1)
        h['out'] = h['in']
except KeyboardInterrupt:
    raise SystemExit
```

Copie el listado anterior en un archivo llamado "passthrough", hágalo ejecutable (*chmod +x*), y colóquelo en su *$PATH*. Luego, pruébelo:

```
Halrun

halcmd: loadusr passthrough

halcmd: show pin

    Component Pins:
    Owner Type  Dir     Value  Name
     03   float IN          0  passthrough.in
     03   float OUT         0  passthrough.out

halcmd: setp passthrough.in 3.14

halcmd: show pin

    Component Pins:
    Owner Type  Dir     Value  Name
     03   float IN       3.14  passthrough.in
     03   float OUT      3.14  passthrough.out
```

14.12.2. Componentes de espacio de usuario y retrasos

Si escribió el segundo "show pin" rápidamente, puede ver que *passthrough.out* todavía tenía su valor anterior de 0. Esto se debe a la llamada a *time.sleep(1)*, que hace que la asignación al pin de salida ocurra al menos una vez por segundo. Como este es un componente de espacio de usuario, la demora real entre asignaciones puede ser mucho más larga si la memoria utilizada por el componente passthrough se intercambia en el disco; la asignación se puede retrasar hasta que la memoria se intercambie nuevamente.

Por lo tanto, los componentes del espacio de usuario son adecuados para elementos interactivos con el usuario como los paneles de control (los retrasos en el rango de milisegundos no seran notados, y son aceptables demoras más largas), pero no para enviar pulsos de paso a una placa de control paso a paso (los retrasos siempre deben estar en el rango de microsegundos).

14.12.3. Crear pines y parámetros

```
h = hal.component("passthrough")
```

El componente en sí es creado por una llamada al constructor *hal.component*. Los argumentos son el nombre del componente HAL y (opcionalmente) el prefijo utilizado para los nombres de los pines y los parámetros. Si el prefijo no se especifica, se usa el nombre del componente.

```
h.newpin("in", hal.HAL_FLOAT, hal.HAL_IN)
```

Los pines se crean mediante llamadas a métodos en el objeto componente. Los argumentos son: sufijo del nombre del pin, tipo de pin y dirección del pin. Para parámetros, los argumentos son: sufijo del nombre del parámetro, tipo de parámetro, y dirección del parámetro.

Cuadro 14.3: Nombres de la opción .HAL

Tipos Pines y Parametros:	HAL_BIT	HAL_FLOAT	HAL_S32	HAL_U32
Direcciones de Pin:	HAL_IN	HAL_OUT	HAL_IO	
Direcciones de Parametros:	HAL_RO	HAL_RW		

El nombre completo del pin o parámetro se forma uniendo el prefijo y el sufijo con un ".", por lo que en el ejemplo, el pin creado se llama *passthrough.in*.

```
h.ready()
```

Una vez que se hayan creado todos los pines y parámetros, llame al método *.ready()*.

14.12.3.1. Cambiar el Prefijo

El prefijo se puede cambiar llamando al método *.setprefix()*. Los prefijo actuales se puede recuperar llamando al método *.getprefix()*.

14.12.4. Lectura y Escritura de Pines y Parámetros

Para los pines y parámetros que también sean identificadores adecuados de Python, el valor puede ser accedido o establecido usando la sintaxis de atributo:

```
h.out = h.in
```

Para todos los pines, sean o no identificadores adecuados de Python, el valor puede ser accedido o establecido usando la sintaxis de subíndice:

```
h['out'] = h['in']
```

14.12.4.1. Manejo de Pines de Salida (HAL_OUT)

Periódicamente, generalmente en respuesta a un temporizador, todos los pines HAL_OUT deben ser "manejados" al asignarles un nuevo valor. Esto debe hacerse, sea o no el valor diferente al último asignado. Cuando un pin es conectado a una señal, su valor de salida anterior no se copia en la señal, por lo que el valor correcto solo aparecerá en la señal una vez que el componente asigna un nuevo valor.

14.12.4.2. Manejo de pines bidireccionales (HAL_IO)

La regla anterior no se aplica a los pines bidireccionales. En cambio, un pin bidireccional solo debe ser manejado por el componente cuando el componente desea cambiar el valor. Por ejemplo, en el interfaz canónico de encoder, el componente solo establece el pin *index-enable* a **FALSE** (cuando se produce un pulso de índice y el valor anterior es **TRUE**), pero nunca, por si mismo, lo establece en **TRUE**. Llevando repetidamente el pin a **FALSE** puede hacer que el otro componente conectado actúe como si hubiese sido visto otro impulso de índice.

14.12.5. Salida

Una solicitud *halcmd unload* para el componente se entrega como una excepción *KeyboardInterrupt*. Cuando llega una solicitud de descarga, el proceso debe o bien salir en poco tiempo o llamar al método *.exit()* en el componente, si un trabajo sustancial (como lectura o escritura de archivos) debe hacerse para completar el proceso de apagado.

14.12.6. Funciones útiles

14.12.6.1. component_exists

Existencia del componente especificado en este momento
Ejemplo:
hal.component_exists("testpanel")

14.12.6.2. component_is_ready

Componente especificado listo en este momento
Ejemplo:
hal.component_is_ready("testpanel")

14.12.6.3. get_msg_level

Nivel msg de tiempo real actual.

14.12.6.4. set_msg_level

Establecer nivel msg de tiempo real.
usado para informacion de depuracion.

14.12.6.5. connect

Conecta un pin a una señal.
ejemplo:
hal.connect("pinname","signal_name")

14.12.6.6. get_value

leer pin, parametro o señal directamente.
ejemplo:
value = hal.get_value("iocontrol.0.emc-enable-in")

14.12.6.7. new_signal

Crea una nueva señal del tipo especificado.
ejemplo "+ hal.new_sig("nombre-de-la-señal",hal.HAL_BIT)

14.12.6.8. pin_has_writer

El pin especificado tiene un pin de manejo conectado
Devuelve verdadero o falso.
h.in.pin_has_writer()

14.12.6.9. get_name

Obtener nombre de objeto HAL
h.in.get_name()
devuelve una cadena

14.12.6.10. get_type

Obtener tipo de objeto HAL
h.in.get_type()
devuelve un entero

14.12.6.11. get_dir

Obtener tipo de direccion de objeto HAL
h.in.get_dir()
devuelve un entero

14.12.6.12. get

Obtener valor del objeto HAL
h.in.get()

14.12.6.13. set

Establecer valor del objeto HAL
h.out.set(10)

14.12.6.14. is_pin

el objeto es pin o parametro?
h.in.is_pin()
devuelve bool

14.12.6.15. sampler_base

TODO

14.12.6.16. stream_base

TODO

14.12.6.17. stream

TODO

14.12.6.18. set_p

Establecer un valor en cualquier pin HAL.
ejemplo:
hal.set_p("pinname","10")

14.12.7. Constantes

Úselas para especificar detalles con el valor que representan.

- HAL_BIT
- HAL_FLOAT
- HAL_S32
- HAL_U32
- HAL_IN
- HAL_OUT
- HAL_RO
- HAL_RW
- MSG_NONE
- MSG_ALL
- MSG_DBG
- MSG_ERR
- MSG_INFO
- MSG_WARN

14.12.8. Información del sistema

Leer estas variables para obtener información sobre el sistema en tiempo real.

- is_kernelspace
- is_rt
- is_sim
- is_userspace

14.12.9. Usar con hal_glib en el handler GladeVCP

GladeVCP usa la biblioteca hal_glib, que puede usarse para conectar una señal "observador" en un pin de entrada HAL.
Esta señal se puede usar para registrar una función a llamar cuando el pin HAL cambia de estado.

Uno debe importar el módulo y el módulo hal:

```
import hal_glib
import hal
```

Luego haga un pin y conecte una señal *value-changed* (el observador) a una llamada de función:

```
class HandlerClass:
    def __init__(self, halcomp,builder,useropts):
        self.example_trigger = hal_glib.GPin(halcomp.newpin('example-trigger', hal.HAL_BIT, ↩
            hal.HAL_IN))
        self.example_trigger.connect('value-changed', self._on_example_trigger_change)
```

Y tener una función que se llamará:

```
    def _on_example_trigger_change(self,pin,userdata=None):
        print "pin value changed to:" % (pin.get())
        print "pin name= %s" % (pin.get_name())
        print "pin type= %d" % (pin.get_type())

        # esto se puede llamar fuera de la función
        self.example_trigger.get()
```

14.12.10. Usar con hal_glib en el handler QtVCP

QtVCP usa la biblioteca hal_glib, que puede usarse para conectar una señal "observador" en un pin de entrada HAL.
Esta señal se puede usar para registrar una función a llamar cuando el pin HAL cambia de estado.

Uno debe importar el módulo hal:

```
import hal
```

Luego haga un pin y conecte una señal *value_changed* (el observador) a una llamada de función:

```
    ########################
    # **** INICIALIZAR **** #
    ########################
    # widgets permite el acceso a widgets desde los archivos qtvcp
    # en este punto, los widgets y los pines hal no están instanciados
    def __init__(self, halcomp,widgets,paths):
        self.hal = halcomp
        self.testPin = self.hal.newpin('test-pin', hal.HAL_BIT, hal.HAL_IN)
        self.testPin.value_changed.connect(lambda s: self.setTestPin(s))
```

Y tener una función que se llamará.
Esto muestra formas de obtener el valor y la información del pin.

```
    ########################
    # funciones generales #
    ########################
    def setTestPin(self, data):
        print "Test pin value changed to:" % (data)
        print 'halpin object =', self.w.sender()
```

```
        print 'Halpin name: ',self.sender().text()
        print 'Halpin type: ',self.sender().get_type()

        # esto se puede llamar fuera de la función
        print self.testPin.get()
```

14.12.11. Ideas de proyectos

- Crea un panel de control externo con botones, interruptores y indicadores. Conecte todo a un microcontrolador, y conecte el microcontrolador a la PC con una interfaz serie. Python tiene un muy eficaz módulo de interfaz serie llamado pyserial (Nombre del paquete de Ubuntu "python-serial", en el repositorio universo)
- Adjunte un módulo LCD compatible con LCDProc- y úselo para mostrar una lectura digital con la información que elija (Nombre del paquete de Ubuntu "lcdproc", en el repositorio universo)
- Crear un panel de control virtual utilizando cualquier biblioteca GUI compatible con Python (gtk, qt, wxwindows, etc.)

Parte V

Advanced Topics

Capítulo 15

Cinemática

15.1. Introducción

Cuando hablamos de máquinas CNC, solemos pensar en máquinas que se les ordena moverse a ciertas ubicaciones y realizar varias tareas. Para tener una vista unificada del espacio de la máquina, y para adaptarlo al punto de vista humano del espacio 3D, la mayoría de las máquinas (si no todas) usan un sistema de coordenadas común llamado Sistema de Coordenadas Cartesianas.

El sistema de coordenadas cartesianas se compone de tres ejes (X, Y, Z) cada uno perpendicular a los otros dos. [1].

Cuando hablamos de un programa de código G (RS274/NGC) hablamos de un conjunto de comandos (G0, G1, etc.) que tienen posiciones cartesianas como parámetros (X- Y- Z-). Estas posiciones se refieren exactamente a las posiciones cartesianas. Parte del controlador de movimiento LinuxCNC es responsable de traducir esas posiciones en posiciones que corresponden a la Cinemática de la máquina. [2]

15.1.1. Articulaciones vs. Ejes

Una articulacion de una máquina CNC es uno de los grados físicos de libertad de la máquina. Este podría ser lineal (tornillos de avance) o rotativo (mesas rotativas, articulaciones de brazo de robot). Puede haber cualquier cantidad de articulaciones en una máquina dada. Por ejemplo, un brazo robot tipico tiene 6 articulaciones, y una fresadora simple típica tiene solo 3.

Hay ciertas máquinas donde las articulacions se disponen para que coincidan con los ejes cinemáticos (articulación 0 a lo largo del eje X, articulación 1 a lo largo del eje Y, articulación 2 a lo largo del eje Z); a estas máquinas se les llama Máquinas cartesianas (o máquinas con Cinemática Trivial). Estas son las máquinas más comunes utilizadas en el fresado, pero no son muy comunes en otros dominios (por ejemplo, soldadura: robots de tipo puma).

LinuxCNC admite nueve ejes con nombres: X Y Z A B C U V W. Los ejes X Y Z típicamente se refieren a las coordenadas cartesianas habituales. Los ejes A B C se refieren a coordenadas de rotación sobre los ejes X Y Z respectivamente. Los ejes U V W se refieren a coordenadas adicionales que comúnmente se hacen colineales a los ejes X Y Z respectivamente.

15.2. Cinemática Trivial

Las máquinas más simples son aquellas en las que se coloca cada articulación a lo largo de uno de los ejes cartesianos. En estas máquinas, el mapeo de espacio cartesiano (el programa de código G) al espacio de articulaciones (los actuadores de la máquina) es trivial. Es un mapeo simple 1:1.

```
pos->tran.x = joints[0];
pos->tran.y = joints[1];
pos->tran.z = joints[2];
```

[1] La palabra "eje" es comúnmente (y erróneamente) usada cuando se habla de máquinas CNC, y se refieren a las direcciones de movimiento de la máquina

[2] Cinemática: una función bidireccional para transformar del espacio cartesiano al espacio de articulaciones

En el fragmento de código de arriba se puede ver cómo se hace la asignación; la posición X es idéntica a la articulación 0, la posición Y a la de articulación 1, etc. Lo anterior se refiere a la cinemática directa (una dirección de la transformación). El siguiente fragmento de código se refiere a la cinemática inversa (o dirección inversa de la transformación):

```
joints[0] = pos->tran.x;
joints[1] = pos->tran.y;
joints[2] = pos->tran.z;
```

En LinuxCNC, la cinemática de identidad (1:1) se implementa con el módulo cinemático *trivkins* extendido a 9 ejes. El valor por defecto de las relaciones entre las coordenadas del eje y los números de articulaciones son:

```
pos->tran.x = joints[0];
pos->tran.y = joints[1];
pos->tran.z = joints[2];
pos->a      = joints[3];
pos->b      = joints[4];
pos->c      = joints[5];
pos->u      = joints[6];
pos->v      = joints[7];
pos->w      = joints[8];
```

Del mismo modo, las relaciones predeterminadas para la cinemática inversa para trivkins son:

```
joints[0] = pos->tran.x;
joints[1] = pos->tran.y;
joints[2] = pos->tran.z;
joints[3] = pos->a;
joints[4] = pos->b;
joints[5] = pos->c;
joints[6] = pos->u;
joints[7] = pos->v;
joints[8] = pos->w;
```

Es sencillo hacer la transformación para cinematica trivial (cinemática *trivkins*) o máquina cartesiana siempre que no haya omisiones en las letras de eje usadas.

Si a la máquina le falta una o más de las letras de eje, se vuelve un poco más complicado. Los problemas de las letras de eje omitidas se tratan utilizando el parámetro *coordinates*= con el módulo trivkins. Los números de articulaciones son asignados consecutivamente a cada coordenada especificada. Se puede describir un torno con *coordinates=xz* Las asignaciones de articulacions serán entonces:

```
joints[0] = pos->tran.x
joints[1] = pos->tran.z
```

Se recomienda el uso del parámetro *coordinates*= para configuraciones que omiten letras del eje. [3]

El módulo cinemático *trivkins* también permite especificar la misma coordenada para más de una articulación. Esta característica puede ser útil en máquinas como un pórtico de dos motores independientes para la coordenada y. Tal máquina podría usar *coordenadas=xyyz* que da como resultado asignaciones de articulaciones:

```
joints[0] = pos->tran.x
joints[1] = pos->tran.y
joints[2] = pos->tran.y
joints[3] = pos->tran.z
```

Consulte la página man de trivkins para obtener más información.

[3] Históricamente, el módulo trivkins no contemplaba el parámetro *coordinates*= por lo que las configuraciones de torno a menudo se configuraran como máquinas XYZ. El eje Y no utilizado se configuró para que 1) hiciera home inmediatamente, 2) usara un lazo de realimentacion simple para conectar su pin HAL de comando de posición Hal a su pin HAL de retroalimentacion de posición, y 3) estaba oculto en las GUI de pantalla. Numerosas configuraciones sim usaron estos métodos para compartir archivos hal comunes.

15.3. Cinemática no trivial

Puede haber bastantes tipos de configuraciones de máquina (robots: puma, scara; hexápodos, etc.). Cada uno de ellos se configura utilizando articulacions lineales y giratorias. Estas articulaciones generalmente no coinciden con las coordenadas cartesianas y, por lo tanto, necesitamos una función cinemática que haga la conversión (en realidad 2 funciones: cinemática directa e inversa).

Para ilustrar lo anterior, analizaremos una maquina de cinemática simple llamada bípode (una versión simplificada del trípode, que es una version simplificada del hexápodo).

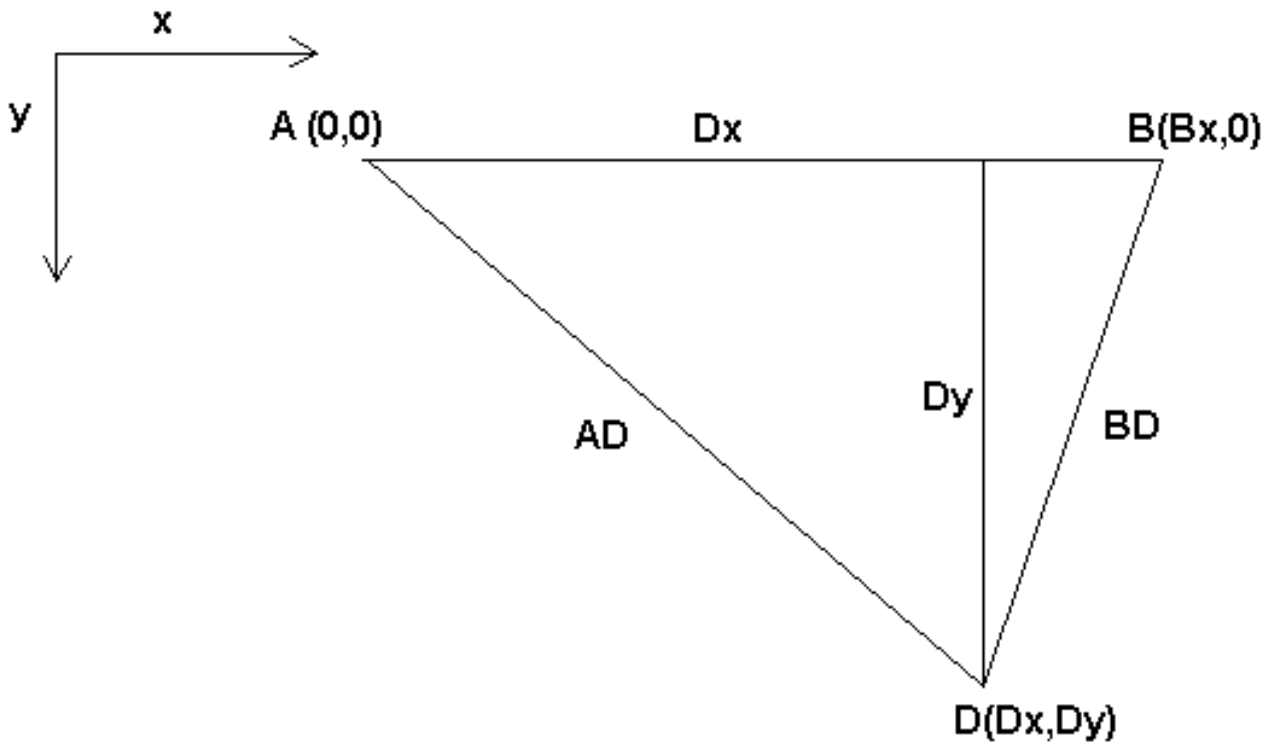


Figura 15.1: Configuración del bípode

El Bipod del que estamos hablando es un dispositivo que consta de 2 motores colocado en una pared, de la cual cuelga un dispositivo usando un cable. Las articulaciones en este caso son las distancias desde los motores al dispositivo (llamadas AD y BD en la figura).

La posición de los motores está fijada por convención. El motor A está en (0,0), lo que significa que su coordenada X es 0, y su coordenada Y es también 0. El motor B se coloca en (Bx, 0), lo que significa que su coordenada X es Bx (y a la misma altura Y que el otro motor).

Nuestra herramienta estará en el punto D que se define por las distancias AD y BD, y por las coordenadas cartesianas Dx, Dy.

El trabajo de la cinemática es transformar a partir de longitudes de articulaciones (AD, BD) a coordenadas cartesianas (Dx, Dy) y viceversa.

15.3.1. Transformación directa

Para transformar del espacio de articulaciones al espacio cartesiano utilizaremos algunas reglas de trigonometría (los triángulos rectángulos determinados por los puntos (0,0), (Dx, 0), (Dx, Dy) y el triángulo (Dx, 0), (Bx, 0) y (Dx, Dy).

Podemos ver fácilmente que:

$$AD^2 = x^2 + y^2 \; BD^2 = (Bx - x)^2 + y^2$$

asi como:

$$BD^2 = (Bx - x)^2 + y^2$$

Si restamos una de la otra obtendremos:

$$AD^2 - BD^2 = x^2 + y^2 - x^2 + 2 * x * Bx - Bx^2 - y^2$$

y por lo tanto:

$$x = \frac{AD^2 - BD^2 + Bx^2}{2 * Bx}$$

A partir de ahí calculamos:

$$y = \sqrt{AD2 - x2}$$

Tenga en cuenta que el cálculo para y implica la raíz cuadrada de una diferencia, que puede dar como resultado un número no real. Si no hay una sola coordenada cartesiana para esta posición de articulacion, la posición se dice que es una singularidad. En este caso, la cinemática directa retorna -1.

Traducido al código actual:

```
double AD2 = joints[0] * joints[0];
double BD2 = joints[1] * joints[1];
double x = (AD2 - BD2 + Bx * Bx) / (2 * Bx);
double y2 = AD2 - x * x;
if(y2 < 0) return -1;
pos->tran.x = x;
pos->tran.y = sqrt(y2);
return 0;
```

15.3.2. Transformación inversa

La cinemática inversa es mucho más fácil en nuestro ejemplo, ya que podemos escribir directamente

$$AD = \sqrt{x^2 + y^2}$$

$$BD = \sqrt{(Bx - x)^2 + y^2}$$

o traducido al código real:

```
double x2 = pos->tran.x * pos->tran.x;
double y2 = pos->tran.y * pos->tran.y;
joints[0] = sqrt(x2 + y2);
joints[1] = sqrt((Bx - pos->tran.x)*(Bx - pos->tran.x) + y2);
return 0;
```

15.4. Detalles de implementación

Un módulo cinemático se implementa como un componente HAL, y tiene permitido exportar pines y parámetros. Consiste en varias funciones "C" (a diferencia de las funciones HAL):

```
int kinematicsForward(const double *joint, EmcPose *world,
const KINEMATICS_FORWARD_FLAGS *fflags,
KINEMATICS_INVERSE_FLAGS *iflags)
```

Implementa la función cinemática directa.

```
int kinematicsInverse(const EmcPose * world, double *joints,
const KINEMATICS_INVERSE_FLAGS *iflags,
KINEMATICS_FORWARD_FLAGS *fflags)
```

Implementa la función cinemática inversa.

```
KINEMATICS_TYPE kinematicsType(void)
```

Devuelve el identificador de tipo de cinemática, típicamente *KINEMATICS_BOTH*.

```
int kinematicsHome(EmcPose *world, double *joint,
KINEMATICS_FORWARD_FLAGS *fflags,
KINEMATICS_INVERSE_FLAGS *iflags)
```

La función cinemática home establece todos sus argumentos a valores de la posición home conocida. Cuando se llama, estos deben establecerse, cuando se conocen, a valores iniciales de, por ejemplo, un archivo INI. Si la cinematica de home puede aceptar puntos de partida arbitrarios, estos valores iniciales deberían ser usados.

```
int rtapi_app_main(void)
void rtapi_app_exit(void)
```

Estas son las funciones estándar de configuración y desmontaje de los módulos RTAPI.

Cuando están contenidos en un solo archivo fuente, los módulos cinemáticos pueden ser compilados e instalados por *halcompile*. Consulte la página de manual de *halcompile(1)* o el manual de HAL para más información.

Capítulo 16

Configuración de parámetros "modificados" Denavi Hartenberg (DH) para

genserkins

16.1. Preludio

LinuxCNC admite varios módulos cinemáticos, incluido uno que admite un conjunto generalizado de cinemáticas serie comúnmente especificado a través de parámetros Denavit-Hartenberg.

Este documento ilustra un método para configurar los parámetros DH para un robot Mitsubishi RV-6SDL en LinuxCNC utilizando la cinemática *genserkins*.

nota
Este documento no cubre la creación de un modelo *vismach* que, aunque ciertamente es muy útil, requiere un modelado igual de cuidadoso para que coincida con el modelo *genserkins* derivado en este documento.

nota
Puede haber errores y/o defectos: ¡use bajo su propio riesgo!

16.2. General

Con la proliferación de robots industriales ha aumentado el interés por controlar robots usados con LinuxCNC. Un tipo común de robot utilizado en la industria y la fabricación es el "manipulador en serie" diseñado como una serie de articulaciones motorizadas conectadas por eslabones rígidos. Los robots en serie a menudo tienen seis articulaciones requeridas para los seis grados de libertad necesarios tanto para posicionar (XYZ) como para orientar (ABC o pitch, roll, yaw) un objeto en el espacio. A menudo, estos robots tienen una estructura de brazo que se extiende desde una base hasta un efector final.

El control de dicho robot en serie requiere el cálculo de la posición y orientación del efector final en relación con un sistema de coordenadas de referencia cuando se conocen los ángulos articulares (* cinemática directa*) y también el cálculo inverso, más complejo, de los ángulos articulares requeridos para una posición de efector final dada y la orientación en relación con el sistema de coordenadas de referencia (* cinematica inversa *). Las herramientas matemáticas estándar utilizadas para estos cálculos son matrices que son, básicamente, tablas de parámetros y fórmulas que facilitan el manejo de rotaciones y traslaciones involucrado en cálculos de cinemática directa e inversa.

No es necesaria familiaridad detallada con las matemáticas para un robot en serie ya que LinuxCNC proporciona un módulo de cinemática que implementa un algoritmo llamado *genserkins* para calcular cinemática directa e inversa para un robot serie genérico. Para controlar un robot serie específico, *genserkins* debe contar con datos para que pueda construir un modelo matemático de la estructura mecánica del robot y así hacer las matemáticas.

Los datos requeridos deben estar en una forma estandarizada, introducida por Jacques Denavit y Richard Hartenberg en los años cincuenta y se llaman parámetros DH. Denavit y Hartenberg utilizan cuatro parámetros para describir cómo se vincula una articulación con la siguiente. Estos parámetros describen básicamente dos rotaciones (*alfa* y *theta*) y dos translaciones (*a* y *d*).

16.3. Parámetros DH modificados

Como suele ser el caso, este "estándar" ha sido modificado por otros autores que han introducido "parámetros DH modificados" y se debe ser muy cuidadoso porque *genserkins* usa "parámetros DH modificados" como los descritos en la publicación "Introducción a la robótica, mecánica y Control" de John J. Craig. Tenga en cuenta que se puede encontrar mucha información sobre los *parámetros DH*, pero rara vez el autor define qué convención se usa realmente. Además, algunas personas han encontrado necesario cambiar el parámetro llamado *a* a *r* y así hemos agregado mas confusión Este documento se adhiere a la convención en la publicación mencionada anteriormente por Craig con la diferencia de que la enumeración de parámetros y articulaciones comienza con el número *0* para ser consistente con *genserkins* y sus pines HAL.

Los parámetros DH estándar y modificados constan de cuatro valores numéricos para cada articulación (*a*, *d*, *alpha* y *theta*) que describen cómo el sistema de coordenadas (CS) asociado a una articulación tiene que ser movido y girado para alinearse con la siguiente articulación. Alineado significa que el el eje Z de nuestro CS coincide con el eje de rotación de la articulación y apunta en la dirección positiva de modo que, usando la regla de la mano derecha con el pulgar apuntando en la dirección positiva del eje Z, los dedos apuntan en la dirección positiva de rotación de la articulación. Queda claro que para hacer esto, uno debe decidir sobre las direcciones positivas de todas las articulaciones antes de comenzar a derivar parámetros!

La diferencia entre notaciones "estándar" y "modificadas" está en cómo se asignan los parámetros a los enlaces. Usando los parámetros DH "estándar" en *genserkins* **no** dará el modelo matemático correcto.

16.4. Parámetros DH modificados usados en *genserkins*

Tenga en cuenta que *genserkins* no maneja offsets de valores theta - theta es la variable articular que es **controlada** por LinuxCNC. Con el CS alineado con la articulación, una rotación alrededor de su eje Z es idéntica a la rotación ordenada a esa articulación por LinuxCNC. Esto hace imposible definir la posición 0° de nuestras articulaciones de robots de forma arbitraria.

Los tres parámetros configurables son:

1. **alpha**: rotación positiva o negativa (en radianes) alrededor del eje X del "sistema de coordenadas actual"
2. **a**: distancia positiva, a lo largo de X, entre dos ejes de articulacion especificados en *unidades máquina* (mm o pulgadas) definidas en el archivo ini del sistema.
3. **d**: longitud positiva o negativa a lo largo de Z (también en *unidades máquina*)

Los conjuntos de parámetros siempre se derivan en el mismo orden y un conjunto es completado configurando el parámetro d. Esto no deja el eje Z de nuestro CS alineado con la próxima articulación! Esto puede parecer confuso pero apegarse a esta regla producirá un conjunto de parámetros de trabajo. Una vez el parámetro **d** está configurado, el eje X de nuestro CS necesita apuntar al eje de la siguiente articulación.

16.5. Numeración de articulaciones y parámetros

La primera articulación en LinuxCNC es joint-0 (porque en el software el conteo comienza con 0) mientras que la mayoría de las publicaciones comienzan con el número *1*. Eso también se aplica a todos los parámetros. Es decir, la numeración comienza con a-0, alpha-0, d-0 y termina con a-5, alpha-5 y d-5. Mantener esto en mente al seguir una publicación para configurar parámetros *genserkins*.

16.6. Cómo comenzar

La convención debe comenzar colocando el CS de referencia en la base del robot con su eje Z coincidiendo con el eje de la primera articulación y su eje X apuntando hacia el siguiente eje de articulación.

Esto también dará como resultado que los valores de DRO en LinuxCNC haga referencia a a ese punto. Una vez hecho esto, establece a-0 y alfa-0 en 0. Lo anterior La publicación mencionada (Craig) también establece d-0 a 0, lo cual es confuso cuando se necesita un offset de desplazamiento para tener la referencia-CS en la parte inferior de la base. Establecer d-0 = al desplazamiento da resultados correctos. De esta manera, el primer conjunto de parámetros son alpha-0 = 0, a-0 = 0, d0 = desplazamiento y el eje X de CS apuntando al eje de la siguiente articulación (articulación-1).

Sigue la derivación del conjunto neto (alpha-1, a-1, d-1) - siempre usando la misma secuencia hasta el sexto conjunto (alpha-5, a-5, d-5).

Y así, el TCP-CS del efector final está sentado en el centro de la mano.

16.7. Casos especiales

Si el siguiente eje articular es paralelo al último, entonces uno podría elejir arbitrariamente un valor para el parámetro d pero no hay apunte a establecerlo de otra manera que 0.

16.8. Ejemplo detallado (RV-6SL)

A continuación se describe un método para derivar los "Parámetros DH modificados" para un Mitsubishi RV-6SDL y cómo configurar los parámetros en el archivo HAL que se utilizará con la cinemática *genserkins* en LinuxCNC. Las dimensiones necesarias se toman mejor de un dibujo dimensional proporcionado por el fabricante del robot.

imagen::rv-6sl/rv-6sl-001.jpg[align="center"]

imagen::rv-6sl/rv-6sl-002.jpg[align="center"]

imagen::rv-6sl/rv-6sl-003.jpg[align="center"]

imagen::rv-6sl/rv-6sl-004.jpg[align="center"]

imagen::rv-6sl/rv-6sl-005.jpg[align="center"]

imagen::rv-6sl/rv-6sl-006.jpg[align="center"]

imagen::rv-6sl/rv-6sl-007.jpg[align="center"]

imagen::rv-6sl/rv-6sl-008.jpg[align="center"]

imagen::rv-6sl/rv-6sl-009.jpg[align="center"]

imagen::rv-6sl/rv-6sl-010.jpg[align="center"]

imagen::rv-6sl/rv-6sl-011.jpg[align="center"]

imagen::rv-6sl/rv-6sl-012.jpg[align="center"]

imagen::rv-6sl/rv-6sl-013.jpg[align="center"]

imagen::rv-6sl/rv-6sl-014.jpg[align="center"]

imagen::rv-6sl/rv-6sl-015.jpg[align="center"]

imagen::rv-6sl/rv-6sl-016.jpg[align="center"]

imagen::rv-6sl/rv-6sl-017.jpg[align="center"]

imagen::rv-6sl/rv-6sl-018.jpg[align="center"]

16.9. Créditos

Gracias al usuario Aciera por todo el texto y los gráficos. para el robot RV-6SL!

Capítulo 17

Cinemática de 5 ejes

17.1. Introducción

Las máquinas herramienta con multiejes coordinados CNC controladas con LinuxCNC requieren un componente cinemático especial para cada tipo de máquina. Este capítulo describe algunas de las configuraciones de máquina de 5 ejes más populares y luego desarrolla las transformaciones directas (desde el trabajo hasta las coordenadas de articulacion) e inversa (desde la articulacion hasta el trabajo) en un proceso matemático general para dos tipos de máquina.

Se proporcionan los componentes cinemáticos y los modelos de simulación de vismach para demostrar su comportamiento en la pantalla de la computadora. También se dan ejemplos de datos de archivos HAL.

17.2. Configuraciones de máquinas herramienta de 5 ejes

En esta sección tratamos con máquinas de fresado o enrutadores típicos de 5 ejes, con cinco articulaciones o grados de libertad, que se controlan en base a movimientos coordinados.

Las máquinas herramienta de 3 ejes no pueden cambiar la orientación de la herramienta. Las máquinas herramienta de 5 ejes utilizan dos ejes adicionales para dotar a la herramienta de corte de una orientación adecuada para el mecanizado eficiente de superficies de cualquier forma.

Las configuraciones típicas de máquinas herramienta de 5 ejes se muestran en las figs. 3, 5, 7 y 9-11 [1,2] en la sección Figuras.

La cinemática de las máquinas herramienta de 5 ejes es mucho más simple que la de los brazos robots de 6 ejes, puesto que 3 de los ejes son normalmente ejes lineales y solo dos son ejes giratorios.

17.3. Orientación y localizacion de la herramienta

Generalmente se utilizan sistemas CAD/CAM para generar los modelos 3D CAD de la pieza de trabajo, así como los datos CAM para la entrada a la máquina CNC de 5 ejes. La herramienta o los datos de localizacion de la herramienta (CL, Cutter Location) se componen de la posición de la punta de la herramienta y la orientación de la misma en relación con el sistema de coordenadas de la pieza. Esa informacion la contienen dos vectores, como los generados por la mayoría de los sistemas CAM, y que se muestran en la Fig. 1,:

$$K = \begin{bmatrix} K_x \\ K_y \\ K_z \\ 0 \end{bmatrix} \quad \text{orientation vector;} \qquad Q = \begin{bmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{bmatrix} \quad \text{position vector} \qquad (1)$$

El vector K es equivalente al 3° vector de la matriz de pose E_6 que se usó en la cinemática del robot de 6 ejes en [3] y el vector Q es equivalente al 4° vector de E_6. En MASTERCAM, por ejemplo, esta información está contenida en el archivo de salida intermedio ".nci".

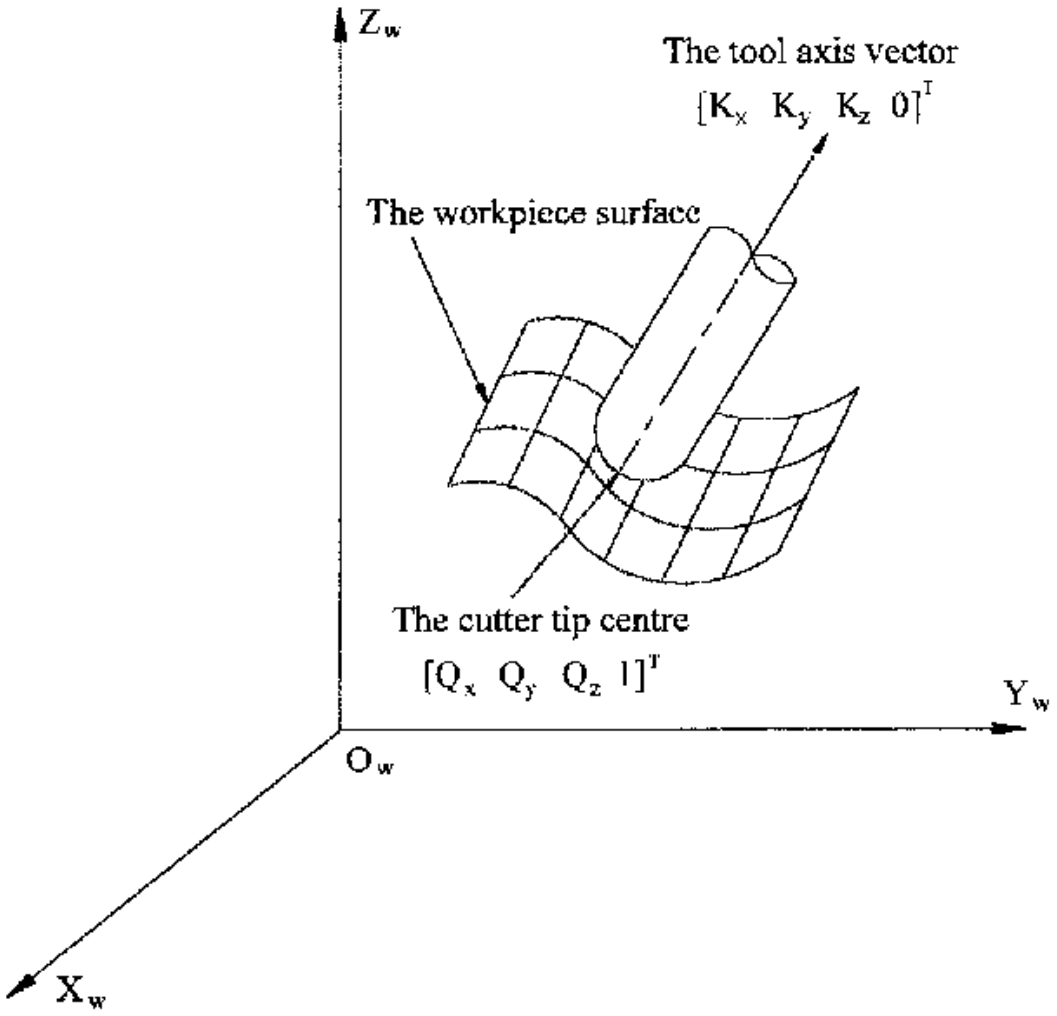


Figura 17.1: Datos de localizacion de la herramienta

17.4. Matrices de translacion y rotación

Las transformaciones homogéneas proporcionan una manera simple de describir las matemáticas de la cinemática de máquinas multieje. Una transformación del espacio H es una matriz 4x4 y puede representar transformaciones de translacion y rotación. Dado un punto x, y, z descrito por un vector u={x, y, z, 1}^{T}, su transformación v está representada por la matriz producto.

$$v = H \cdot u$$

Hay cuatro matrices de transformación fundamentales en las que se basa la cinemática de 5 ejes:

$$T(a,b,c) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R(X,\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\theta & -S\theta & 0 \\ 0 & S\theta & C\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

$$R(Y,\theta) = \begin{bmatrix} C\theta & 0 & S\theta & 0 \\ 0 & 1 & 0 & 0 \\ -S\theta & 0 & C\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R(Z,\theta) = \begin{bmatrix} C\theta & -S\theta & 0 & 0 \\ S\theta & C\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

La matriz T(a,b,c) implica una traslación en las direcciones de coordenadas X, Y y Z, en las cantidades a, b y c respectivamente. Las matrices R implican rotaciones del ángulo theta alrededor de los ejes de coordenadas X, Y y Z, respectivamente. Los símbolos *C* y *S* se refieren a las funciones coseno y seno, respectivamente.

17.5. Configuraciones de 5 ejes con mesas giratorias/inclinables

En estas máquinas herramienta, los dos ejes giratorios se montan en la mesa de trabajo de la máquina. Se usan tipicamente dos formas:

- Una mesa giratoria que gira alrededor del eje Z vertical (rotación C, secundaria) montado en una mesa basculante que gira alrededor del eje X o Y (rotación A o B, primaria). La pieza de trabajo está montada en la mesa giratoria.
- Una mesa inclinable que gira alrededor del eje X o Y (rotación A o B, secundaria) está montada en una mesa giratoria que gira alrededor del eje Z (rotación C, primaria), con la pieza de trabajo en el mesa basculante.

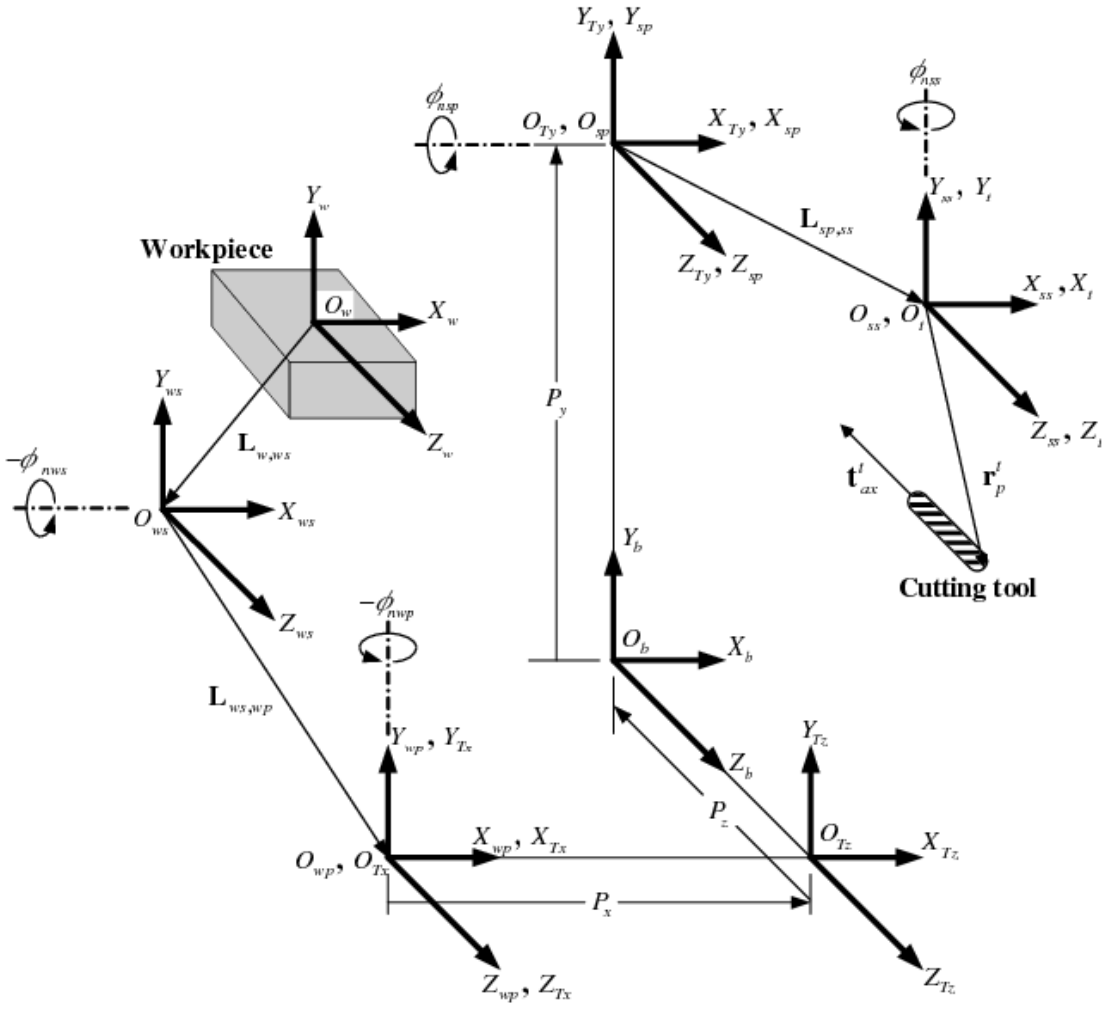


Figura 17.2: Configuración general y sistemas de coordenadas

Se puede considerar que una máquina multieje consiste en una serie de eslabones conectados por articulaciones. Al asignar un sistema de coordenadas en cada eslabon de la máquina, y usando transformaciones homogéneas, podemos describir la posición relativa y la orientación entre estos sistemas de coordenadas.

Necesitamos describir una relación entre el sistema de coordenadas de la pieza de trabajo y el sistema de coordenadas de la herramienta. Esto se puede definir mediante una matriz de transformación ${}^{w}A_t$, que se puede encontrar mediante transformaciones sucesivas entre los diferentes elementos estructurales o eslabones de la máquina, cada uno con su propio sistema de coordenadas definido. En general, tal transformación puede verse de la siguiente manera:

$$ {}^{w}A_t = {}^{w}A_1 \cdot {}^{1}A_2 \cdot {}^{2}A_3 \cdots {}^{n}A_t \qquad (4) $$

donde cada matriz ${}^{i-1}A_j$ es una matriz de traslacion T o una matriz de rotación R de la forma (2),(3).

La multiplicación de matrices es un proceso simple en el que los elementos de cada fila de la matriz A de la izquierda se multiplican por los elementos de cada columna de la matriz B de la derecha y se suman para obtener un elemento en la matriz resultante C, es decir.

$$ C_{ij} = \sum_{k=1,n}^{n} A_{ik} B_{kj}; \quad i = 1, n; \quad j = 1, n $$

En la Fig. 2 se muestra una configuración genérica con sistemas de coordenadas [4]. Incluye ejes giratorios/basculantes en mesa o en husillo. En una máquina herramienta se usan realmente solo dos de los tres ejes giratorios.

Primero desarrollaremos las transformaciones para el primer tipo de configuración mencionado anteriormente, es decir. un tipo de mesa inclinable/giratoria (la llamaremos "trt") sin offsets de ejes giratorios. Podemos darle el nombre de configuración xyzac-trt.

También desarrollamos las transformaciones para el mismo tipo (xyzac-trt), pero con offsets de eje giratorios.

Luego desarrollamos las transformaciones para una configuración xyzbc-trt con offsets de ejes giratorios.

17.5.1. Transformaciones para una máquina herramienta xyzac-trt con offsets de pieza de trabajo

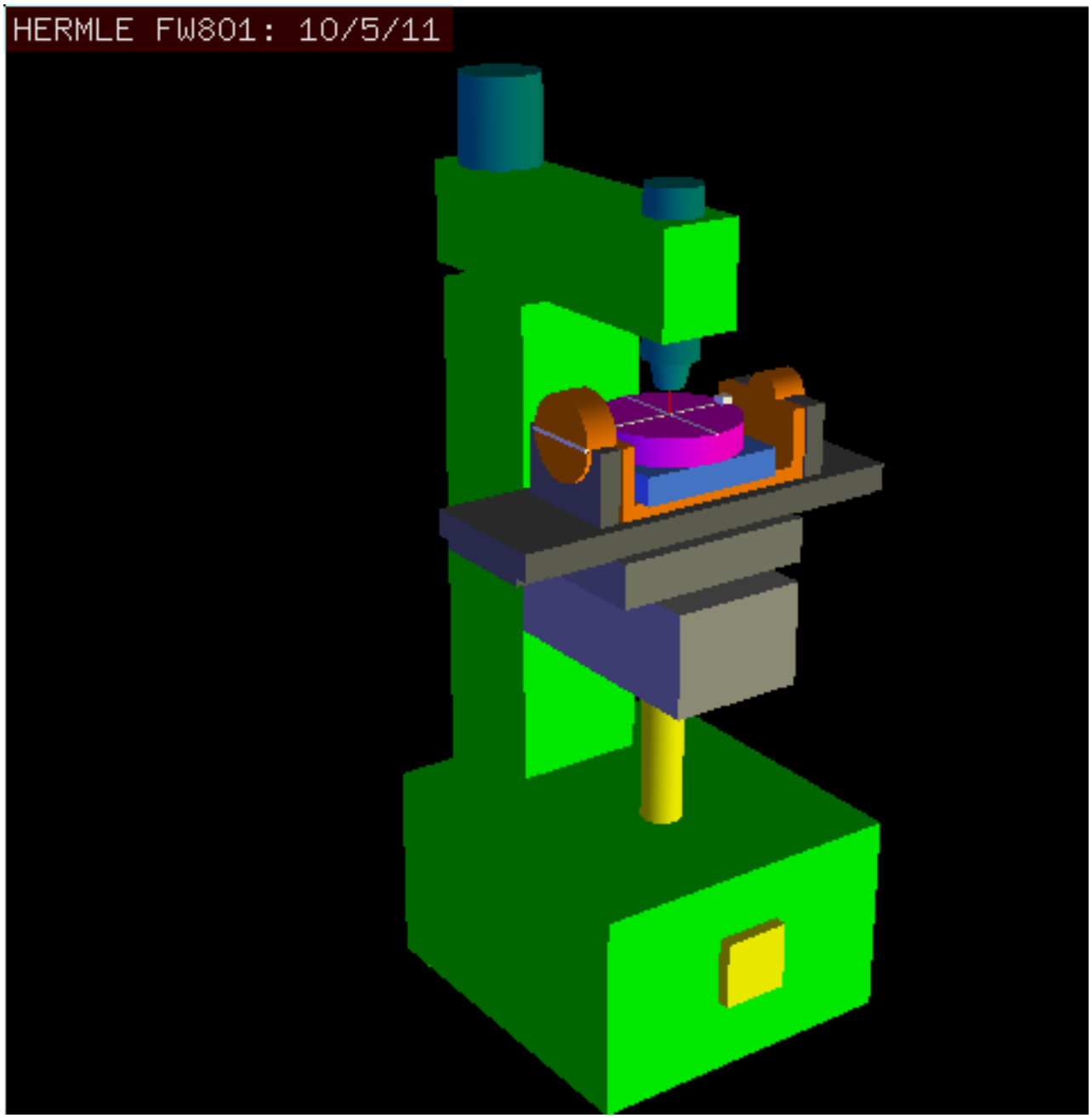


Figura 17.3: Modelo vismach de xyzac-trt con ejes de rotación coincidentes

Tratamos aquí con una configuración simplificada en la que el eje basculante y el eje giratorio se cruzan en un punto llamado punto pivote como se muestra en la figura 4. Por lo tanto, los dos sistemas de coordenadas O_{ws} y O_{wp} de la fig. 2 son coincidentes.

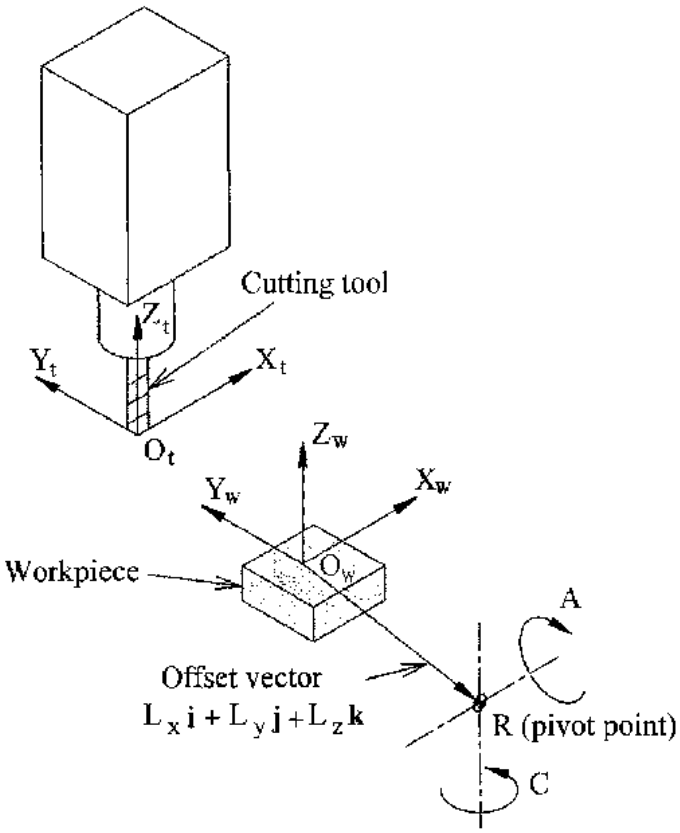


Figura 17.4: Configuración mesa inclinable/giratoria

17.5.1.1. Transformación directa

La transformación puede definirse por la multiplicación secuencial de las matrices:

$$^{w}A_t = {}^{w}A_C \cdot {}^{C}A_A \cdot {}^{A}A_P \cdot {}^{P}A_t \qquad (5)$$

con las matrices construidas de la siguiente manera:

$$ {}^{w}A_C = \begin{bmatrix} 1 & 0 & 0 & L_x \\ 0 & 1 & 0 & L_y \\ 0 & 0 & 1 & L_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{C}A_A = \begin{bmatrix} C_C & S_C & 0 & 0 \\ -S_C & C_C & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (6) $$

$$ {}^{A}A_P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C_A & S_A & 0 \\ 0 & -S_A & C_A & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{P}A_t = \begin{bmatrix} 1 & 0 & 0 & P_x \\ 0 & 1 & 0 & P_y \\ 0 & 0 & 1 & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (7) $$

En estas ecuaciones, L_x, L_y, L_z define los desplazamientos del punto pivote de los dos ejes giratorios A y C con relación al origen del sistema de coordenadas de la pieza. Además, P_x, P_y, P_z son las distancias relativas del punto pivote a la posición de la punta de la herramienta, que también se pueden llamar las "coordenadas de articulacion" del punto pivote. El punto pivote está en la intersección de los dos ejes giratorios. Los signos de los términos S_A y S_C son diferentes a los de [2,3] dado que las rotaciones de la tabla son negativas con respecto a los ejes de coordenadas de la pieza de trabajo (tenga en cuenta que sen(-theta) = -sen(theta)), cos(-theta) = cos(theta)).

Cuando se multiplica de acuerdo con (5), obtenemos:

$$ {}^{w}A_t = \begin{bmatrix} C_C & S_C C_A & S_C S_A & C_C P_x + S_C C_A P_y + S_C S_A P_z + L_x \\ -S_C & C_C C_A & C_C S_A & -S_C P_x + C_C C_A P_y + C_C S_A P_z + L_y \\ 0 & -S_A & C_A & -S_A P_y + C_A P_z + L_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (8) $$

Ahora podemos equiparar la tercera columna de esta matriz con nuestro vector de orientación de herramienta K, es decir:

$$ K = \begin{bmatrix} K_x \\ K_y \\ K_z \\ 0 \end{bmatrix} = \begin{bmatrix} S_C S_A \\ C_C S_A \\ C_A \\ 0 \end{bmatrix} \qquad (9) $$

A partir de estas ecuaciones podemos resolver los ángulos de rotación $theta_A$, $theta_C$. De la tercera fila encontramos:

$$ \theta_A = cos^{-1}(K_z) \qquad (0 < \theta_A < \pi) \qquad (10) $$

y al dividir la primera fila por la segunda fila encontramos:

$$ \theta_C = tan2^{-1}(K_x, K_y) \qquad (-\pi < \theta_C < \pi) \qquad (11) $$

Estas relaciones se usan típicamente en el postprocesador CAM para convertir los vectores de orientación de la herramienta en ángulos de rotación.

Al igualar la última columna de (8) con el vector Q de posición de herramienta, podemos escribir:

$$ Q = \begin{bmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{bmatrix} = \begin{bmatrix} C_C P_x + S_C C_A P_y + S_C S_A P_z + L_x \\ -S_C P_x + C_C C_A P_y + C_C S_A P_z + L_y \\ -S_A P_y + C_A P_z + L_z \\ 1 \end{bmatrix} \qquad (12) $$

El vector en el lado derecho también se puede escribir como el producto de una matriz y un vector, que da como resultado:

$$ Q = \begin{bmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{bmatrix} = \begin{bmatrix} C_C & S_C C_A & S_C S_A & L_x \\ -S_C & C_C C_A & C_C S_A & L_y \\ 0 & -S_A & C_A & L_z \\ 0 & 0 & & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} = {}^{Q}A_P \cdot P \qquad (13) $$

Esto se puede expandir para dar

$$\begin{aligned} Q_x &= C_c P_x + S_C C_A P_y + S_C S_A P_z + L_x \\ Q_y &= -S_C P_x + C_C C_A P_y + C_C S_A P_z + L_y \\ Q_z &= -S_A P_y + C_A P_z + L_z \end{aligned} \qquad (14)$$

que es la "transformación directa" de la cinemática.

17.5.1.2. Transformación inversa

Podemos resolver para P de la ecuación (13) como $P = ({}^{Q}A_P)^{-1} * Q$. Observando la matriz cuadrada, es una matriz 4x4 homogénea que contiene una matriz de rotación R y un vector de translacion q, por lo que el inverso se puede escribir como:

$$ {}^{q}A_p = \begin{bmatrix} R & q \\ 0 & 1 \end{bmatrix} \qquad ({}^{Q}A_P)^{-1} = \begin{bmatrix} R^T & -R^T q \\ 0 & 1 \end{bmatrix} \qquad (15)$$

donde R^T es la transpuesta de R (filas y columnas intercambiadas). Por lo tanto, obtenemos:

$$\begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} = \begin{bmatrix} C_C & -S_C & 0 & -C_C L_x + S_C L_y \\ S_C C_A & C_C C_A & -S_A & -S_C C_A L_x - C_C C_A L_y + S_A L_z \\ S_C S_A & C_C S_A & C_A & -S_C S_A L_x - C_C S_A L_y - C_A L_z \\ 0 & 0 & & 1 \end{bmatrix} \begin{bmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{bmatrix} \qquad (16)$$

Las ecuaciones que se necesitan para la "transformación inversa" de la cinemática se pueden escribir así:

$$\begin{aligned} P_x &= C_c(Q_x - L_x) - S_C(Q_y - L_y) \\ P_y &= S_C C_A(Q_x - L_x) + C_C C_A(Q_y - L_y) - S_A(Q_z - L_z) \\ P_z &= S_C S_A(Q_x - L_x) + C_C S_A(Q_y - L_y) + C_A(Q_z - L_z) \end{aligned} \qquad (17)$$

17.5.2. Transformaciones para una máquina xyzac-trt con offsets en ejes rotativos

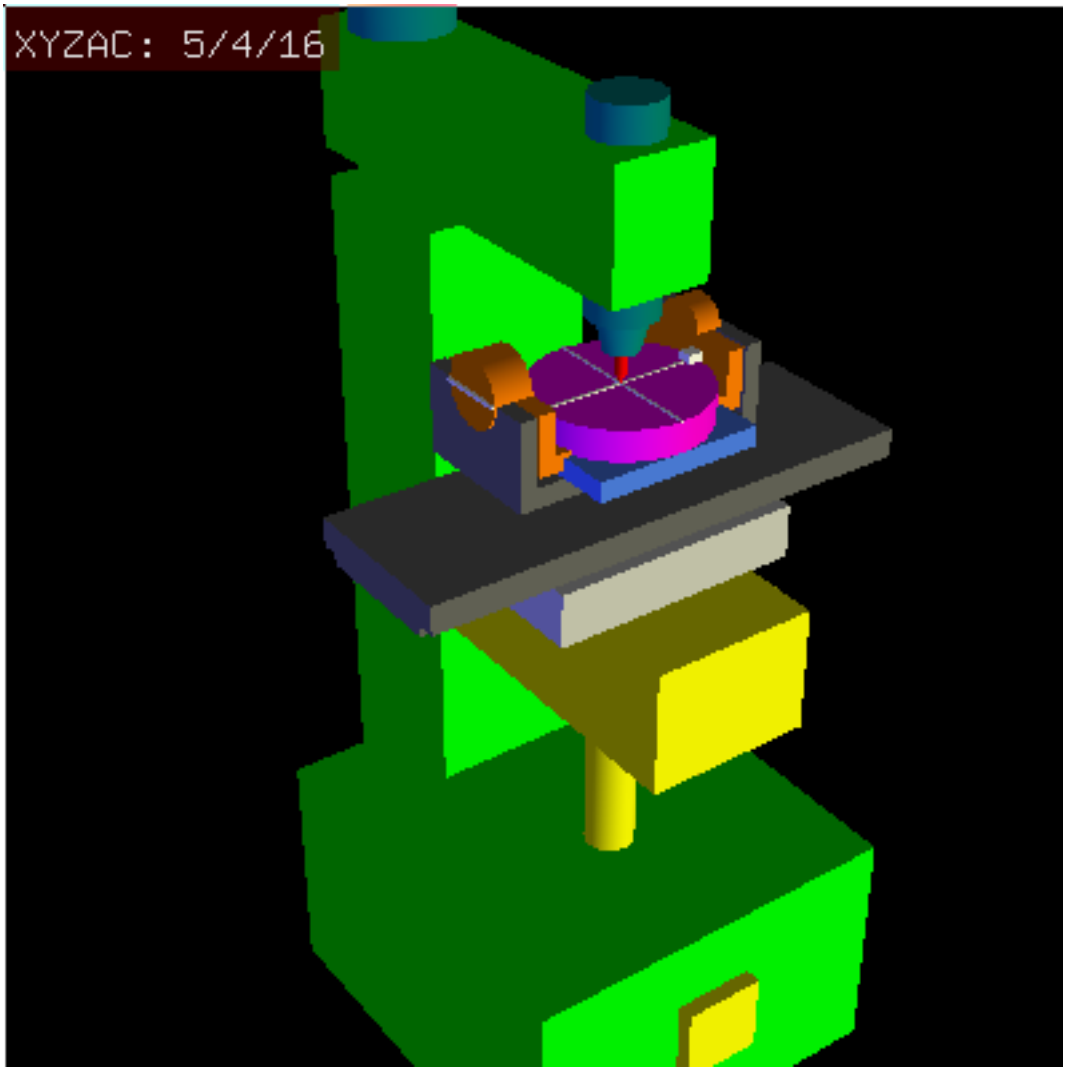


Figura 17.5: Modelo vismach de xyzac-trt con offsets con ejes rotatorios (positivos)

Tratamos aquí con una configuración extendida en la que el eje basculante y el eje rotativo no se cruzan en un punto, sino que tienen un desplazamiento D_y. Además, también hay un desplazamiento z entre los dos sistemas de coordenadas O_{ws} y O_{wp} de la figura 2, llamado D_z. En la fig. 5 se muestra un modelo vismach y los desplazamientos se muestran en la fig. 6 (desplazamientos positivos en este ejemplo). Para simplificar la configuración, los desplazamientos L_x, L_y, L_z del caso anterior no estan incluidos. Probablemente no sean necesarios si se usan los offsets G54 en LinuxCNC por medio de la función "touch of".

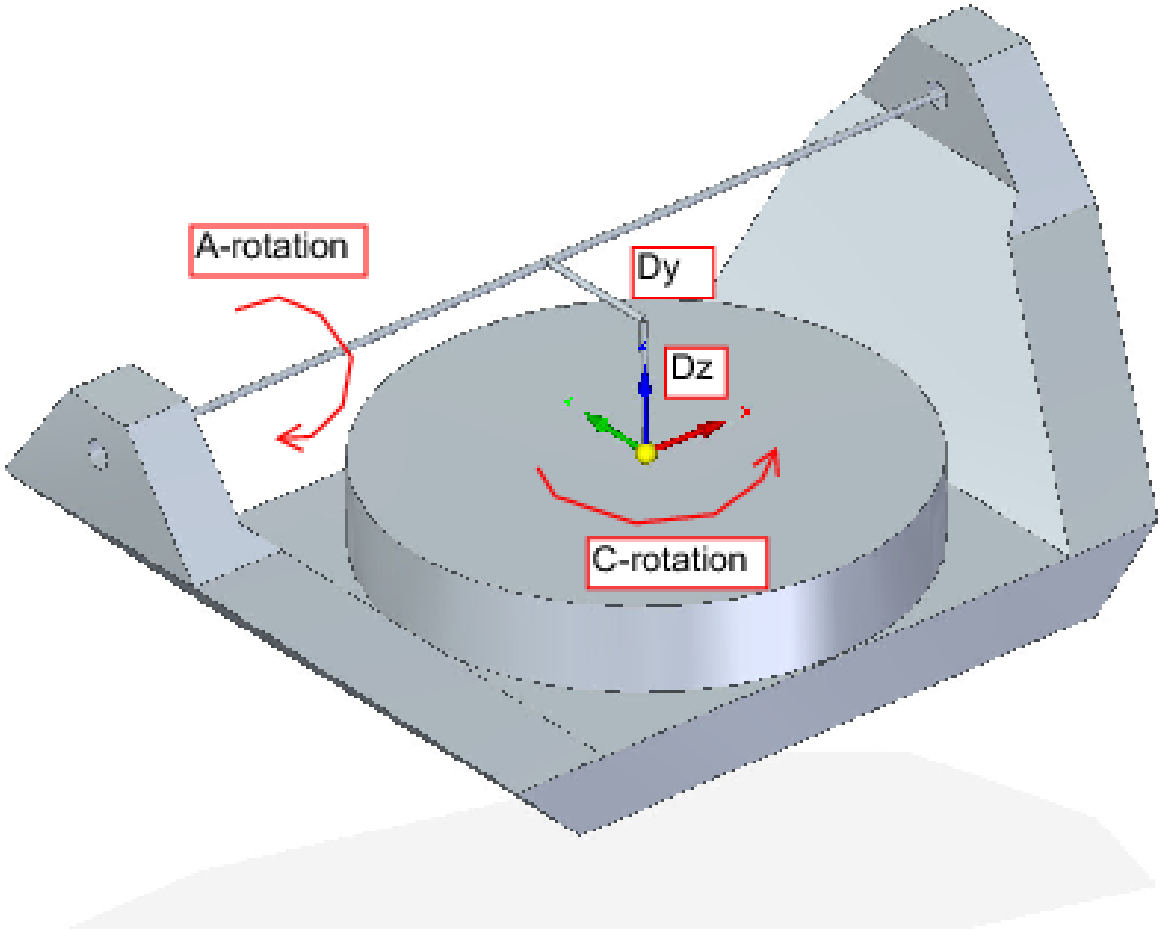


Figura 17.6: Configuración inclinable/giratoria xyzac-trt, con offsets de ejes

17.5.2.1. Transformación directa

La transformación puede definirse por la multiplicación secuencial de las matrices:

$$ {}^{w}A_t = {}^{w}A_O \cdot {}^{O}A_A \cdot {}^{A}A_P \cdot {}^{P}A_t \qquad (18) $$

con las matrices construidas de la siguiente manera:

$$ {}^{w}A_O = \begin{bmatrix} C_C & S_C & 0 & 0 \\ -S_C & C_C & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{O}A_A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & D_y \\ 0 & 0 & 1 & D_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (19) $$

$$ {}^{A}A_P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C_A & S_A & 0 \\ 0 & -S_A & C_A & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{P}A_t = \begin{bmatrix} 1 & 0 & 0 & P_x \\ 0 & 1 & 0 & P_y - D_y \\ 0 & 0 & 1 & P_z - D_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (20) $$

En estas ecuaciones, D_y y D_z definen los desplazamientos del punto pivote de los ejes rotativos A relativos al origen del sistema de coordenadas de la pieza. Además, P_x, P_y, P_z son las distancias relativas del punto pivote a la posición de la punta de la herramienta, que también se pueden llamar "coordenadas de articulacion" del punto pivote. El punto pivote está en el eje de rotación A.

Cuando se multiplica de acuerdo con (18), obtenemos:

$$ {}^{w}A_t = \begin{bmatrix} C_C & S_C C_A & S_C S_A & C_C P_x + S_C C_A(P_y - D_y) + S_C S_A(P_z - D_z) + S_C D_y \\ -S_C & C_C C_A & C_C S_A & -S_C P_x + C_C C_A(P_y - D_y) + C_C S_A(P_z - D_z) + C_C D_y \\ 0 & -S_A & C_A & -S_A(P_y - D_y) + C_A(P_z - D_z) + D_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (21) $$

Ahora podemos equiparar la tercera columna de esta matriz con nuestro vector de orientación de herramienta K, es decir:

$$ K = \begin{bmatrix} K_x \\ K_y \\ K_z \\ 0 \end{bmatrix} = \begin{bmatrix} S_C S_A \\ C_C S_A \\ C_A \\ 0 \end{bmatrix} \qquad (22) $$

A partir de estas ecuaciones podemos resolver los ángulos de rotación $theta_A$, $theta_C$. De la tercera fila encontramos:

$$ \theta_A = cos^{-1}(K_z) \qquad (0 < \theta_A < \pi) \qquad (23) $$

y al dividir la segunda fila por la primera fila encontramos:

$$ \theta_C = tan2^{-1}(K_x, K_y) \qquad (-\pi < \theta_C < \pi) \qquad (24) $$

Estas relaciones se usan típicamente en el postprocesador CAM para convertir los vectores de orientación de la herramienta en ángulos de rotación.

Al igualar la última columna de (21) con el vector de posición de herramienta Q, podemos escribir:

$$ Q = \begin{bmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{bmatrix} = \begin{bmatrix} C_C P_x + S_C C_A(P_y - D_y) + S_C S_A(P_z - D_z) + S_C D_y \\ -S_C P_x + C_C C_A(P_y - D_y) + C_C S_A(P_z - D_z) + C_C D_y \\ -S_A(P_y - D_y) + C_A(P_z - D_z) + D_z \\ 1 \end{bmatrix} \qquad (25) $$

El vector en el lado derecho también se puede escribir como el producto de una matriz y un vector, que da como resultado:

$$ Q = \begin{bmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{bmatrix} = \begin{bmatrix} C_C & S_C C_A & S_C S_A & -S_c C_A D_y - S_C S_A D_z + S_C D_y \\ -S_C & C_C C_A & C_C S_A & -C_C C_A D_y - C_C S_A D_z + C_C D_y \\ 0 & -S_A & C_A & S_A D_y - C_A D_z + D_z \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} = {}^{Q}A_P \cdot P \qquad (26) $$

que es la "transformación directa" de la cinemática.

17.5.2.2. Transformación inversa

Podemos resolver para P de la ecuación (25) como $P=({}^{Q}A_P)^{-1}*Q$ usando (15) como antes. Por lo tanto obtenemos:

$$ \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} = \begin{bmatrix} C_C & -S_C & 0 & 0 \\ S_C C_A & C_C C_A & -S_A & -C_A D_y + S_A D_z + D_y \\ S_C S_A & C_S S_A & C_A & -S_A D_y - C_A D_z + D_z \\ 0 & 0 & & 1 \end{bmatrix} \begin{bmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{bmatrix} \qquad (27) $$

Las ecuaciones que se necesitan para la "transformación inversa" de la cinemática se pueden escribir así:

$$ \begin{aligned} P_x &= C_c Q_x - S_C Q_y \\ P_y &= S_C C_A Q_x + C_C C_A Q_y - S_A Q_z - C_A D_y + S_A D_z + D_y \\ P_z &= S_C S_A Q_x + C_S S_A Q_y + C_A Q_z - S_A D_y - C_A D_z + D_z \end{aligned} \qquad (28) $$

17.5.3. Transformaciones para una máquina xyzbc-trt con offsets de ejes rotativos

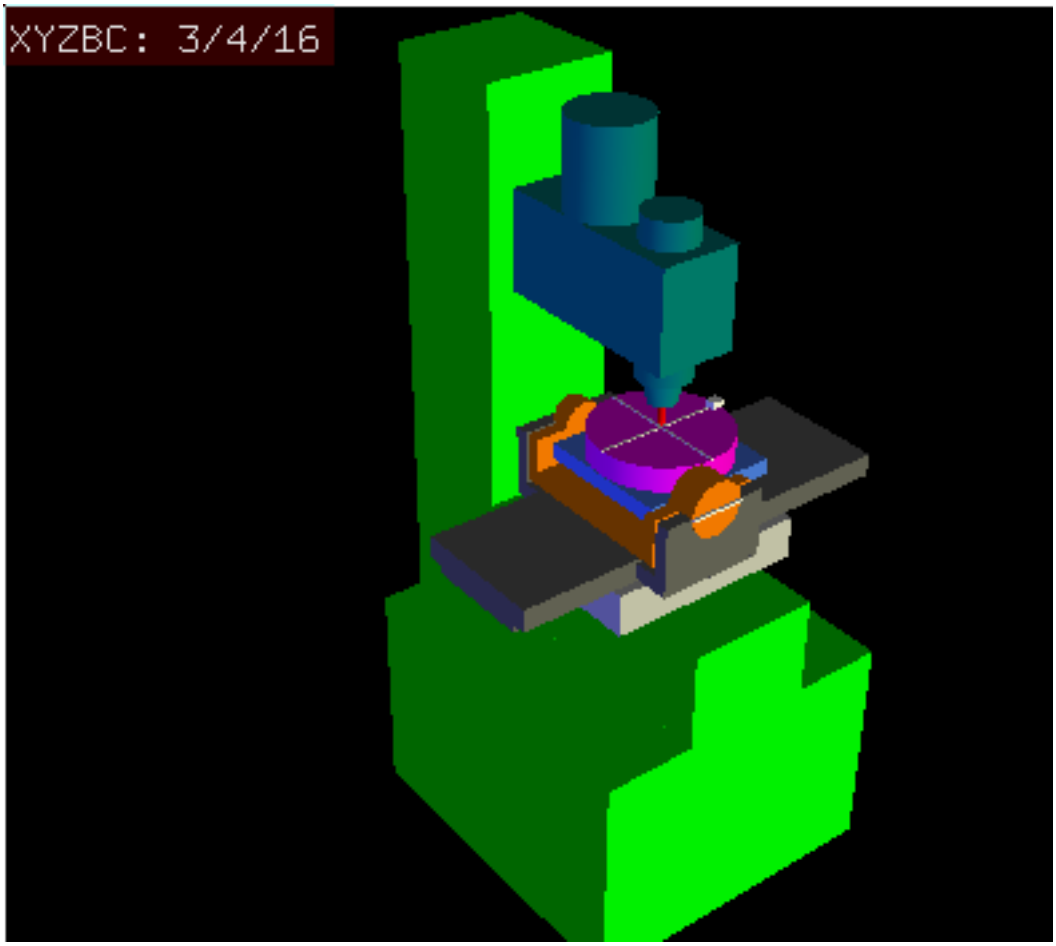


Figura 17.7: modelo vismach de xyzbc-trt con compensaciones de eje rotacional (negativo)

Tratamos aquí de nuevo con una configuración extendida en la que el eje de inclinación (alrededor del eje y) y el eje giratorio no se cruzan en un punto, pero tienen un desplazamiento D_x. Además, también hay un desplazamiento z entre los dos sistemas de coordenadas O_{ws} y O_{wp} de la figura 2, llamado D_z. Un modelo vismach se muestra en la fig. 7 (desplazamientos negativos en este ejemplo) y los desplazamientos positivos se muestran en la fig. 8.

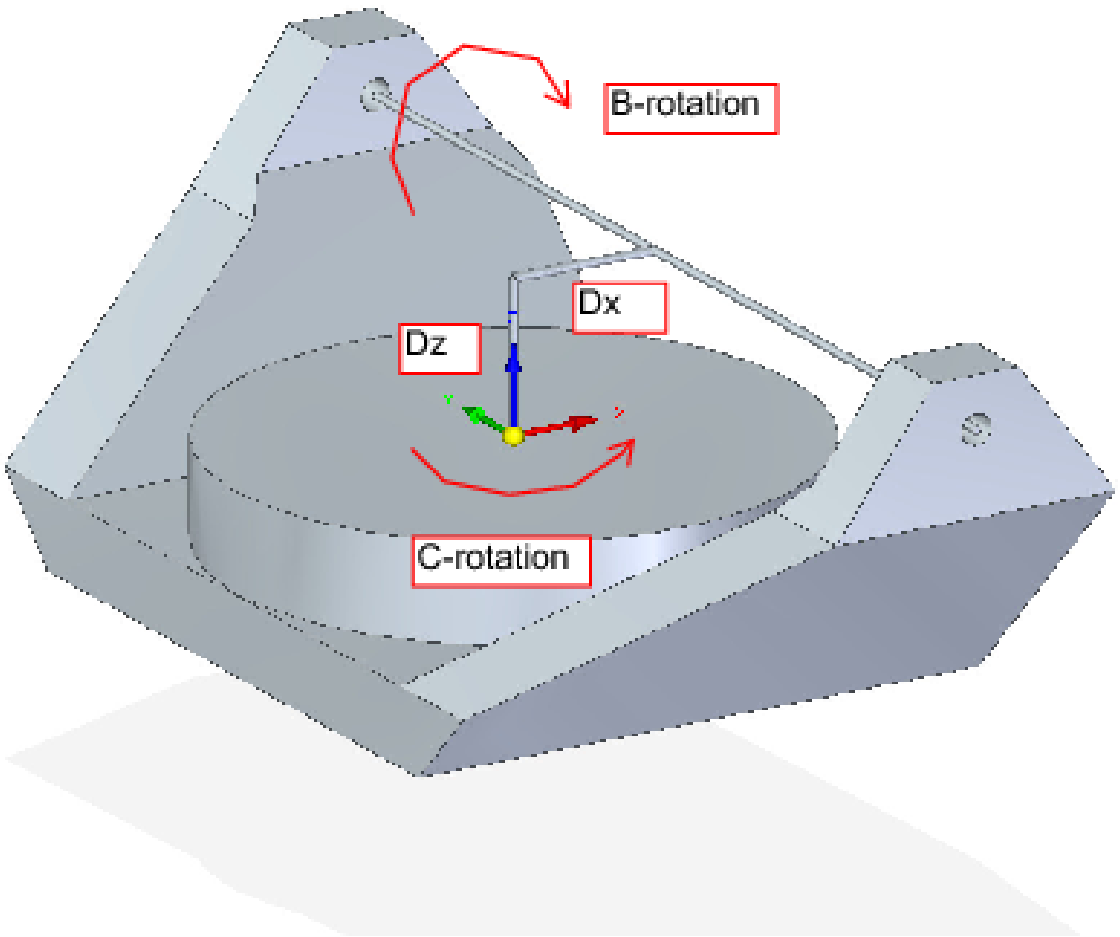


Figura 17.8: Configuración inclinable/giratoria xyzbc-trt, con offsets de ejes

17.5.3.1. Transformación directa

La transformación puede definirse por la multiplicación secuencial de las matrices:

$$ {}^{w}A_t = {}^{w}A_O \cdot {}^{O}A_B \cdot {}^{B}A_P \cdot {}^{P}A_t \qquad (29) $$

con las matrices construidas de la siguiente manera:

$$ {}^{w}A_O = \begin{bmatrix} C_C & S_C & 0 & 0 \\ -S_C & C_C & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{O}A_B = \begin{bmatrix} 1 & 0 & 0 & D_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & D_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (30) $$

$$ {}^{B}A_P = \begin{bmatrix} C_B & 0 & -S_B & 0 \\ 0 & 1 & 0 & 0 \\ S_B & 0 & C_B & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{P}A_t = \begin{bmatrix} 1 & 0 & 0 & P_x - D_x \\ 0 & 1 & 0 & P_y \\ 0 & 0 & 1 & P_z - D_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (31) $$

En estas ecuaciones D_x, D_z define los desplazamientos del punto pivote de los ejes giratorios B en relación con el origen del sistema de coordenadas de la pieza. Además, P_x, P_y, P_z son las distancias relativas del punto pivote a la posición de la punta de la herramienta, que también se pueden llamar las "coordenadas de articulacion" del punto pivote. El punto pivote está en el eje rotativo B.

Cuando se multiplica de acuerdo con (29), obtenemos:

$$ {}^{w}A_t = \begin{bmatrix} C_C C_B & S_C & -C_C S_B & C_C C_B(P_x - D_x) + S_C P_y - C_C S_B(P_z - D_z) + C_C D_x \\ -S_C C_B & C_C & S_C S_B & -S_C C_B(P_x - D_X) + C_C P_y + S_C S_B(P_z - D_z) - S_C D_x \\ S_B & 0 & C_B & S_B(P_x - D_x) + C_B(P_z - D_z) + D_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (32) $$

Ahora podemos equiparar la tercera columna de esta matriz con nuestro vector de orientación de herramienta K, es decir:

$$ K = \begin{bmatrix} K_x \\ K_y \\ K_z \\ 0 \end{bmatrix} = \begin{bmatrix} -C_C S_B \\ S_C S_B \\ C_B \\ 0 \end{bmatrix} \qquad (33) $$

A partir de estas ecuaciones podemos resolver los ángulos de rotación $theta_B$, $theta_C$. De la tercera fila encontramos:

$$ \theta_B = cos^{-1}(K_z) \qquad (0 < \theta_B < \pi) \qquad (34) $$

y al dividir la segunda fila por la primera fila encontramos:

$$ \theta_C = tan2^{-1}(K_y, K_x) \qquad (-\pi < \theta_C < \pi) \qquad (35) $$

Estas relaciones se usan típicamente en el postprocesador CAM para convertir los vectores de orientación de la herramienta en ángulos de rotación.

Al igualar la última columna de (32) con el vector de posición de herramienta Q, podemos escribir:

$$ Q = \begin{bmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{bmatrix} = \begin{bmatrix} C_C C_B(P_x - D_x) + S_C P_y - C_C S_B(P_z - D_z) + C_C D_x \\ -S_C C_B(P_x - D_X) + C_C P_y + S_C S_B(P_z - D_z) - S_C D_x \\ S_B(P_x - D_x) + C_B(P_z - D_z) + D_z \\ 1 \end{bmatrix} \qquad (36) $$

El vector en el lado derecho también se puede escribir como el producto de una matriz y un vector que da como resultado:

$$ Q = \begin{bmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{bmatrix} = \begin{bmatrix} C_C C_B & S_C & -C_C S_B & -C_c C_B D_x + C_C S_B D_z + C_C D_x \\ -S_C C_B & C_C & S_C S_B & S_C C_B D_x - S_C S_B D_z - S_C D_x \\ S_B & 0 & C_B & -S_B D_x - C_B D_z + D_z \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} = {}^{Q}A_P \cdot P \qquad (37) $$

que es la "transformación directa" de la cinemática.

17.5.3.2. Transformación inversa

Podemos resolver para P de la ecuación (37) como $P=({}^{Q}A_P)^{-1}*Q$. Con el mismo enfoque que antes, obtenemos:

$$ \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} = \begin{bmatrix} C_C C_B & -S_C C_B & S_B & -C_B D_x - S_B D_z + D_x \\ S_C & C_C & 0 & 0 \\ -C_C S_B & S_C S_B & C_B & S_B D_x - C_B D_z + D_z \\ 0 & 0 & & 1 \end{bmatrix} \begin{bmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{bmatrix} \qquad (38) $$

Las ecuaciones que se necesitan para la "transformación inversa" de la cinemática se pueden escribir así:

$$ \begin{aligned} P_x &= C_c C_B Q_x - S_C C_B Q_y + S_B Q_z - C_B D_x - S_B D_z + D_x \\ P_y &= S_C Q_x + C_C Q_y \\ P_z &= -C_C S_B Q_x + S_C S_B Q_y + C_B Q_z + S_B D_x - C_B D_z + D_z \end{aligned} \qquad (39) $$

17.6. Ejemplos de mesa giratoria/inclinable

LinuxCNC incluye módulos cinemáticos para las topologías *xyzac-trt* y *xyzbc-trt* descrito en las matemáticas detalladas arriba. Para usuarios interesados, el código fuente está disponible en el árbol git en el directorio *src/emc/kinematics/*.

Los ejemplos de configuraciones de simulación de xyzac-trt y xyzbc-trt en las estan en Configuraciones de Ejemplo (*configs/sim/axis/visn rotary-tilting/*).

Las configuraciones de ejemplo incluyen los archivos ini requeridos y un subdirectorio de ejemplos con archivos g-code (.ngc). Estas configuraciones sim invocan un modelo tridimensional realista usando la utilidad vismach de LinuxCNC.

17.6.1. Modelos de simulación de Vismach

Vismach es una biblioteca de rutinas Python para mostrar una simulación dinámica de una máquina CNC en la pantalla de la PC. La secuencia de comandos python para una máquina en particular se carga en HAL y los datos se pasan a los pines HAL. El modelo de vismach de espacio de usuario se carga con un comando hal como:

```
loadusr -W xyzac-trt-gui
```

y las conexiones se realizan usando comandos HAL como:

```
net  :table-x   joint.0.pos-fb xyzac-trt-gui.table-x
net  :saddle-y  joint.1.pos-fb xyzac-trt-gui.saddle-y
...
```

Consulte los archivos ini de simulación para obtener detalles sobre las conexiones HAL utilizadas para el modelo de vismach.

17.6.2. Offsets de longitud de la herramienta

Para utilizar las herramientas de una tabla de forma secuencial con offset de la longitud de la herramienta aplicada automáticamente, se requiere un offset Z adicional. Para una herramienta que es más larga que la herramienta "maestra", que normalmente tiene una longitud de herramienta cero, LinuxCNC tiene una variable llamada "motion.tooloffset.z". Si esta variable se transfiere al componente cinemático (y a la secuencia de comandos vismach python), se puede tener en cuenta el offset Z adicional necesario para una nueva herramienta agregando la instrucción del componente, por ejemplo:

$$D_z = D_z + \text{tool-offset}$$

La conexión HAL requerida (para xyzac-trt) es:

```
net :tool-offset motion.tooloffset.z xyzac-trt-kins.tool-offset
```

dónde:

```
:tool-offset ---------------- nombre de la señal
motion.tooloffset.z --------- pin de salida HAL del módulo motion de LinuxCNC
xyzac-trt-kins.tool-offset -- pin HAL de entrada a xyzac-trt-kins
```

17.7. Componentes cinemáticos personalizados

LinuxCNC implementa la cinemática utilizando un componente HAL que está cargado al inicio de LinuxCNC. El módulo cinemático más común, *trivkins*, implementa la cinemática de identidad (trivial) donde hay correspondencia uno-a-uno entre una letra de coordenadas del eje y una articulación del motor. Están disponibles módulos cinemáticos adicionales para sistemas más complejos (incluido *xyzac-trt* y *xyzbc-trt* descritos anteriormente).

Consulte la página de manual de kins (**\$ man kins**) para obtener descripciones breves de los módulos cinemáticos.

Los módulos cinemáticos proporcionados por LinuxCNC suelen escribirse en lenguaje C. Dado que se utiliza una estructura estándar, la creación de un módulo cinemático personalizado se facilita copiando un archivo fuente existente a un archivo de usuario con un nombre nuevo, modifícandolo y luego instálandolo.

La instalación se realiza utilizando halcompile:

```
sudo halcompile --install kinsname.c
```

donde "kinsname" es el nombre que se le da a su componente. El prefijo sudo es requerido para instalarlo y se le pedirá su contraseña de root. Ver la página del comando halcompile para obtener más información (**\$ man halcompile**)

Una vez compilado e instalado, puede hacer referencia a él en la configuración de su máquina. Esto se hace en el archivo ini de su directorio config. Por ejemplo, la especificación ini común:

```
[KINS]
KINEMATICS = trivkins
```

seria reemplazada por

```
[KINS]
KINEMATICS = kinsname
```

donde "kinsname" es el nombre de su programa. Los pines HAL adicionales pueden ser creados por el módulo para elementos de configuración variable tales como D_x, D_y, D_z, offset de herramienta-offset utilizado en el módulo de cinemática xyzac-trt. Estos pines se pueden conectar a una señal para control dinámico o establecer conexiones HAL como:

```
# establecer los parámetros de offsets
net :tool-offset motion.tooloffset.z xyzac-trt-kins.tool-offset
setp xyzac-trt-kins.y-offset 0
setp xyzac-trt-kins.z-offset 20
```

17.8. Figuras

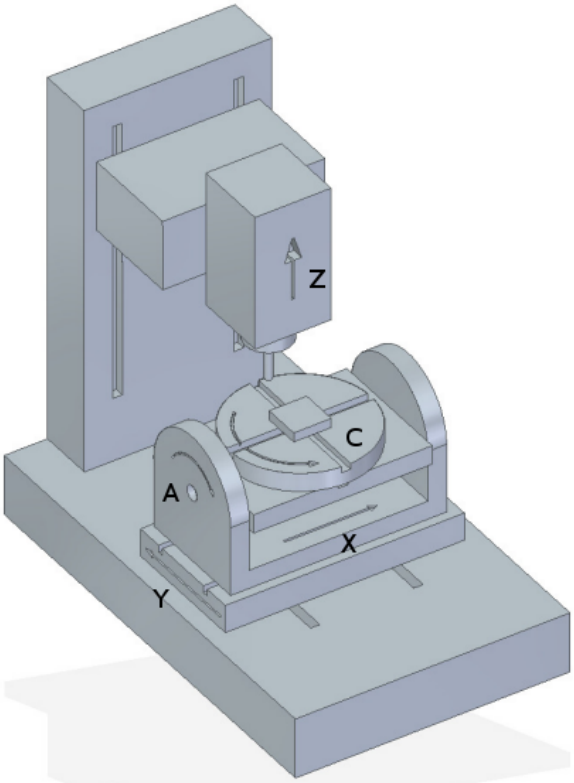


Figura 17.9: Configuración de mesa basculante/giratoria

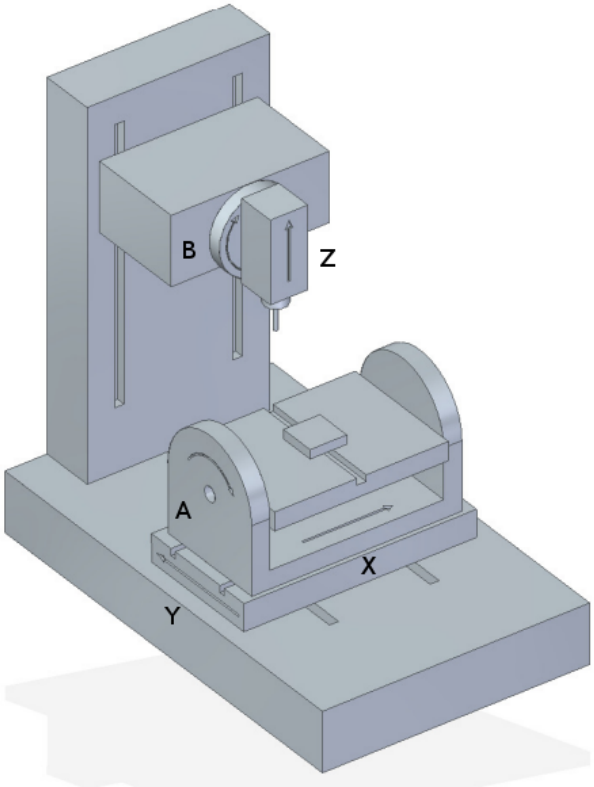


Figura 17.10: Configuración con inclinación de husillo/mesa

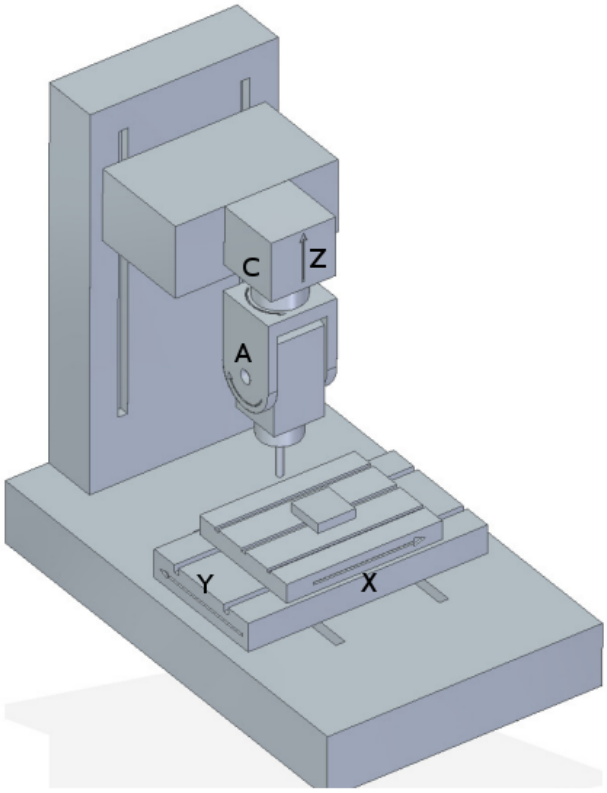


Figura 17.11: Configuracion husillo oscilante / giratoria

// /// == Referencias nope nope nope pdf nope // ///

17.9. REFERENCIAS

1. A Postprocessor Based on the Kinematics Model for General Five-Axis machine Tools: C-H She, R-S Lee, J Manufacturing Processes, V2 N2, 2000.
2. NC Post-processor for 5-axis milling of table-rotating/tilting type: YH Jung, DW Lee, JS Kim, HS Mok, J Materials Processing Technology,130-131 (2002) 641-646.
3. 3D 6-DOF Serial Arm Robot Kinematics, RJ du Preez, SA-CNC-CLUB, Dec. 5, 2013.
4. Design of a generic five-axis postprocessor based on generalized kinematics model of machine tool: C-H She, C-C Chang, Int. J Machine Tools & Manufacture, 47 (2007) 537-545.

Capítulo 18

Ajuste PID

18.1. Controlador PID

Un controlador PID (Controlador Proporcional-Integral-Derivativo) es un mecanismo de control por realimentación ampliamente usado en sistemas de control industrial. [1]

Este controlador calcula la desviación o error entre un valor medido de un proceso (típicamente un proceso industrial) y un valor deseado. La diferencia (o señal de *error*) se usa para calcular un nuevo valor de la entrada al proceso que lleva al valor medido del proceso a su punto de ajuste deseado.

A diferencia de algoritmos de control más simples, el controlador PID puede ajustar las salidas basandose en el historial y tasa de cambio de la señal de error, lo que proporciona un control más preciso y estable. (Se puede demostrar matemáticamente que un bucle PID producirá un control preciso y estable en casos donde un control proporcional simple tendría un error de estado estable o haría que el proceso oscilara).

18.1.1. Conceptos básicos de lazo de control

Intuitivamente, el bucle PID intenta automatizar lo que haria un operario inteligente con un dial medidor y un boton de control. El operador lee el medidor que muestra la medición de la salida de un proceso, y usa el mando para ajustar la entrada del proceso (la *acción*) hasta que la medición de salida del proceso se estabiliza al valor deseado.

En la literatura de control más antigua, este proceso de ajuste se denomina acción *reset*. La posición de la aguja en el medidor es una *medida*, *valor de proceso* o *variable de proceso*. El valor deseado en el medidor se llama *punto de ajuste* (también llamado *valor de ajuste*). La diferencia entre la aguja del indicador y el punto de ajuste es el "error".

Un lazo de control consta de tres partes:

1. Medición por un sensor conectado al proceso (por ejemplo, un codificador),
2. Decisión en un elemento controlador,
3. Acción a través de un dispositivo de salida, como un motor.

A medida que el controlador lee el sensor, resta esta medida del *punto de ajuste* para determinar el *error*. Luego usa el error para calcular una corrección de la variable de entrada del proceso (la *acción*) para que esta corrección elimine el error de la salida del proceso.

En un bucle PID, la corrección se calcula a partir del error de tres maneras: cancelar el error actual directamente (Proporcional), cantidad de tiempo que el error ha persistido sin corregir (Integral), y anticipar el error futuro a partir de la tasa de cambio del error en el tiempo (Derivado).

[1] Esta subsección está tomada de un articulo mucho más extenso encontrado en http://en.wikipedia.org/wiki/PID_controller

Un controlador PID se puede usar para controlar cualquier variable medible que pueda verse afectada por la manipulación de alguna otra variable de proceso. Por ejemplo, se puede usar para controlar la temperatura, la presión, la velocidad de flujo, composición química, velocidad u otras variables. El control de la velocidad de crucero de un automóvil es un ejemplo de un proceso fuera de la industria que utiliza control PID.

Algunos sistemas de control organizan los controladores PID en cascadas o redes. Es decir, un control *maestro* produce señales utilizadas por controladores *esclavos*. Una situación común es el control del motor; a menudo se quiere que el motor tenga una velocidad controlada, con el controlador *esclavo* (a menudo integrado en un driver de frecuencia variable) que gestiona directamente la velocidad basada en una entrada proporcional.Esta entrada *esclava* es alimentada por la salida del controlador *maestro*, que está controlando basándose en una variable relacionada.

18.1.2. Teoría

PID lleva el nombre de sus tres cálculos de corrección; todos añaden o ajustan la cantidad controlada. Estas adiciones son en realidad *restas* de error, porque las proporciones son generalmente negativas:

Proporcional Para manejar el estado presente, el error se multiplica por una P constante (de *proporcional*, negativa) y se añade (restando el error) a la cantidad controlada. P solo es válido en la banda sobre la que la salida del controlador es proporcional al error del sistema. Note que cuando el error es cero, la salida de un controlador proporcional es cero.

Integral Para aprender del pasado, el error está integrado (acumulado) sobre un período de tiempo, y luego multiplicado por una constante I (negativa, haciendo un promedio), y añadido a (restando el error de) la cantidad controlada. Se promedia el error medido para encontrar el error promedio del proceso desde el punto de ajuste. Un sistema proporcional simple oscila, avanzando y retrocediendo alrededor del punto establecido porque no hay nada para eliminar el error cuando se sobrepasa, u oscila y/o se estabiliza a un valor demasiado bajo o demasiado alto. Al agregar una proporción negativa (es decir, restar parte de) del error promedio de la entrada de proceso, la diferencia promedio entre la salida del proceso y el punto de ajuste siempre se está reduciendo. Por lo tanto, eventualmente, una salida del proceso del bucle PID bien ajustado se establecerá en el punto de ajuste.

Derivativo Para manejar el futuro, se calcula la primera derivada (la pendiente del error) con el tiempo y se multiplica por otra constante (negativa) D, y también se añade a (restando el error de) la cantidad controlada. El término derivativo controla la respuesta a un cambio en el sistema. Cuanto mayor es el término derivado, más rápidamente responde el controlador a los cambios en el resultado del proceso.

Más técnicamente, un bucle PID se puede caracterizar como un filtro aplicado a un complejo sistema de dominio de frecuencia. Esto es útil para calcular si realmente alcanzará un valor estable. Si los valores se eligen incorrectamente, la entrada del proceso controlado puede oscilar, y la salida del proceso nunca permanecerá en el punto de ajuste.

18.1.3. Afinacion del lazo

Afinar un lazo de control es ajustar sus parámetros de control (ganancia/banda proporcional, ganancia integral/reset , ganancia derivativa/tasa) a los valores óptimos para la respuesta deseada de control. El comportamiento óptimo en un cambio de proceso o cambio de punto de ajuste varía según la aplicación. Algunos procesos no deben permitir un sobre-exceso de la variable de proceso sobre el punto de ajuste. Otros procesos deben minimizar la energía gastada para alcanzar un nuevo punto de referencia. Generalmente se requiere estabilidad de respuesta y el proceso no debe oscilar para ninguna combinación de condiciones de proceso y puntos de ajuste.

El ajuste de los bucles se hace más complicado por el tiempo de respuesta del proceso; puede tomar minutos u horas que un cambio de punto de ajuste produzca un efecto estable. Algunos procesos tienen un grado de no linealidad y, por lo tanto, parámetros que funcionan bien en condiciones de carga completa no funcionan cuando el el proceso se inicia sin carga. Esta sección describe algunos métodos manuales tradicionales para sintonización de bucle.

Hay varios métodos para ajustar un ciclo PID. La elección del método dependerá en gran medida de si el ciclo se puede sintonizar *fuera de línea* y de la velocidad de respuesta del sistema. Si el sistema puede ser puesto fuera de línea, el mejor método de ajuste a menudo implica someter al sistema a un cambio de un paso la entrada, midiendo la salida en función del tiempo y usando esta respuesta para determinar los parámetros de control.

Método simple Si el sistema debe permanecer en línea, se debe configurar primero un ajuste a cero de los valores I y D. Aumentar P hasta que la salida del lazo oscile. Luego aumentar I hasta que la oscilación se detenga. Por último, aumentar D hasta que el ciclo sea aceptablemente rápido en alcanzar su referencia. Un PID rápido generalmente sobrepasa ligeramente el punto de ajuste establecido para alcanzarlo con más rapidez. Sin embargo, algunos sistemas no pueden aceptar estos rebasamientos.

| ========== | Parámetro | Tiempo de subida | Sobredisparo | Tiempo de estabilizacion | Error de estado estable | P | Disminuir | Aumentar | Pequeño cambio | Disminuir | I | Disminuir | Aumentar | Aumentar | Eliminar | D | Pequeño cambio | Disminuir | Disminuir | Pequeño cambio | ==========

Efectos del aumento de parámetros

Método Ziegler-Nichols Otro método de ajuste se conoce formalmente como el *método Ziegler-Nichols*, presentado por John G. Ziegler y Nathaniel B. Nichols. Comienza de la misma manera que el método descrito anteriormente: primero, se configura I y D a cero y luego se aumenta la ganancia P y se expone el ciclo a una interferencia externa, por ejemplo golpeando el eje del motor para sacarlo del equilibrio, con el fin de determinar la ganancia crítica y el período de oscilación hasta que la salida del bucle comience a oscilar. Anote la ganancia crítica (K_c) y el periodo de la oscilación de la salida (P_c). Luego ajuste los controles P, I y D como muestra la tabla:

| ========== | Tipo de control | P | I | D | P | $.5K_c$ | | | PI | $.45K_c$ | $P_c/1.2$ | | PID | $.6K_c$ | $P_c/2$ | $P_c/8$ | ==========

Pasos Finales Después de ajustar el eje, verifique el error de seguimiento con Halscope para asegurarse de que está dentro de los requisitos de su máquina. Hay más información sobre Halscope en el manual de usuario de HAL.

Capítulo 19

Offsets Externos de Ejes

Los offsets externos de ejes estan soportados durante los movimientos teleop (universal) y coordinado (gcode). Los offsets externos de ejes estan habilitados para cada eje en la configuración del archivo ini y controlados dinámicamente por pines Hal de entrada. La interfaz Hal es similar a la que se utiliza para un volante jog. Este tipo de interfaz está implementado normalmente con un generador de pulsos manual (mpg), conectado a un componente codificador hal que cuenta los pulsos.

19.1. Configuración del archivo ini

Para cada letra de eje (**L** será uno de x,y,z,a,b,c,u,v,w):

```
[AXIS_L]OFFSET_AV_RATIO = valor (controla acel/vel para offsets externos)
```

1. Valores permitidos: 0 <= valor <= 0.9
2. Los valores no permitidos se reemplazan con 0.1, con un mensaje a la salida estándar
3. Valor por defecto: 0 (desactiva el offset externo). Consecuencia: omitiendo [AXIS_L]OFFSET_AV_RATIO se desactiva el offset externo para el eje L.
4. Si no es cero, OFFSET_AV_RATIO (**r**), ajusta la velocidad y aceleración máxima convencional (planificada) para preservar las restricciones en [AXIS_L]:

   ```
   velocidad máxima planificada   = (1-r) * MAX_VELOCITY
   velocidad de offset externo    = (  r) * MAX_VELOCITY

   aceleración máxima planificada = (1-r) * MAX_ACCELERATIOIN
   aceleración de offset externo  = (  r) * MAX_ACCELERATION
   ```

19.2. Pines Hal

19.2.1. Pines Hal de movimiento por eje

Para cada letra de eje (**L** será uno de x,y,z,a,b,c,u,v,w)

1. **axis.L.eoffset-enable** Entrada (bit): habilitacion.
2. **axis.L.eoffset-scale** Entrada (float): factor de escala.

3. **axis.L.eoffset-count** Entrada (s32): entrada al registro de conteo.
4. **axis.L.eoffset-clear** Entrada (bit): borrar el offset solicitado.
5. **axis.L.eoffset** Salida (float): offset externo actual.
6. **axis.L.eoffset-request** Salida (float): offset externo solicitado.

19.2.2. Otros Pines Hal Motion

1. **motion.eoffset-active** Salida (bit): se aplican offsets externos distintos de cero.
2. **motion.eoffset-limited** Salida (bit): movimiento inhibido debido a límite soft.

19.3. Uso

Los pines hal de entrada del eje (enable,scale,counts) son similares a los pines utilizados para volante jog.

19.3.1. Cálculo de offset

En cada período servo, el pin *axis.L.eoffset-count* se compara con el valor en el periodo anterior. El aumento o disminución (delta positivo o negativo) del pin *axis.L.eoffset-count* se multiplica por el valor actual del pin *axis.L.eoffset-scale*. Este producto es acumulado en un registro interno y exportado al pin Hal *axis.L.eoffset-request*. El registro de acumulación se reinicia a cero en cada encendido de la máquina.

El valor de offset solicitado se utiliza para planificar el movimiento para el offset que se aplica a la coordenada *L* y se representa por el pin hal *axis.L.eoffset*. El movimiento previsto respeta las restricciones asignadas de velocidad y aceleración, que pueden ser limitadas si el movimiento neto (offset más jogging teleop o movimiento coordinado) alcanza un límite soft de la coordenada *L*.

Para muchas aplicaciones, el pin *axis.L.eoffset-scale* es constante y la respuesta neta *axis.L.eoffset-request* a *axis.L.eoffset-count* es equivalente al producto del valor acumulado de *axis.L.eoffset-count* y el valor (constante) del pin *axis.L.eoffset-scale*.

19.3.2. Máquina apagada/encendida

Cuando la máquina está apagada, la **posición actual con los offsets se mantiene**, por lo que no hay movimiento inesperado al apagar o encender.

En cada arranque (máquina-on), los registros internos de conteos para cada uno de los pines hal *axis.L.eoffset-count* se ponen a cero y el pin correspondiente de la salida hal *axis.L.eoffset* se restablece a cero.

En otras palabras, los offsets externos **se definen como CERO en cada inicio** (machine-on) independientemente del valor de los pines *axis.L.eoffset-count*. Para evitar confusiones, se recomienda que todos los pines *axis.L.eoffset-count* estén configurados a cero cuando la máquina se apaga.

19.3.3. Límites soft

Los movimientos por offsets externos de eje se planifican independientemente con ajustes de velocidad y aceleración especificados por *[AXIS_L]OFFSET_AV_RATIO*. El movimiento de offset no está coordinado con jogs teleop ni con movimientos coordinados (gcode). Durante el movimiento teleop y el movimiento coordinado (gcode), los límites soft de eje (*[AXIS_L]MIN_LIMIT, MAX_LIMIT*) restringen el movimiento del eje.

Cuando se aplican offsets externos y el movimiento alcanza un límite soft (por aumentos de offset externo o jogging teleop o movimiento coordinado), se activa el pin hal *motion.eoffset-limited* y el valor del eje se mantiene nominalmente al del límite soft. Este pin hal puede ser utilizado por la lógica hal asociada para truncar cuentas de eoffsets adicionales o para detener la

máquina (conectando a *halui.machine.off* por ejemplo). Si el eje se mueve dentro de los límites soft, el pin *motion.eoffset-limited* se reinicia.

Cuando se opera en un límite soft durante el movimiento coordinado que continúa cambiando el valor planificado del eje, el pin de salida *axis.L.eoffset* indicará el offset actual - la distancia necesaria para alcanzar el límite en lugar de la solicitud de offset calculada. Este valor indicado cambiará a medida que cambie el valor planificado del eje.

El pin hal *axis.L.eoffset-request* indica el offset actual solicitado, que es el producto del registro de conteos interno y eoffset-scale. En general, el valor del pin *axis.L.eoffset* retrasa el valor de *axis.L.eoffset-request* puesto que el offset externo está sujeto a un límite de aceleración. Cuando se opera en un límite soft, las actualizaciones adicionales de *axis.L.eoffset-count* continuará afectando al offset externo solicitado como se refleja en el pin Hal *axis.L.eoffset-request*.

Cuando se hace jogging teleop con offsets externos habilitados **y** con valores distintos de cero, el movimiento se detendrá al encontrar un límite soft en el eje infractor **sin un intervalo de desaceleración**. Del mismo modo, durante el movimiento coordinado con offsets externos habilitados, alcanzar un límite soft detendrá el movimiento sin fase de desaceleración. Para este caso, no importa si los offsets son cero.

Cuando el movimiento se detiene sin fase de desaceleración, **los límites de aceleración del sistema pueden violarse** y dar lugar a: 1) a un error de seguimiento (y/o un golpe brusco) en un sistema servo, 2) una pérdida de pasos para un sistema de motor paso a paso. En general, se recomienda que los offsets externos se apliquen de forma que se evite la cercania a los límites softs.

19.3.4. Notas

Los offsets externos se aplican a las letras de coordenadas de eje (xyzabcuvw). Todas las articulaciones deben tener home antes de que los offsets externos del eje sean aplicados.

Si un pin hal *axis.L.eoffset-enable* se restablece cuando su offset no es cero, el offset se mantiene. El offset puede ser borrado por:

1. una conmutacion *Máquina apagada/Máquina encendida*
2. reactivación del pin de habilitación e incremento/decremento del pin Hal *axis.L.eoffset-count* para devolver el offset a cero.
3. pulso en el pin hal *axis.L.eoffset-clear*

Los offsets externos están diseñados para usarse con offsets *pequeños* que se aplican dentro de los límites soft.

Cuando se aplican offsets externos, se respetan los límites soft tanto para jogging teleop como para movimiento coordinado. Sin embargo, cuando se alcanza un límite soft durante el movimiento coordinado, la reduccion del offset externo culpable **no produce alejamiento** del límite soft **si el movimiento planificado continúa en la misma dirección**. Esta circunstancia puede ocurrir ya que la tasa de eliminación de corrección de offset (según lo establecido en *[AXIS_L]OFFSET_AV_RATIO*) puede ser menor que la velocidad de movimiento planificada opuesta. En tales casos, **pausando** (o parando) lo planificado, el movimiento coordinado permitirá el alejamiento del límite soft cuando se hacen cambios correctivos en el offset externo ofensivo.

19.3.5. Advertencia

El uso de offsets externos puede alterar el movimiento de la máquina de una manera significativa. El control de los offsets externos con componentes y conexiones hal y cualquier interfaz de usuario asociada, debe ser cuidadosamente diseñado y probado antes de la implementación.

19.4. Componentes de Hal relacionados

19.4.1. eoffset_per_angle.comp

Componente para calcular un offset externo desde una función basada en un ángulo medido (coordenada rotativa o husillo). Ver la página man para detalles (**$ man eoffset_per_angle**).

19.5. Pruebas

La capacidad de offset externo de un eje se habilita agregando una configuración [AXIS_L] para cada eje candidato. Por ejemplo:

```
[AXIS_Z]
OFFSET_AV_RATIO = 0.2
```

Para las pruebas, es conveniente simular una interfaz de jog de volante utilizando la gui **sim_pin**. Por ejemplo, en una terminal:

```
$ sim_pin axis.z.eoffset-enable axis.z.eoffset-scale axis.z.eoffset-counts
```

El uso de offsets externos se facilita mostrando información relacionada con los offsets actuales: el valor actual de eoffset y el valor de eoffset solicitado, pos-cmd del eje, y (para una máquina con cinemática de identidad) pos-cmd y offset del motor de la articulación correspondiente. La configuración sim proporcionada (ver más abajo) muestra un ejemplo de panel pyvcp para la gui Axis.

En ausencia de una pantalla personalizada, se puede iniciar **halshow** como una aplicación auxiliar con una lista de observación personalizada.

Ejemplo de configuración de archivos ini para simular conexiones del pin hal eoffset y mostrar información de eoffset para el eje z (para cinemática de identidad con z==joint2):

```
[APPLICATIONS]
APP = sim_pin \
      axis.z.eoffset-enable \
      axis.z.eoffset-scale \
      axis.z.eoffset-counts \
      axis.z.eoffset-clear

APP = halshow --fformat "%0.5f" ./z.halshow
```

Donde el archivo z.halshow (en el directorio de configuración) es:

```
pin+joint.2.motor-pos-cmd
pin+joint.2.motor-offset
pin+axis.z.pos-cmd
pin+axis.z.eoffset
pin+axis.z.eoffset-request
pin+motion.eoffset-limited
```

19.6. Ejemplos

Las configuraciones de simulación proporcionadas demuestran el uso de offsets externos como punto de partida para personalización de hardware real del usuario.

Las configuraciones sim utilizan la configuración ini *[HAL]HALFILE = LIB:basic_sim.tcl* para configurar todas las rutina de conexiones hal para los ejes especificados en el archivo ini *[TRAJ]COORDINATES =*. La lógica hal necesaria para demostrar la funcionalidad de offset externo y las conexiones de pines de gui hal para un panel pyvcp se realizan en archivos hal separados. Una configuración no simulada debería reemplazar el elemento *LIB:basic_sim.tcl* con HALFILEs apropiados para la máquina. Los archivos pyvcp proporcionados (.hal y .xml) podrían ser un punto de partida para interfaces gui específicas de la aplicación.

19.6.1. eoffsets_demo.ini

La configuración sim *sim/configs/axis/external_offsets/eoffsets_demo.ini* demuestra una máquina cartesiana XYZ con controles para habilitar offsets externos en cualquier eje.

Se proporcionan pantallas para mostrar todas las posiciones importantes y valores de offset.

Una gui sim_pin proporciona controles para los pines de offset del eje: eoffset-scale, eoffset-count (a través de la señal e:<L>counts) y eoffset-clear (a través de la señal e:clearall)

Se usa un script (eoffsets_monitor.tcl) para establecer los pines *axis.L.counts* en cero en el apagado de la máquina.

19.6.2. jwp_z.ini

La configuración sim *sim/configs/axis/external_offsets/jwp_z.ini* demuestra la implementacion de una capacidad *jog-while-pause* en una sola coordenada (Z):

Los LED del panel se proporcionan para mostrar los estados de elementos importantes.

Se proporcionan controles para establecer el factor de escala eoffset y para incrementar/decrementar/borrar las cuentas eoffset.

19.6.3. dynamic_offsets.ini

Esta configuración sim *sim/configs/axis/external_offsets/dynamic_offsets.ini* demuestra los offsets aplicados dinámicamente mediante la conexión de una forma de onda sinusoidal a las entradas externas de offset de la coordenada z.

Los LED del panel se proporcionan para mostrar los estados de elementos importantes.

Se proporcionan controles para modificar la configuración del archivo ini para velocidad máxima y aceleración máxima del eje Z.

Se proporcionan controles para configurar los parámetros del generador de forma de onda.

Se inicia una aplicación de halscope para mostrar la forma de onda aplicada, la respuesta de offset, y la respuesta del comando motor.

Nota: no se reconocen cambios en la coordenada z, max-acceleration y max-speed mientras se está ejecutando un programa.

19.6.4. opa.ini (eoffset_per_angle)

La configuración opa.ini utiliza el componente hal eoffset_per_angle (**$ man eoffset_per_angle**) para demostrar una máquina XZC con offsets funcionales calculados a partir de la coordenada C (ángulo) y aplicados a la coordenada transversal (X). Los cálculos de offset se basan en un radio de referencia especificado normalmente establecido por un M68 programado (o comando MDI) para controlar un pin **motion.analog-out-NN**.

Los LED del panel se proporcionan para mostrar los estados de elementos importantes.

Se proporcionan funciones para los polígonos internos y externos (nsides> = 3), ondas sinusoidales y ondas cuadradas. Las funciones se pueden multiplicar en frecuencia usando el pin fmul y modificar la amplitud usando el pin rfrac (fracción del radio de referencia).

Se proporcionan controles para iniciar/detener las formas de onda de offset y para establecer el tipo de función y sus parámetros.

Capítulo 20

Intérprete independiente

El intérprete autónomo rs274 está disponible para su uso a través de la línea de comandos.

20.1. Uso

```
Uso: rs274 [-p interp.so] [-t tool.tbl] [-v var-file.var] [-n 0 | 1 | 2]
~~~~~~~~~~[-b] [-s] [-g] [archivo de entrada [archivo de salida]]

~~~~-p: especifique el intérprete conectable a utilizar
~~~~-t: especifique el archivo .tbl (tabla de herramientas) a usar
~~~~-v: especifique el archivo .var (parámetros) a usar
~~~~-n: especifique el modo de continuación:
~~~~~~~~~~~0: continuar
~~~~~~~~~~~1: ingresar al modo MDI
~~~~~~~~~~~2: detener (predeterminado)
~~~~-b: Activa o desactiva el indicador de "eliminación de bloque" (predeterminado:  ↩
    DESACTIVADO)
~~~~-s: Activa o desactiva el indicador 'pila de impresión' (predeterminado: DESACTIVADO)
~~~~-g: Activa o desactiva el indicador 'ir (modo por lotes)' (predeterminado: DESACTIVADO)
~~~~-i: especifique el archivo .ini (predeterminado: sin archivo ini)
~~~~-T: llamar a task_init()
~~~~-l: especifique el log_level (predeterminado: -1)
```

20.2. Ejemplo

Para ver la salida de un bucle, por ejemplo, podemos ejecutar rs274 en el siguiente archivo y ver que el ciclo nunca termina. Para salir del bucle, use Ctrl+Z. Se necesitan los siguientes archivos para ejecutar el ejemplo.

test.ngc

```
#<test> = 123.352

o101 while [[#<test> MOD 60 ] NE 0]
(debug,#<test>)
    #<test> = [#<test> + 1]
o101 endwhile

M2
```

test.tbl

```
T1 P1 Z0.511 D0.125; fresa 1/8
T2 P2 Z0.1 D0.0625; fresa de extremo 1/16
T3 P3 Z1.273 D0.201; taladro de rosca n\textdegree{}7
```

comando

```
rs274 -g test.ngc -t test.tbl
```

Parte VI

Glosario

Lista de términos y su significado. Algunos términos tienen un significado general y varios significados adicionales para usuarios, instaladores y desarrolladores.

Alimentación (feed)
Movimiento relativamente lento y controlado de la herramienta, utilizado al hacer un corte. También se conoce como avance.

Archivo ini
Un archivo de texto que contiene la mayor parte de la información que configura LinuxCNC para una máquina en particular.

Articulacion
Punto o puntos de un par cinemático donde quedan restringidos algunos grados de libertad de uno de los eslabones del par sobre el otro eslabón. Las articulaciones mas típicas son la cilíndrica, la rotula y la guía corredera.

AXIS (GUI)
Una de las interfaces gráficas de usuario disponibles para los usuarios de LinuxCNC. Presenta un uso moderno de menús y botones de mouse mientras automatiza y oculta algunos de los controles LinuxCNC más tradicionales. Es la única interfaz de código abierto que muestra toda la ruta de la herramienta tan pronto como se abre un archivo de código numérico.

Backlash
La cantidad de "juego" o movimiento perdido que ocurre cuando se invierte la dirección en un tornillo de avance u otro sistema mecánico de transmisión de movimiento. Puede ser el resultado de tuercas demasiado sueltas en sus tornillos de avance, deslizamiento en las correas, holgura de cables, "wind-up" en acoplamientos, y otros lugares donde el sistema mecánico no esta "ajustado". La holgura dará lugar a un movimiento impreciso o, en el caso de movimiento causado por fuerzas externas (piense que la herramienta de corte tira de la pieza de trabajo), el resultado puede ser herramientas de corte rotas. Esto puede suceder por el aumento repentino en la carga del cortador cuando la pieza de trabajo es arrastrada a través de la distancia de backlash por la herramienta de corte.

Cinemática
La relación de posición entre las coordenadas universales y las coordenadas de articulación de una máquina. Hay dos tipos de cinemática. La cinemática directa se usa para calcular coordenadas universales desde coordenadas de articulaciones. La cinemática inversa se usa para exactamente lo opuesto. Tenga en cuenta que la cinemática no tiene en cuenta las fuerzas, momentos etc. en la máquina. Es solo para posicionamiento.

Coordenadas de articulación
Especifican los ángulos entre las articulaciones individuales de la máquina. Ver también Cinemática

Coordenadas universales
Este es el marco de referencia absoluto Da coordenadas en términos de un marco de referencia fijo que se asigna a algún punto (generalmente la base) de la máquina.

Comp
Una herramienta software utilizada para construir, compilar e instalar componentes HAL en LinuxCNC.

Compensaciones (offsets)
Una cantidad arbitraria, agregada al valor de algo para hacerlo igual a algún otro valor deseado. Por ejemplo, los programas gcode se escriben a menudo alrededor de algún punto conveniente, como X0, Y0. Las compensaciones del montaje se pueden usar para cambiar el punto de ejecución real de ese programa gcode para adaptarse adecuadamente a la verdadera ubicación de las mordazas o fijaciones de otro tipo. Los desplazamientos de herramienta se pueden usar para cambiar la longitud "no corregida" de una herramienta para obtener la longitud real de esa herramienta.

Compensación del Backlash
Cualquier técnica que intente reducir el efecto del backlash sin realmente eliminarlo del sistema mecánico. Esto generalmente se hace en el software del controlador. Esto puede corregir el lugar final de la pieza en movimiento pero falla para resolver problemas relacionados con los cambios de dirección mientras hay movimiento (piense en la interpolación circular) y el movimiento que se produce cuando las fuerzas externas (piense que la herramienta de corte tira de la pieza de trabajo) es la fuente del movimiento.

CNC

Computer Numerical Control. El término general utilizado para referirse al control por computadora de una maquina. En lugar de un operador humano girando volantes para mover una herramienta de corte, CNC usa una computadora y motores para mover la herramienta, en función de un programa de pieza.

Configuración (como nombre)

Un directorio que contiene un conjunto de archivos de configuración. Las configuraciones personalizadas se guardan normalmente en el directorio de usuario home/linuxcnc/configs. Estos archivos incluyen el archivo INI tradicional de LinuxCNC y archivos HAL. Una configuración también puede contener varios archivos generales que describen herramientas, parámetros y conexiones NML.

Configuración (como verbo)

La tarea de configurar LinuxCNC para que coincida con el hardware en una máquina en particular.

DRO

Una lector digital es un sistema de dispositivos de medición de posición montados sobre una máquina herramienta, que están conectados a una pantalla numérica que muestra la posición actual de la herramienta con respecto a alguna posición de referencia. Los DRO son muy populares en las máquinas herramienta manuales porque miden la posición verdadera de la herramienta sin juego, incluso si la máquina tiene tornillos Acme con holgura. Algunos DRO usan codificadores lineales de cuadratura para llevar la información de posición de la máquina, y algunos otras usan métodos similares a resolvers giratorios.

EDM

EDM es un método para eliminar metal en condiciones difíciles, difíciles de máquinar o metales duros, o donde las herramientas rotatorias no podrían producir la forma deseada de una manera rentable. Un excelente ejemplo es la matriz de un punzón rectangular, donde se desean esquinas internas agudas. Las operaciones de fresado no pueden dar ángulos internos agudos con herramientas de diámetro finito. Una máquina EDM *wire* puede hacer esquinas internas con un radio solo un poco más grande que el radio del hilo. Un EDM de electrodo sumergido puede hacer esquinas internas con un radio solo un poco más grande que el radio en la esquina del electrodo.

Eje

Una de las partes movibles de la máquina, controladas por computadora. Para una fresadora vertical típica, la mesa es el eje X, el el carro transversal es el eje Y, y el cabezal es el eje Z. Los ejes angulares, como mesas giratorias, se conocen como A, B y C. Adicionalmente, los ejes lineales relativos a la herramienta se llaman U, V y W respectivamente.

EMC

Controlador de máquina mejorado (Enhanced Machine Controller). Inicialmente un proyecto NIST. Renombrado a LinuxCNC en 2012.

EMCIO

El módulo dentro de LinuxCNC que maneja E/S de propósito general, sin relación con el movimiento real de los ejes.

EMCMOT

El módulo dentro de LinuxCNC que maneja el movimiento real de la herramienta de corte. Se ejecuta como un programa en tiempo real y controla directamente los motores.

Encoder

Un dispositivo para medir la posición. Usualmente un dispositivo mecánico-óptico, que emite una señal de cuadratura. La señal puede ser contada por hardware especial, o directamente por puerto paralelo con LinuxCNC.

Entero con signo

((((Entero con signo))) Un número entero que puede tener un valor positivo o negativo. En HAL se lo conoce como s32. (Un entero de 32 bits con signo tiene un rango utilizable de -2,147,483,647 a +2,147,483,647.)

Entero sin signo

((((Entero sin signo))) Un número entero que no tiene signo. En HAL se lo conoce como u32. (Un entero de 32 bits sin signo tiene un rango útil de cero a 4,294,967,296.)

Error de seguimiento

La diferencia entre valores de la señal de retroalimentación con la señal ordenada para verificar que el motor está siguiendo el comando. Hay un límite ajustable del error permitido durante un movimiento. Cuando se excede el límite, se crea una condición de error que informa un error en este límite.

Ganancia FF

Mientras que las ganancias P-I-D son reactivas (son función del error que ya ha ocurrido) las ganancias FF (o Feed-Forward o en avance) son proactivas. Las ganancias de avance, que incluyen avance de velocidad y avance de aceleración, preceden a los comandos necesarios para lograr un error cero y los inyectan en el bucle de control.

Gmoccapy (GUI)

Interface de usuario gráfica disponible para los usuarios de LinuxCNC. Presenta el uso y la sensación de un control industrial y puede ser utilizado con pantalla táctil, mouse y teclado. Admite pestañas incrustadas y mensajes de usuario manejados por hal. Ofrece muchos recursos para ser controlados con hardware. Gmoccapy es altamente configurable.

G-Code

Término genérico utilizado para referirse a la parte común de la mayoría de lenguajes de programación de piezas. Hay varios dialectos de código G. LinuxCNC usa RS274/NGC.

GUI

Interfaz gráfica de usuario.

General

Un tipo de interfaz que permite las comunicaciones entre una computadora y un ser humano (en la mayoría de los casos) a través de la manipulación de iconos y otros elementos (widgets) en una pantalla de computadora.

LinuxCNC

Una aplicación que presenta una pantalla gráfica al operador de la máquina, que permite su manipulación, y el correspondiente programa de control.

HAL

Capa de abstracción de hardware. Al más alto nivel, es simplemente una manera de disponer de una serie de bloques de construcción que se cargarán e interconectarán para ensamblar un sistema complejo. Muchos de los componentes básicos son controladores para dispositivos de hardware. Sin embargo, HAL puede hacer más que configurar controladores de hardware.

Home

Una ubicación específica en el entorno de trabajo de la máquina que se usa para asegurar que tanto la computadora como la máquina están de acuerdo sobre la posición de la herramienta.

Homing

Acción de definicion de la posición home.

Husillo

La parte de una máquina herramienta que gira para hacer el corte. En una fresadora o taladro, el husillo sostiene la herramienta para cortar. En un torno, el husillo sostiene la pieza de trabajo.

Instancia

Una instancia es un objeto software real creado en tiempo de ejecución. En la jerga de los programadores, el objeto Lassie es una instancia de la clase perro

Jog

Mover manualmente un eje de una máquina. El Jogging mueve los ejes una cantidad fija para cada pulsación de tecla, o mueve los ejes a una velocidad constante siempre que mantenga presionada la tecla. En modo manual, la velocidad de desplazamiento se puede establecer desde la interfaz gráfica.

kernel-space

Ver tiempo real.

Lubricación

Operación de aporte de lubricante a las partes móviles en fricción de una maquina.

Máquina de medición de coordenadas

Una máquina de medición de coordenadas es utilizada para hacer muchas mediciones precisas en piezas. Estas máquinas pueden ser usadas para crear datos CAD para piezas donde no se pueden encontrar los planos, cuando un prototipo hecho a mano debe ser digitalizado para la fabricación de moldes, o para verificar la precisión de las piezas mecanizadas o moldeadas.

MDI
Entrada de datos manual. Este es un modo de operación donde el controlador ejecuta líneas únicas de código G a medida que son introducidas por el operador.

Motor paso a paso
Un tipo de motor que gira pasos fijos. Al contar los pasos, es posible determinar la distancia que el motor ha hecho recorrer. Si la carga excede la capacidad de par del motor, omitirá uno o más pasos, causando errores de posición.

Movimiento transversal (Traverse Move)
Un movimiento en línea recta desde un punto de inicio hasta un punto final.

NIST
Instituto Nacional de Estándares y Tecnología. Una agencia del Departamento de Comercio de Estados Unidos.

NML
Neutral Message Language; proporciona un mecanismo para manejar múltiples tipos de mensajes en el mismo buffer así como simplifica la interfaz para codificar y descodificar almacenamientos intermedios en formato neutral y su mecanismo de configuración.

Número de punto flotante
Un número que tiene un punto decimal. (12.300). En HAL se lo conoce como float.

Offset
Ver Compensaciones

Par cinemático
Conjunto de dos sólidos rígidos, unidos de forma que permite ciertos movimientos relativos y restringe otros.

Pncconf
Asistente de configuración de tarjetas Mesa

Programa de pieza
Una descripción de una pieza, en un lenguaje que el controlador puede entender. Para LinuxCNC, ese lenguaje es RS-274/NGC, comúnmente conocido como código G.

Python
Lenguaje de programación de propósito general y muy alto nivel. Utilizado en LinuxCNC para la GUI AXIS, la herramienta de configuración Stepconf y programación de varios scripts de códigos G.

Rápid
Movimiento rápido, posiblemente menos preciso, de la herramienta. Comúnmente utilizado para moverse entre cortes. Si la herramienta se encuentra con la pieza de trabajo o la fijación de la misma durante un rápid, ¡algo va mal!. El desastre queda asegurado.

Reajuste (override) de avance
Un cambio manual, controlado por el operador, en la velocidad a la que la herramienta se mueve durante el corte. A menudo solía permitir que el operador ajustase a herramientas que están algo embotadas, o cualquier otra cosa que requiera que la velocidad de alimentación sea "retocada".

Reajuste de la velocidad del husillo
Un cambio manual controlado por el operador en la velocidad a la que la herramienta gira durante el corte. A menudo se usa para permitir que el operador ajuste el "chatter" causado por los dientes del cortador. El reajuste de velocidad del husillo asume que el software LinuxCNC ha sido configurado para controlar la velocidad del husillo.

Refrigerante
Fluido destinado a la extracción del calor producido durante el corte

Retroalimentación (feedback)
Un método (por ejemplo, señales de encoder en cuadratura) mediante el cual LinuxCNC recibe información sobre la posición de los motores

RS-274/NGC
El nombre formal del dialecto utilizado por los programas de piezas de LinuxCNC.

RTAI
Interfaz de aplicaciones en tiempo real. Consulte en https://www.rtai.org/, una de las extensiones en tiempo real para Linux que LinuxCNC puede usar para lograr un rendimiento en tiempo real.

RTAPI
Una interfaz portátil para sistemas operativos en tiempo real incluyendo pthreads RTAI y POSIX (como PREEMPT-RT) con extensiones en tiempo real.

RTLINUX
Ver https://en.wikipedia.org/wiki/RTLinux, una antigua extensión en tiempo real para Linux que LinuxCNC solía usar para lograr un rendimiento en tiempo real. Obsoleto; reemplazado por RTAI.

Servo Loop
Un lazo de control utilizado para controlar la posición o velocidad de un motor equipado con un dispositivo de retroalimentación.

Servo motor
Generalmente, cualquier motor que use retroalimentación de detección de errores para corregir la posición de un actuador. Además, un motor que está especialmente diseñado para proporcionar mejores rendimientos en tales aplicaciones.

Stepconf
Un asistente de configuración de LinuxCNC. Puede manejar muchas máquinas basadas en comandos de movimiento de paso y dirección. Escribe una configuración completa después de que el usuario responda algunas preguntas sobre la computadora y la máquina en la que LinuxCNC se ejecutará.

TASK
El módulo dentro de LinuxCNC que coordina la ejecución general e interpreta el programa de pieza.

Tcl/Tk
Un juego de herramientas de widgets gráficos y lenguaje de scripting con el que fueron escritos varias de las GUI de LinuxCNCs y asistentes de selección.

Tiempo real
Software que está destinado a cumplir plazos de tiempo muy estrictos. En Linux, para cumplir estos requisitos es necesario instalar un kernel en tiempo real, como RTAI (o PREEMPT-RT), y construir el software para ejecutarse en el ambiente especial de tiempo real. Por esta razón, el software en tiempo real se ejecuta en espacio kernel.

Tornillo Acme
Un tipo de tornillo de avance que usa la forma de filete Acme. Los filetes Acme tienen una fricción y desgaste algo menores que los filetes triangulares simples, pero los tornillos de bola son aún mejores. La mayoría de las máquinas herramientas manuales usan tornillos acme.

Tornillo de avance
Un tornillo girado por un motor para mover una mesa u otra parte de una máquina. Los tornillos suelen ser tornillos de bola o tornillos acme, aunque se pueden usar roscados triangulares convencionales en tornillos donde la precisión y la larga duración no son tan importantes como el bajo costo.

Tornillo de bolas
Un tipo de tornillo de avance que usa pequeñas bolas de acero endurecido entre la tuerca y el tornillo para reducir la fricción. Los tornillos de bolas tienen muy baja fricción y backlash, pero generalmente son bastante caros.

Tuerca de bolas
Una tuerca especial diseñada para usarse con tornillos de bolas. Contiene un pasaje interno para recircular las bolas de un extremo del tornillo al otro.

Unidades
Consulte "Unidades de máquina", "Unidades de visualización" o "Unidades de programa".

Unidades de máquina
Las unidades lineales y angulares utilizadas para configuración de la máquina. Estas unidades se especifican y usan en el archivo ini. Los pines HAL y los parámetros generalmente también están en unidades de máquina.

Unidades de programa

Las unidades lineales y angulares utilizadas en un programa de pieza. Las unidades lineales de programa no tienen que ser las mismas que las unidades de máquina. Ver G20 y G21 para más información. Las unidades angulares de programa son siempre grados.

Unidades de visualización

Las unidades lineales y angulares utilizadas para mostrar en monitor.

Velocidad de alimentación

La velocidad a la que se produce un movimiento de corte. En el modo automático o mdi, la velocidad de avance se ordena usando una palabra F. F10 significaría diez unidades máquina por minuto.

Parte VII

Seccion Legal

N.T. Estos textos legales solo se muestra en Inglés ya que las traducciones no se reconocen oficialmente.

Capítulo 21

Terminos de Copyright

Copyright (c) 2000-2020 LinuxCNC.org

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Capítulo 22

Licencia GNU de Documentation Libre

GNU Free Documentation License Version 1.1, March 2000

Copyright © 2000 Free Software Foundation, Inc. 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA. Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission. B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five). C. State on the Title page the name of the publisher of the Modified Version, as the publisher. D. Preserve all the copyright notices of the Document. E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices. F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below. G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice. H. Include an unaltered copy of this License. I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating

the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence. J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission. K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein. L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles. M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version. N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Capítulo 23

Índice alfabético

_
ÍNDICE HOME SIN REINICIO DEL CODIFICADOR, 388
0-10v Spindle Speed Example, 465
5-Axis Kinematics, 597
7i65, 541

A
abs, 539
ajustes ini (pines HAL), 520
alimentación, 628
and2, 538
Arc Distance Mode, 313
ARCHIVO DE PARÁMETROS, 369
Archivo INI
    Sección AXIS, 376
    Sección CONJUNTA, 376
    Sección de APLICACIONES, 372
    Sección DISPLAY, 365
    Sección EMC, 365
    Sección EMCIO, 382
    Sección EMCMOT, 371
    Sección FILTER, 368
    Sección HAL, 371
    Sección HALUI, 372
    Sección KINS, 375
    Sección RS274NGC, 369
    Sección TAREA, 371
    Sección TRAJ, 373
Archivos de llamada, 333
at_pid, 542
Automatic Login, 73
Automatic Startup, 74
Axis
    Cambio de herramienta manual, 117
    Tool Touch Off, 107
axis, 538
Axis Configuration, 42

B
Backlash, 377
backlash, 628
biquad, 540
Bit, 483
bldc_hall3, 542
blend, 539
borde, 538
bucles, 331
bucles condicionales, 331

C
CÓDIGO DE INICIO RS274NGC, 370
Cambio de herramienta manual, 117
cd, 75
Changing Directories, 75
charge_pump, 542
Ciclo de acabado del torno G70, 297
Ciclo de desbaste del torno G71 G72, 298
Cinco Fases, 552
cinemática, 590, 628
Cinemática Trivial, 590
clarke2, 542
clarke3, 542
clarkeinv, 542
ClassicLadder, 476
classicladder, 538
CNC, 471, 629
codificador, 629
Comments, 328, 362
comp, 539, 628
compensación del backlash, 628
Compensation, 377
Configuracion de Ejes, 35
constant, 539
control de la altura de la antorcha, 541
conv_bit_s32, 540
conv_bit_u32, 540
conv_float_s32, 540
conv_float_u32, 540
conv_s32_bit, 540
conv_s32_float, 540
conv_s32_u32, 540
conv_u32_bit, 540
conv_u32_float, 540
conv_u32_s32, 540
coordenadas de articulación, 628
Coordenadas de la máquina G53, 295

coordenadas universales, 628
counter, 539
Cuatro Fases, 551

D
ddt, 539
deadzone, 539
debounce, 538, 561
Determinación de la calibración del husillo, 47
DH parameters Examples, 594
Diagrama de bloque del codificador, 556
Diagrama de bloques PID, 558
dir, 75
Distance to accelerate to max speed, 43
Dos y Tres Fases, 550
Driver Microstepping, 43
DRO, 629
Dwell
    Feed Out, 313
dwell, 25

E
Editing a Root File, 74
EDM, 629
eje, 629
EMC, 629
EMCIO, 629
EMCMOT, 629
encoder, 542, 555
encoder_ratio, 542
Encontrar la máxima aceleración, 45
Error de seguimiento, 629
Errores de código O, 334
estop_latch, 542

F
F: Establecer velocidad de alimentación, 334
Feed Out, 312, 313
feedcomp, 542
FERROR, 378
find, 76
Finding a File, 76
flipflop, 538
Float, 483

G
G Code Table, 273
G-Code, 630
G0 Rapid Move, 274
G1 Linear Move, 275
G10 L1 Tool Table, 284
G10 L10 Set Tool Table, 286
G10 L11 Set Tool Table, 286
G10 L2 Coordinate System, 284
G10 L20 Set Coordinate System, 287
G17 - G19.1 Plane Select, 287
G2
    G3 Arc Move, 276
G20 Units, 287
G28 Go/Set Predefined Position, 288
G3 Arc Move, 276
G30 Go/Set Predefined Position, 288
G33 Spindle Synchronized Motion, 289
G33.1 Rigid Tapping, 290
G38.n Probe, 291
G4 Dwell, 280
G40 Cutter Compensation Off, 292
G41 G42 Cutter Compensation, 292
G41.1 G42.1 Dynamic Compensation, 293
G43 Tool Length Offset, 293
G43.1 Dynamic Tool Length Offset, 294
G43.2 Apply additional Tool Length Offset, 294
G49 Cancel Tool Length Offset, 295
G5 Cubic spline, 280
G5.1 Quadratic spline, 281
G5.2 G5.3 NURBS Block, 281
G54-G59.3 Select Coordinate System, 295
G61 G61.1 G64 Path Mode, 296
G64 Path Blending, 296
G7 Lathe Diameter Mode, 283
G73 Drilling Cycle Chip Break, 299
G74 Left-hand Tapping Cycle Dwell, 299
G76 Threading Cycle, 300
G8 Lathe Radius Mode, 283
G80 Cancel Modal Motion), 305
G80-G89 Canned Cycles, 302
G81 Drilling Cycle, 307
G82 Drilling Cycle Dwell, 311
G83 Peck Drilling, 311
G84 Right-hand Tapping Cycle Dwell, 311
G85 Boring
    Feed Out, 312
G86 Boring
    Spindle Stop
        Rapid Move Out, 312
G89 Boring
    Dwell
        Feed Out, 313
G90
    G91 Distance Mode, 313
G91 Distance Mode, 313
G92 Coordinate System Offset, 313
G92.3 Restaurar compensaciones G92, 314
G93
    G94
        G95 Feed Rate Mode, 315
G94
    G95 Feed Rate Mode, 315
G95 Feed Rate Mode, 315
G96
    G97 Spindle Control Mode, 315
G97 Spindle Control Mode, 315
G98
    G99 Canned Cycle Return, 316

G99 Canned Cycle Return, 316
gantrykins, 541
gearchange, 542
genhexkins, 541
genserkins, 541
gksudo, 75
gladevcp, 538
grep, 76
GUI, 628, 630

H
HAL, 471, 630
HAL Component, 474
HAL Physical-Pin, 475
HAL Pin, 475
HAL Signal, 475
hal-ax5214h, 476
hal-gm, 476
hal-m5i20, 476
hal-motenc, 477
hal-parport, 477
hal-ppmc, 477
hal-stg, 477
hal-vti, 477
halcmd, 477
Halmeter, 523
    Halmeter Tutorial, 494
halmeter, 477
Halmeter Tutorial, 494
halscope, 477
halui, 476
hm2_7i43, 541
hm2_pci, 541
HOME, 388
home, 630
HOME COMPARTIDO, 389
HOME IGNORE LIMITS, 388
Home Latch Direction, 43
HOME LATCH VEL, 387
Home Location, 43
HOME OFFSET, 388
HOME POR CODIFICADOR ABSOLUTO, 389
HOME SEARCH VEL, 378, 387
Home Search Velocity, 43
Home Switch Location, 43
HOME USA ÍNDICE, 388
homing, 630
hostmot2, 541
husillo, 23, 630
hypot, 539

I
ilowpass, 542
INDEXADOR CON BLOQUEO, 390
Indirección, 332
Inhibición de referencia, 391
INI, 628
INI File
    Comments, 362
Instance, 630
integ, 539
invert, 539
iocontrol, 476
iocontrol (pins HAL), 519

J
Jitter Maximo del Periodo Base, 36
jog, 630
Joint, 628
joyhandle, 543

K
kins, 541
knob2float, 543

L
LÍMITE MÁXIMO, 376
Latency Test, 36
Leadscrew Pitch, 43
limit1, 540
limit2, 540
limit3, 540
Listing files in a directory, 75
Local Offsets, 295
logic, 538
loop, 632
ls, 75
lut5, 538, 562

M
M0
    M1 Program Pause, 316
M1 Program Pause, 316
M100 - M199 User Defined Commands, 326
M19 Orient Spindle, 319
M2
    M30 Program End, 317
M3
    M4
        M5 Spindle Control, 318
M30 Program End, 317
M4
    M5 Spindle Control, 318
M48
    M49 Speed and Feed Override Control, 320
M49 Speed and Feed Override Control, 320
M5 Spindle Control, 318
M50 Feed Override Control, 320
M51 Spindle Speed Override, 320
M52 Adaptive Feed Control, 320
M53 Feed Stop Control, 321
M6-Tool-Change, 318
M60 Pallet Change Pause, 317
M61 Set Current Tool, 321

M62 - M65 Digital Output Control, 321
M66 Wait on Input, 322
M67 Analog Output
Synchronized, 322
M68 Analog Output, 323
M7
M8
M9 Coolant Control, 319
M70 Save Modal State, 323
M71 Invalidate Stored Modal State, 324
M72 Restore Modal State, 324
M73 Save and Autorestore Modal State, 325
M8
M9 Coolant Control, 319
M9 Coolant Control, 319
M98
M99, 329
M99, 329
máquina de medición de coordenadas, 630
máquinas cartesianas, 590
maj3, 540
Man Pages, 74
match8, 538
MAX ACCELERATION, 375
MAX LIMIT, 378
Max Step Rate, 36
MAX VELOCITY, 375
Maximum Acceleration, 43
Maximum Velocity, 43
maxkins, 541
MDI, 631
Min Base Period, 36
MIN FERROR, 378
MIN LIMIT, 376, 378
minmax, 543
modo de control de ruta, 26
motion, 476, 538
motion(pines HAL), 516
motor paso a paso, 631
Motor Steps Per Revolution, 42
Movimiento transversal, 631
mult2, 539
mux16, 539
mux2, 539
mux4, 539
mux8, 539

N
near, 539
net, 480
NIST, 631
NML, 631
Nombre de la Maquina, 35
not, 538

O
offset, 539
offsets, 628
oneshot, 538
Operando sin Switches Home, 48
or2, 538
ORIENT OFFSET, 370

P
Parámetro HAL, 474
Parallel Port Setup, 38
paso bajo, 540
PID, 558
pluto_servo, 541
pluto_step, 541
Programa de pieza, 631
Programacion del planificador, 17
Pulley Ratio, 43
Pulse rate at max speed, 43
pumakins, 541
pwd, 75
PWM Spindle Speed Example, 465
PWMgen, 554
pwmgen, 542

R
rápido, 631
Rapid Move Out, 312
reajuste de avance, 631
Referencia inmediata, 390
refrigerante, 23, 25
repetir bucle, 332
retroalimentación, 631
rotatekins, 541
RS274NGC, 631
RTAI, 632
RTAPI, 632
RTLINUX, 632
RUTA SUBROUTINA, 370

S
s32, 483
S: Establecer velocidad del husillo, 334
sample_hold, 543
sampler, 543
scale, 540
scarakins, 541
Searching for Text, 76
Sección AXIS, 376
Sección CONJUNTA, 376
Sección de APLICACIONES, 372
Sección DISPLAY, 365
Sección EMC, 365
Sección EMCIO, 382
Sección EMCMOT, 371
Sección FILTER, 368
Sección HAL, 371
Sección HALUI, 372
Sección KINS, 375

Sección RS274NGC, 369
Sección TAREA, 371
Sección TRAJ, 373
SECUENCIA HOME, 389
select8, 538
Selector de configuración, 28
serport, 541
servo motor, 632
Siggen, 561
sim_encoder, 543
Simulated Encoder, 560
sphereprobe, 543
Spindle At Speed Example, 467
Spindle Direction Example, 465
Spindle Enable Example, 465
Spindle Soft Start Example, 465
Spindle Stop
    Rapid Move Out, 312
Spindle Synchronized Motion Example, 466
stepgen, 497, 542, 545
Stepgen Block Diagram, 546, 548
steptest, 543
streamer, 543
Subrutinas, 328
    bucles, 331
    bucles condicionales, 331
    M98
        M99, 329
    repetir bucle, 332
sudo gedit, 75
sum2, 539
supply, 476, 543
Synchronized, 322

T
T: Seleccionar herramienta, 334
Table Travel, 43
TASK, 632
Terminal Commands, 75
Test this axis, 43
threads, 538
threadtest, 543
tiempo, 483
tiempo real, 632
time, 543
Time to accelerate to max speed, 43
timedelay, 543
timedelta, 543
Tipo de Driver, 35
Tipo HAL, 475
Tk, 632
tmax, 483
toggle, 543
toggle2nist, 543
Tool Touch Off, 107
Tool-Table-Format, 339
tornillo acme, 632
tornillo de avance, 632
tornillo de bola, 632
Touch Off, 338
Trajectory Control, 16
tripodkins, 541
tristate_bit, 543
tristate_float, 543
trivkins, 541
tuerca de bolas, 632
Tutorial Halscope, 501

U
u32, 483
unidades, 25, 632
UNIDADES ANGULARES, 375
unidades de máquina, 632
unidades de programa, 633
unidades de visualización, 633
UNIDADES LINEALES, 375
Unidades Maquina, 35
UNITS, 377
updown, 539
USER M PATH, 370

V
Valores de retorno, 333
velocidad de avance, 25, 633
VOLATILE HOME, 390

W
watchdog, 543
wcomp, 539
weighted_sum, 539
Working Directory, 75

X
xor2, 538