Manual del Desarrollador V2.8.4, 2022-09-18

Contents

Chapter 1

Introducción

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.

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.

Chapter 2

Notas sobre el código

2.1 Audiencia

Este documento es una colección de notas sobre los aspectos internos de LinuxCNC. Tiene principalmente interés para los desarrolladores. Sin embargo, gran parte de la información también puede ser de interés para integradores de sistemas u otros que estén simplemente interesados sobre cómo funciona LinuxCNC. Mucha de esta información está ahora desactualizada y nunca ha sido revisada su precisión.

2.2 Organización

Habrá un capítulo para cada uno de los componentes principales de LinuxCNC, así como capítulos que cubren cómo esos componentes trabajan juntos. Este documento es un trabajo en progreso y su diseño puede cambiar en el futuro.

2.3 Términos y definiciones

- *EJE*: un eje es uno de los nueve grados de libertad que define la posición de una herramienta en el espacio cartesiano tridimensional. Los nueve ejes son referidos como X, Y, Z, A, B, C, U, V y W. Las coordenadas lineales ortogonales X, Y y Z determinan dónde está posicionada la punta de la herramienta. Las coordenadas angulares A, B y C determinan la orientación de la herramienta. Un segundo conjunto de coordenadas lineales ortogonales U, V y W permite el movimiento de la herramienta (generalmente para acciones de corte) en relación con los ejes previamente desplazados y rotados. Lamentablemente, "eje" se usa a veces para significar un grado de libertad de la máquina en sí, como los carros longitudinal y transversal o el avance fino del husillo de una fresadora vertical. En estas maquinas, esto no causa confusión ya que, por ejemplo, el movimiento de la mesa corresponde directamente al movimiento a lo largo del eje X. Sin embargo, las articulaciones de hombro y codo de un brazo robótico y los actuadores lineales de un hexápodo no se corresponde al movimiento a lo largo de ningún eje cartesiano y en en general es importante hacer la distinción entre el eje cartesiano y grados de libertad de la máquina. En este documento, esto último se llamarán *articulaciones*, no ejes. (Las GUI y algunas otras partes de el código no siempre sigue esta distinción, pero las partes internas de el controlador de movimiento si lo hacen.)
- *ARTICULACIÓN*: una articulación es cada una de las partes móviles de la máquina. Las articulaciones son distintas de los ejes, aunque los dos términos a veces se usan (incorrectamente) para significa lo mismo. En LinuxCNC, una articulación es un objeto físico que puede ser movido, no una coordenada en el espacio. Por ejemplo, todos los carros, la palanca del husillo o un plato giratorio de una fresadora vertical son articulaciones. El hombro, el codo y la muñeca de un brazo robótico son articulaciones, al igual que los actuadores lineales de un hexápodo. Cada articulación tiene un motor o actuador de algún tipo asociado con ella. Las articulaciones no corresponden necesariamente a los ejes X, Y y Z, aunque para máquinas con cinemática trivial, puede ser el caso. Incluso en esas máquinas, la posición articular y la posición del eje son cosas inherentemente diferentes. En este documento, los términos *articulación* y *eje* se utilizan con cuidado para respetar sus distintos significados. Desafortunadamente, eso no es necesariamente cierto en ningún otro lado. En en particular, las GUI para máquinas con

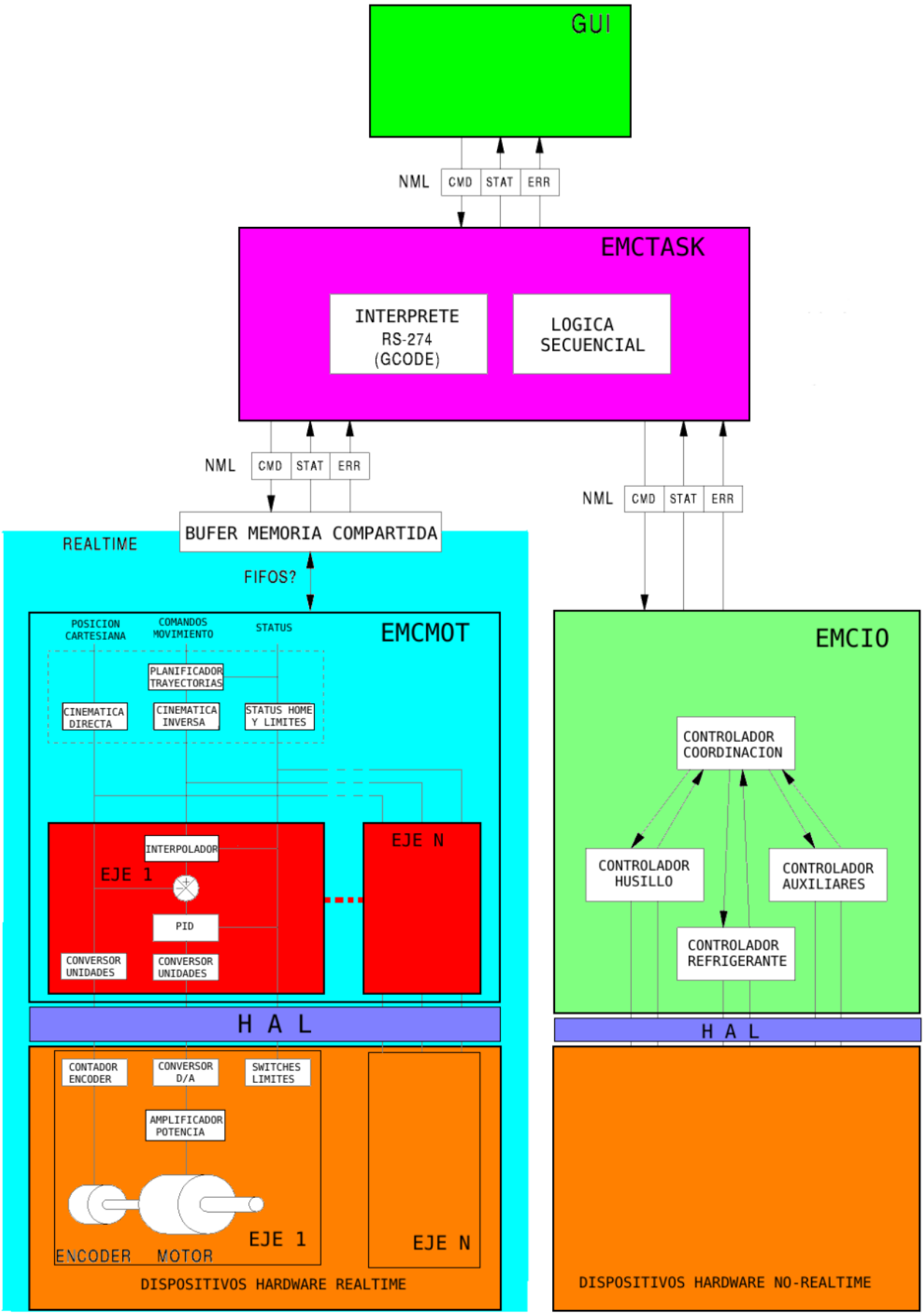
cinemática trivial pueden pasar por alto o oculta completamente la distinción entre articulaciones y ejes. Adicionalmente, el archivo ini usa el término *eje* para datos que serían más precisos describirse como datos de articulaciones, como las escalas de entrada y salida, etc.

```
    N.T. En la version 2.8 de Linuxcnc ya se hace esta distinción.
El archivo .ini cuenta con la nueva sección [JOINT_<num>]. Muchos de los  ↩
   parámetros que antes
    eran propios de la sección [AXIS_<letter>] están ahora en la nueva sección.  ↩
       Otras secciones,
como por ejemplo [KINS], también adquieren nuevos parámetros para ajustarse a  ↩
   esto.
Se ha previsto un mecanismo para transformar archivos .ini antiguos a la nueva  ↩
   configuración
ejes/articulaciones.
```

- *POSE*- una pose es una posición completamente especificada en un espacio cartesiano 3-D. En el controlador de movimiento LinuxCNC, cuando nos referimos a una pose nos referimos a una estructura EmcPose, que contiene seis coordenadas lineales (X, Y, Z, U, V y W) y tres angulares (A, B y C).
- *coord*, o modo coordinado, significa que todas las articulaciones están sincronizadas y se mueven juntas según lo ordenado por el código de nivel superior. Es el modo normal al mecanizar. En el modo coordinado, se supone que los comandos se dan en el marco de referencia cartesiano, y si la máquina no es cartesiana, los comandos son traducidos por la cinemática para impulsar cada articulación en el espacio articular según sea necesario.
- *free*, o modo libre, significa que los comandos se interpretan en el espacio articular. Se usa para mover manualmente (jog) articulaciones individuales, aunque no impide que se muevan múltiples articulaciones a la vez (creo). El homing también se realiza en modo libre; de hecho, las máquinas con cinemática no trivial deben ser homeadas antes de que puedan pasar al modo coord o teleop.
- *teleop* es el modo que probablemente necesite si está haciendo *jogging* con un hexápodo. Los comandos de jog implementados por el controlador de movimiento son jogs articulares, que funcionan en modo free. Pero si desea mover un hexápodo o una máquina similar a lo largo de un eje cartesiano en particular, debe operar más de una articulación. Para eso está *teleop*.

2.4 Descripción general de la arquitectura

Hay cuatro componentes contenidos en la Arquitectura LinuxCNC: un controlador de movimiento (EMCMOT), un controlador de E/S discreto (EMCIO), un ejecutor de tareas que los coordina (EMCTASK) y varios interfaces de usuario en modos texto y gráficos. Cada uno de ellos se describirá en el presente documento, tanto desde el punto de vista del diseño como del punto de vista de los desarrolladores (dónde encontrar los datos necesarios, cómo ampliar/modificar cosas fácilmente, etc.).

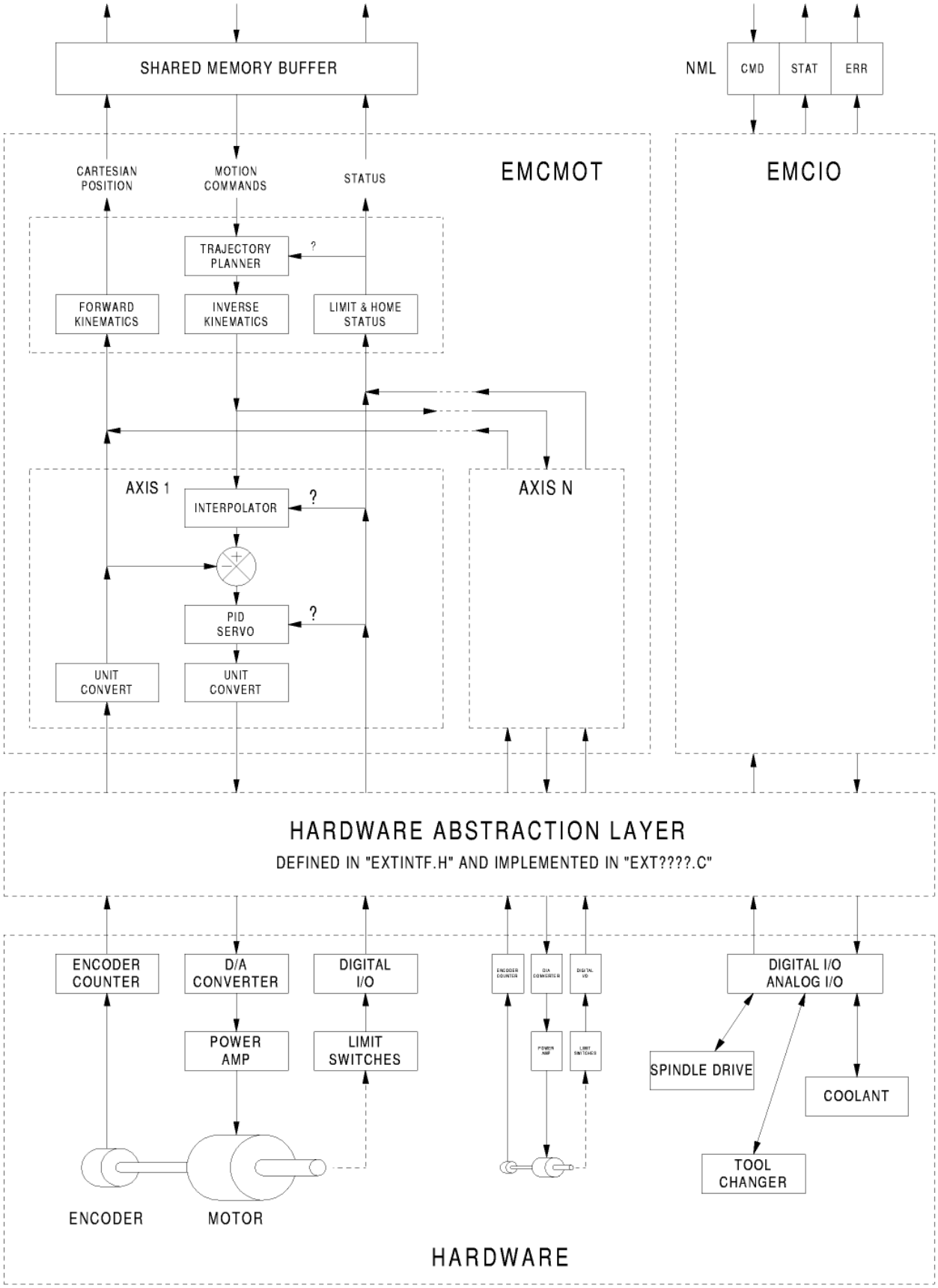

GUI
NML
CMD
STAT
ERR
EMCTASK
INTERPRETE
RS-274
(GCODE)
LOGICA
SECUENCIAL
NML
CMD
STAT
ERR
NML
CMD
STAT
ERR
REALTIME
BUFER MEMORIA COMPARTIDA
FIFOS?
POSICION
CARTESIANA
COMANDOS
MOVIMIENTO
STATUS
EMCMOT
PLANIFICADOR
TRAYECTORIAS
CINEMATICA
DIRECTA
CINEMATICA
INVERSA
STATUS HOME
Y LIMITES
INTERPOLADOR
EJE 1
PID
CONVERSOR
UNIDADES
CONVERSOR
UNIDADES
EJE N
H A L
CONTADOR
ENCODER
CONVERSOR
D/A
SWITCHES
LIMITES
AMPLIFICADOR
POTENCIA
ENCODER
MOTOR
EJE 1
EJE N
DISPOSITIVOS HARDWARE REALTIME
EMCIO
CONTROLADOR
COORDINACION
CONTROLADOR
HUSILLO
CONTROLADOR
AUXILIARES
CONTROLADOR
REFRIGERANTE
H A L
DISPOSITIVOS HARDWARE NO-REALTIME

2.5 Arquitectura del software LinuxCNC.

Al nivel más general, LinuxCNC es un jerarquía de tres controladores: el manejador de comandos a nivel de tarea y programa intérprete, el controlador de movimiento y el controlador de E/S discretas. El controlador de E/S discretas se implementa como una jerarquía de controladores, en este caso para husillo, refrigerante y subsistemas auxiliares (p. ej., Estop, lubricante). El controlador de tareas coordina las acciones del controlador de movimiento y del controlador de E/S discretas. Sus acciones están programadas en programas de control numérico "código G y M" convencional, que son interpretados por el controlador de tareas en mensajes NML y enviados al controlador de movimiento o de E/S discretas en los momentos apropiados.

2.6 Introducción al controlador de movimiento

El controlador de movimiento recibe comandos de los módulos de espacio de usuario a través de memoria compartida y ejecuta esos comandos en tiempo real. El estado del controlador está disponible para los módulos de espacio de usuario a través de la misma área de memoria compartida. El controlador de movimiento interactúa con los motores y otro hardware utilizando HAL (Capa de Abstracción de Hardware). Este documento asume que el lector tiene una comprensión básica de HAL, y comprende términos como pines HAL, señales HAL, etc., por lo que no se explican. Para obtener más información sobre HAL, consulte el Manual HAL. Otro capítulo de este documento entrará eventualmente en las interioridades del propio HAL, pero en este capítulo, solo usamos la API HAL, definida en src/hal/hal.h.


SHARED MEMORY BUFFER
NML
CMD
STAT
ERR
CARTESIAN POSITION
MOTION COMMANDS
STATUS
EMCMOT
EMCIO
TRAJECTORY PLANNER
?
FORWARD KINEMATICS
INVERSE KINEMATICS
LIMIT & HOME STATUS
AXIS 1
INTERPOLATOR
?
AXIS N
PID SERVO
?
UNIT CONVERT
UNIT CONVERT
HARDWARE ABSTRACTION LAYER
DEFINED IN "EXTINTF.H" AND IMPLEMENTED IN "EXT????.C"
ENCODER COUNTER
D/A CONVERTER
DIGITAL I/O
POWER AMP
LIMIT SWITCHES
DIGITAL I/O ANALOG I/O
SPINDLE DRIVE
COOLANT
TOOL CHANGER
ENCODER
MOTOR
HARDWARE

2.7 Diagrama de bloque y flujo de datos

La siguiente figura es un diagrama de bloques de un controlador articular. Hay un controlador por articulación. Los controladores articulares funcionan a un nivel más bajo que la cinemática; un nivel donde todas las articulaciones son completamente independientes. Todos los datos para una articulación está en una sola estructura articular. Algunos miembros de esa estructura son visible en el diagrama de bloques, como coarse_pos, pos_cmd y motor_pos_fb.

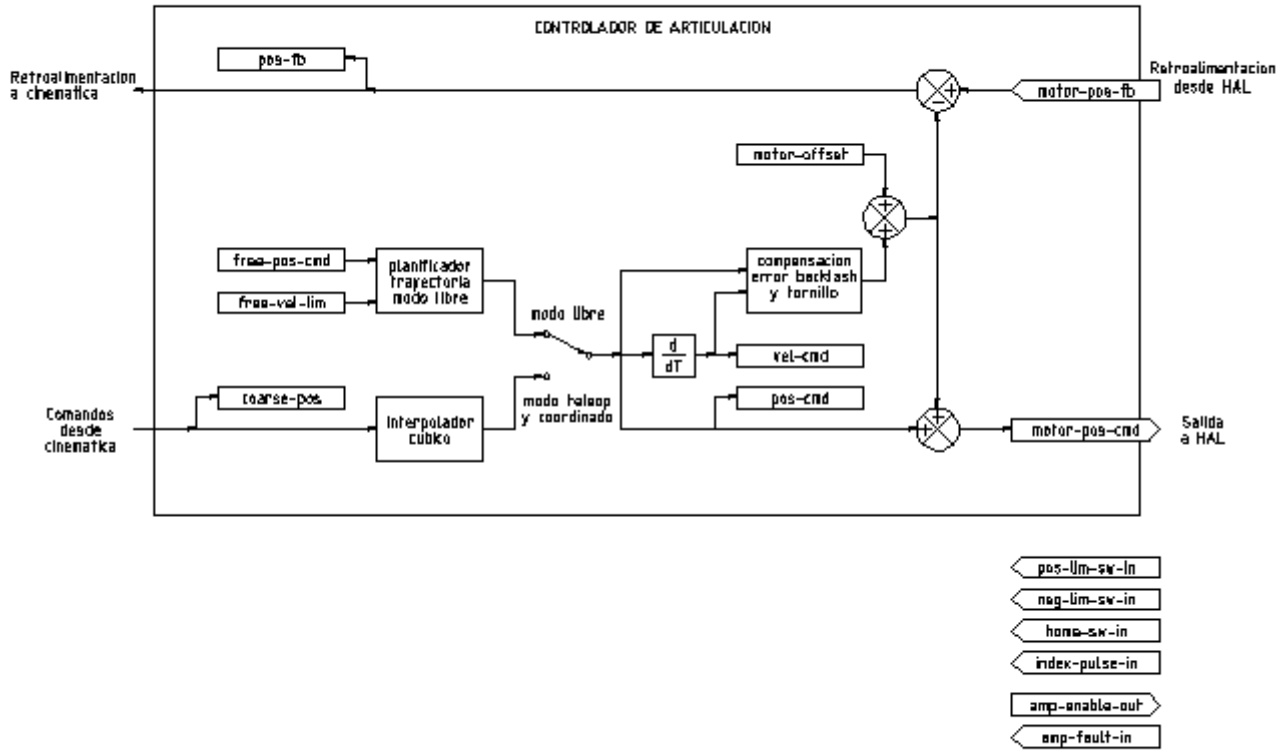


Diagrama de bloque del controlador articular La figura anterior muestra cinco de los siete conjuntos de información de posición que forman el flujo principal de datos a través del controlador de movimiento. Las siete formas de datos de posición son las siguientes:

1. *emcmotStatus->carte_pos_cmd* - Esta es la posición deseada, en coordenadas cartesianas. Se actualiza a tasa traj, no a tasa servo. En modo coord, se determina por el planificador traj. En modo teleop, está determinado por el planificador traj?. En modo libre, es copiado de actualPos, o generado mediante la aplicación de cinemática directa a (2) o (3)
2. *emcmotStatus->joints[n].coarse_pos* - Esta es la posición deseada, en coordenadas articulares, pero antes de interpolación. Se actualiza a tasa traj, no a tasa servo. En modo coord, se genera aplicando cinematica inversa a (1). En modo teleop, se genera aplicando cinemática inversa a (1). En modo libre, creo que se copia de (3).
3. *emcmotStatus->joints[n].pos_cmd* - Esta es la posición deseada, en coordenadas articulares, después de interpolación. En cada período servo, se genera un nuevo conjunto de estas coordenadas. En modo coord, se genera a partir de (2) por el interpolador. En modo teleop, se genera a partir de (2) por el interpolador. En modo libre, es generado por el planificador traj de modo libre.
4. *emcmotStatus->joints[n].motor_pos_cmd* - Esta es la posición deseada, en coordenadas de motor. Las coordenadas del motor se generan agregando compensación backlash, compensación de error del tornillo de avance y offset (para homing) a (3). Se genera de la misma manera independientemente del modo, y es la salida al lazo PID u otro bucle de posición.

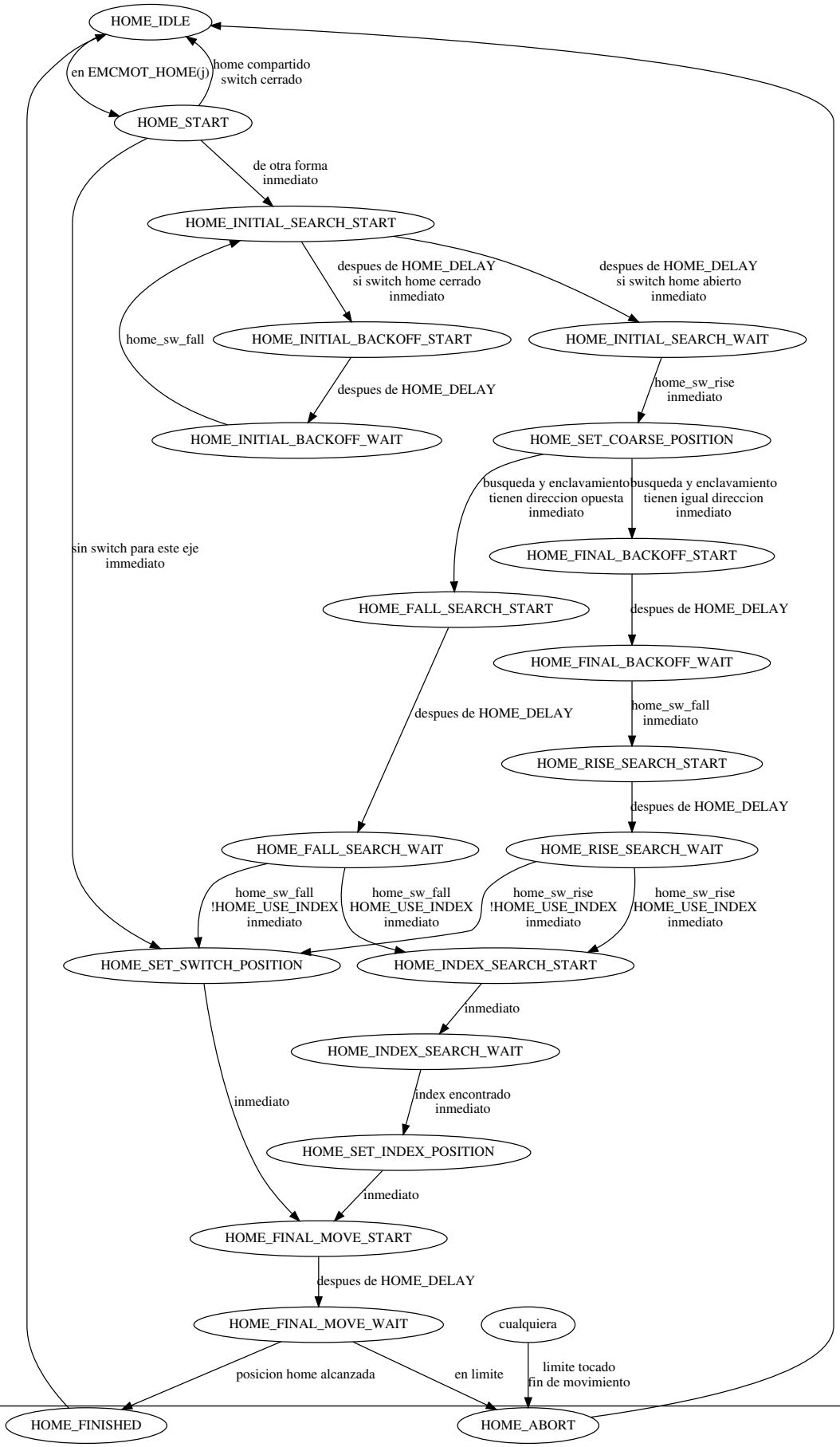
5. *emcmotStatus->joints[n].motor_pos_fb* - Esta es la posición real, en en coordenadas de motor. Es la entrada de codificadores u otro dispositivo de retroalimentación (o desde codificadores virtuales en máquinas de bucle abierto). Es "generado" por la lectura del dispositivo de retroalimentación.

6. *emcmotStatus->joints[n].pos_fb*: esta es la posición real, en coordenadas articulares. Se genera restando offsets, compensación de error del tornillo de avance y compensación de backlash de (5). Se genera del mismo modo, independientemente del modo operativo.

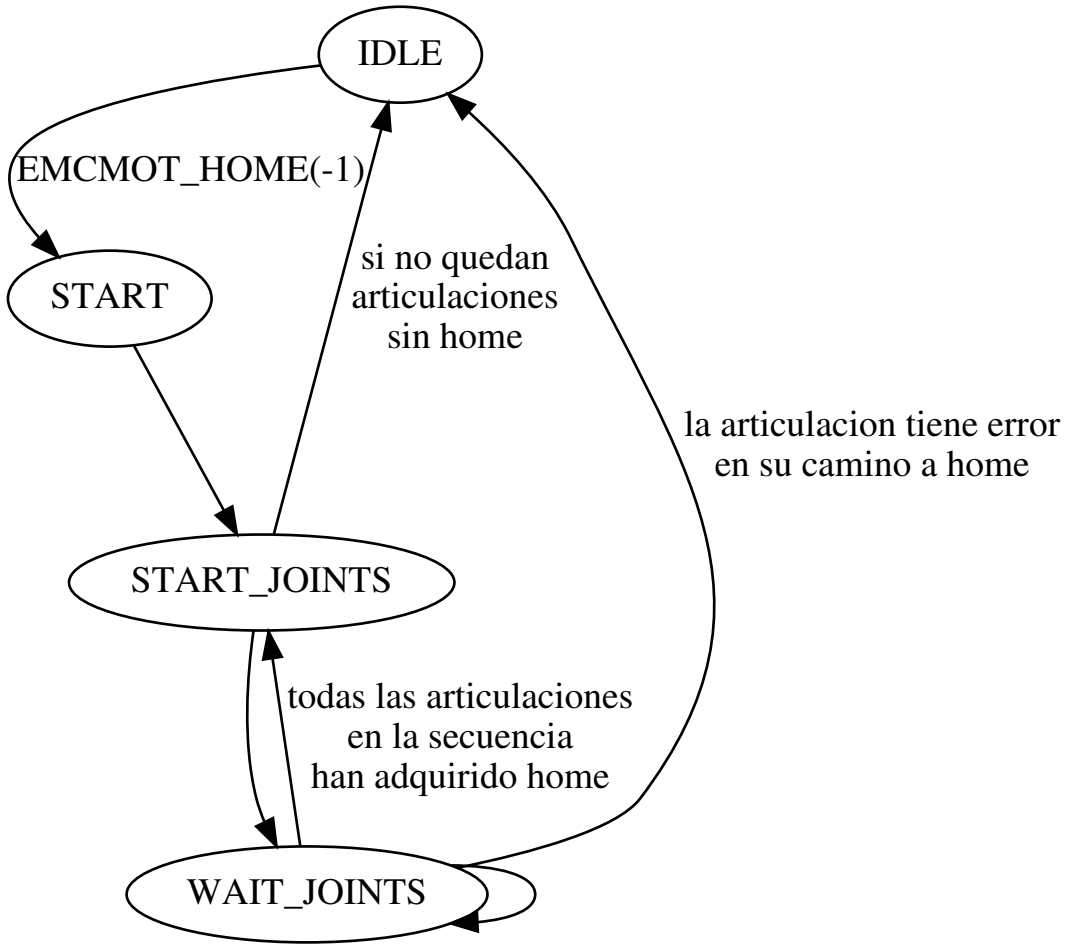
7. *emcmotStatus->carte_pos_fb* - Esta es la posición real, en coordenadas cartesianas. Se actualiza a tasa traj, no a tasa servo. Idealmente, actualPos siempre se calcularía aplicando cinemática directa a (6). Sin embargo, la cinemática directa puede no estar disponible, o pueden ser inutilizable porque uno o más ejes no están homeados. En ese caso, las opciones son: A) fingirla, copiando (1), o B) admitir que realmente no se conocen las coordenadas cartesianas, y simplemente no actualizar actualPos. Cualquiera que sea el enfoque utilizado, no veo ninguna razón para no hacerlo de la misma manera, independientemente del modo de operación. Yo propondría lo siguiente; si hay cinemática directa, usarla, a menos que no funcionen debido a ejes sin home u otros problemas, en cuyo caso hacer (B). Si no hay cinemática directa, hacer (A), ya que de lo contrario actualPos *nunca* obtendrá actualización.

2.8 Homing

2.8.1 Diagrama de estado de homing



2.8.2 Otro diagrama de homing



2.9 Comandos

Esta sección simplemente enumera todos los comandos que se pueden enviar al módulo de movimiento, junto con explicaciones detalladas de lo que hacen. los los nombres de los comandos se definen en cmd_code_t, que es un tipo definido por código, cada nombre de comando comienza con *EMCMOT_*, que se omite aquí).

Los comandos se implementan mediante una gran instrucción switch en la función emcmotCommandHandler() del archivo fuente src/emc/motion/command.c, que se llama a la tasa servo. Más sobre esa función más adelante. N.T. Esta ultima afirmación es erronea. No se vuelve a hablar de emcmotCommandHandler(), ni en este texto, ni en el resto del Manual del Desarrollador.

Hay aproximadamente 44 comandos: esta lista todavía está bajo construcción.

N.T. La enumeracion cmd_code_t, en motion.h, contiene 73 comandos (a 6-5-2020) N.T. La instrucción switch en command.c contempla 70 N.T. Los comandos ENABLE_WATCHDOG / DISABLE_WATCHDOG están en motion-logger.c. Quizas sean obsoletos. N.T. El comando SET_TELEOP_VECTOR solo aparece en motion-logger.c, sin mas efecto que su propio log.

2.9.1 ABORT

El comando ABORT simplemente detiene todo movimiento. Se puede emitir en cualquier momento y siempre será aceptado. No deshabilita el controlador de movimiento ni cambia ninguna información de estado; simplemente cancela cualquier movimiento que esté actualmente en progreso. [1] Actuá conforme al modo actual. En modo teleop, asigna cero a las velocidades deseadas.

[1] Parece que el código de nivel superior (TASK y superior) también usa ABORT para borrar fallos. Siempre que haya un fallo persistente (como estar fuera de interruptores hardware de límite), el código de nivel superior envía un constante flujo de ABORT al controlador de movimiento en su intento de sobrepasar el fallo. Miles de ellos … Eso significa que el controlador de movimiento debe evitar fallos persistentes. Esto necesita ser investigado.

??? En modo coordinado, llama a la función abort del planificador de trayectorias tpAbort()

2.9.1.1 Requisitos

Ninguno. El comando ABORT siempre se acepta y actúa inmediatamente.

2.9.1.2 Resultados

En modo libre, los planificadores de trayectoria de modo libre quedan deshabilitados. Esto da como resultado que cada articulación se detenga tan rápido como su límite de aceleración (desaceleración) permita. La parada no está coordinada. En modo teleop, la velocidad cartesiana comandada se establece a cero. No sé exactamente qué tipo de parada resulta (coordinada, descoordinada, etc.), pero lo resolveré finalmente. En modo coord, se le dice al planificador de trayectoria del modo coord que aborte el movimiento actual. De nuevo, no sé el resultado exacto de esto, pero lo documentaré cuando lo resuelva.

2.9.2 FREE

El comando FREE pone el controlador de movimiento en modo libre. Modo libre significa que cada articulación es independiente de todas las demás articulaciones. cartesiano Las coordenadas, poses y cinemática se ignoran cuando está en modo libre. En esencia, cada articulación tiene su propio planificador de trayectoria simple, y cada articulación ignora por completo las otras articulaciones. Algunos comandos (como Joint JOG y HOME) solo funcionan en modo libre. Otros comandos, incluso cualquier cosa que trata con coordenadas cartesianas, no funciona en absoluto en modo libre.

2.9.2.1 Requisitos

El controlador de comandos no aplica requisitos al comando FREE, Siempre será aceptado. Sin embargo, si alguna articulación está en movimiento (GET_MOTION_INPOS_FLAG () == FALSE), entonces el comando será ignorado. Este comportamiento está controlado por un código que ahora se encuentra en la función *set_operating_mode ()* en control.c, ese código debe limpiarse. Creo que el comando no debe ignorarse en silencio, sino que El controlador de comandos debe determinar si se puede ejecutar y devolver un error si no puede

2.9.2.2 Resultados

Si la máquina ya está en modo libre, nada. De lo contrario, el La máquina se coloca en modo libre. La trayectoria del modo libre de cada articulación el planificador se inicializa en la ubicación actual de la articulación, pero el los planificadores no están habilitados y las articulaciones son estacionarias.

2.9.3 TELEOP

El comando TELEOP coloca la máquina en modo de teleoperación. En teleop modo, el movimiento de la máquina se basa en coordenadas cartesianas utilizando cinemática, en lugar de en articulaciones individuales como en modo libre. sin embargo el planificador de trayectoria per se no se usa, en cambio el movimiento es controlado por un vector de velocidad. El movimiento en modo teleop es muy parecido a trotar, excepto que se hace en espacio cartesiano en lugar de articulación espacio. En una máquina con cinemática trivial, hay poca diferencia entre el modo teleop y el modo libre, y las GUI para esas máquinas podrían Ni siquiera emita este comando. Sin embargo, para máquinas no triviales como robots y hexápodos, el modo teleop se utiliza para la mayoría de los jog ordenados por el usuario movimientos de tipo

2.9.3.1 Requisitos

El controlador de comandos rechazará el comando TELEOP con un error mensaje si la cinemática no se puede activar porque uno o más las articulaciones no han sido dirigidas. Además, si alguna articulación está en movimiento (GET_MOTION_INPOS_FLAG () == FALSE), entonces el comando será ignorado (sin mensaje de error). Este comportamiento está controlado por un código que es ahora ubicado en la función *set_operating_mode ()* en control.c. yo cree que el comando no debe ser ignorado en silencio, sino el comando El controlador debe determinar si se puede ejecutar y devolver un error si no puede

2.9.3.2 Resultados

Si la máquina ya está en modo teleop, nada. De lo contrario el la máquina se coloca en modo teleop. El código cinemático está activado, los interpoladores son drenados y enjuagados, y la velocidad cartesiana los comandos se ponen a cero.

2.9.4 COORD

El comando COORD coloca la máquina en modo coordinado. En coord modo, el movimiento de la máquina se basa en coordenadas cartesianas utilizando cinemática, en lugar de en articulaciones individuales como en modo libre. En Además, el planificador de trayectoria principal se utiliza para generar movimiento, basado en los comandos LINE, CIRCLE y / o PROBE en cola. El modo coord es el modo que se usa al ejecutar un programa de código G.

2.9.4.1 Requisitos

El controlador de comandos rechazará el comando COORD con un error mensaje si la cinemática no se puede activar porque uno o más las articulaciones no han sido dirigidas. Además, si alguna articulación está en movimiento (GET_MOTION_INPOS_FLAG () == FALSE), entonces el comando será ignorado (sin mensaje de error). Este comportamiento está controlado por un código que es ahora ubicado en la función *set_operating_mode ()* en control.c. yo cree que el comando no debe ser ignorado en silencio, sino el comando El controlador debe determinar si se puede ejecutar y devolver un error si no puede

2.9.4.2 Resultados

Si la máquina ya está en modo coord, nada. De lo contrario, el La máquina se coloca en modo coord. El código cinemático está activado, los interpoladores son drenados y enjuagados, y el planificador de trayectoria las colas están vacías El planificador de trayectoria está activo y en espera de una LÍNEA, CÍRCULO o comando SONDA.

2.9.5 ENABLE

El comando ENABLE habilita el controlador de movimiento.

2.9.5.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.5.2 Resultados

Si el controlador ya está habilitado, nada. Si no, el controlador está habilitado. Las colas y los interpoladores se sonrojan. Cualquier movimiento o las operaciones de referencia se terminan. Las salidas de habilitación de amplificador asociadas con articulaciones activas se encienden. Si la cinemática hacia adelante no es disponible, la máquina se cambia al modo libre.

2.9.6 DISABLE

El comando DISABLE deshabilita el controlador de movimiento.

2.9.6.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.6.2 Resultados

Si el controlador ya está deshabilitado, nada. Si no, el controlador está desactivado. Las colas y los interpoladores se sonrojan. Cualquier movimiento o las operaciones de referencia se terminan. Las salidas de habilitación de amplificador asociadas con las articulaciones activas están apagadas. Si la cinemática hacia adelante no es disponible, la máquina se cambia al modo libre.

2.9.7 ENABLE_AMPLIFIER (JOINT_ENABLE_AMPLIFIER)

El comando ENABLE_AMPLIFIER activa la salida de habilitación del amplificador para un Amplificador de salida única, sin cambiar nada más. Puede ser usado para habilitar un controlador de velocidad del husillo.

2.9.7.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.7.2 Resultados

Actualmente nada. (Una llamada a la antigua función extAmpEnable es actualmente comentado). Eventualmente configurará el pin de habilitación del amplificador HAL cierto.

2.9.8 DISABLE_AMPLIFIER (JOINT_DISABLE_AMPLIFIER)

El comando DISABLE_AMPLIFIER apaga la salida de habilitación del amplificador para un Amplificador único, sin cambiar nada más. De nuevo, útil para Controladores de velocidad del husillo.

2.9.8.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.8.2 Resultados

Actualmente nada. (Una llamada a la antigua función extAmpEnable es actualmente comentado). Eventualmente configurará el pin de habilitación del amplificador HAL falso.

2.9.9 ACTIVATE_JOINT (JOINT_ACTIVATE)

El comando ACTIVATE_JOINT activa todos los cálculos asociados. con una sola articulación, pero no cambia la salida de habilitación del amplificador de la articulación alfiler.

2.9.9.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.9.2 Resultados

Los cálculos para la articulación especificada están habilitados. El pin de habilitación del amplificador no se cambia, sin embargo, cualquier comando ENABLE o DISABLE posterior modificar el pin de habilitación del amplificador de la articulación.

2.9.10 DEACTIVATE_JOINT (JOINT_DEACTIVATE)

El comando DEACTIVATE_JOINT desactiva todos los cálculos asociados. con una sola articulación, pero no cambia la salida de habilitación del amplificador de la articulación alfiler.

2.9.10.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.10.2 Resultados

Los cálculos para la articulación especificada están habilitados. El pin de habilitación del amplificador no se cambia, y los siguientes comandos ENABLE o DISABLE no modifique el pin de habilitación del amplificador de la articulación.

2.9.11 ENABLE_WATCHDOG

El comando ENABLE_WATCHDOG habilita un perro guardián basado en hardware (si presente).

2.9.11.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.11.2 Resultados

Actualmente nada. El viejo perro guardián era una cosa extraña que usaba una tarjeta de sonido específica. Se puede diseñar una nueva interfaz de vigilancia en el futuro.

2.9.12 DISABLE_WATCHDOG

El comando DISABLE_WATCHDOG deshabilita un perro guardián basado en hardware (si presente).

2.9.12.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.12.2 Resultados

Actualmente nada. El viejo perro guardián era una cosa extraña que usaba un Tarjeta de sonido específica. Se puede diseñar una nueva interfaz de vigilancia en el futuro.

2.9.13 PAUSE

El comando PAUSE detiene el planificador de trayectoria. No tiene efecto en modo libre o teleop. En este punto no sé si detiene todo el movimiento inmediatamente, o si completa el movimiento actual y luego se detiene antes tirando de otro movimiento de la cola.

2.9.13.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.13.2 Resultados

El planificador de trayectoria hace una pausa.

2.9.14 RESUME

El comando RESUME reinicia el planificador de trayectoria si está en pausa. Eso no tiene efecto en modo libre o teleop, o si el planificador no está en pausa.

2.9.14.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.14.2 Resultados

Se reanuda el planificador de trayectoria.

2.9.15 STEP

El comando STEP reinicia el planificador de trayectoria si está en pausa, y le dice al planificador que se detenga nuevamente cuando llegue a un punto específico. Eso no tiene efecto en modo libre o teleop. En este punto no se exactamente cómo funciona esto. Agregaré más documentación aquí cuando excave más profundo en el planificador de trayectoria.

2.9.15.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.15.2 Resultados

El planificador de trayectoria se reanuda y luego se detiene cuando llega a un punto específico.

2.9.16 SCALE (SPINDLE_SCALE para husillo, FEED_SCALE para avance, RAPID_SCALE para rapidos)

```
N.T. El siguiente párrafo necesita nueva redacción
```

El comando SCALE escala todos los límites de velocidad y comandos por un cantidad especificada Se utiliza para implementar la anulación de la velocidad de alimentación y otros funciones similares El escalado funciona en modo libre, teleop y coord, y afecta todo, incluidas las velocidades de referencia, etc. Sin embargo, los límites individuales de velocidad conjunta no se ven afectados.

2.9.16.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado.

2.9.16.2 Resultados

Todos los comandos de velocidad son escalados por la constante especificada.

2.9.17 OVERRIDE_LIMITS

El comando OVERRIDE_LIMITS evita que los límites se disparen hasta que Fin del siguiente comando JOG. Normalmente se usa para permitir que una máquina salir de un interruptor de límite después de disparar. (El comando puede en realidad se puede usar para anular límites o para cancelar una anulación anterior).

2.9.17.1 Requisitos

Ninguna. El comando puede emitirse en cualquier momento y siempre será aceptado. (Creo que solo debería funcionar en modo libre).

2.9.17.2 Resultados

Los límites en todas las articulaciones se anulan hasta el final del próximo JOG mando. (Esto está roto actualmente ... una vez que un comando OVERRIDE_LIMITS se recibe, los límites se ignoran hasta que otro comando OVERRIDE_LIMITS los vuelve a habilitar).

2.9.18 HOME (JOINT_HOME)

El comando HOME inicia una secuencia de referencia en una articulación especificada. los La secuencia de referencia real está determinada por una serie de configuraciones parámetros, y puede variar desde simplemente establecer la posición actual hasta cero, a una búsqueda en varias etapas para un interruptor de inicio y pulso de índice, seguido de un traslado a una ubicación de inicio arbitraria. Para más información sobre la secuencia de referencia, consulte la sección de referencia del Manual del integrador.

2.9.18.1 Requisitos

El comando se ignorará en silencio a menos que la máquina esté en modo libre.

2.9.18.2 Resultados

Se anula cualquier movimiento u otro movimiento conjunto, y la secuencia de referencia empieza.

2.9.19 JOG_CONT

El comando JOG_CONT inicia un avance continuo en una sola articulación. UNA el avance continuo se genera al establecer la trayectoria del modo libre posición objetivo del planificador hasta un punto más allá del final de la articulación rango de viaje. Esto asegura que el planificador se moverá constantemente hasta que sea detenido por los límites conjuntos o por un comando ABORTAR. Normalmente, una GUI envía un comando JOG_CONT cuando el usuario presiona un jog botón, y ABORTAR cuando se suelta el botón.

2.9.19.1 Requisitos

El controlador de comandos rechazará el comando JOG_CONT con un error mensaje si la máquina no está en modo libre, o si alguna junta está en movimiento (GET_MOTION_INPOS_FLAG () == FALSE), o si el movimiento no está habilitado. Eso también ignorará silenciosamente el comando si la articulación ya está en o más allá de su límite y el trote ordenado lo empeoraría.

2.9.19.2 Resultados

El planificador de trayectoria de modo libre para la articulación identificada por El eje emcmotCommand \ -> está activado, con una posición de destino más allá del final de recorrido conjunto, y un límite de velocidad de emcmotCommand \ -> vel. Esta comienza el movimiento de la articulación, y el movimiento continuará hasta que se detenga ABORTAR el comando o al alcanzar un límite. El planificador de modo libre acelera en el límite de aceleración conjunta al comienzo del movimiento, y desacelerar en el límite de aceleración conjunta cuando se detiene.

2.9.20 JOG_INCR

El comando JOG_INCR inicia un avance gradual en una sola articulación. Los jogs incrementales son acumulativos, en otras palabras, emiten dos JOG_INCR comandos que cada uno pide 0.100 pulgadas de movimiento resultarán en 0.200 pulgadas de recorrido, incluso si el segundo comando se emite antes del el primero termina. Normalmente, los jogs incrementales se detienen cuando tienen recorrieron la distancia deseada, sin embargo, también se detienen cuando golpean un límite, o en un comando ABORTAR.

2.9.20.1 Requisitos

El controlador de comandos rechazará silenciosamente el comando JOG_INCR si la máquina no está en modo libre, o si alguna junta está en movimiento (GET_MOTION_INPOS_FLAG () == FALSE), o si el movimiento no está habilitado. Eso también ignorará silenciosamente el comando si la articulación ya está en o más allá de su límite y el trote ordenado lo empeoraría.

2.9.20.2 Resultados

El planificador de trayectoria de modo libre para la articulación identificada por emcmotCommand \ -> el eje está activado, la posición de destino es incrementado / decrementado por emcmotCommand \ -> offset, y la velocidad el límite se establece en emcmotCommand \ -> vel. El planificador de trayectoria de modo libre generará un movimiento trapezoidal suave desde la posición actual hasta La posición de destino. El planificador puede manejar correctamente los cambios en posición objetivo que ocurre mientras el movimiento está en progreso, por lo que múltiples Los comandos JOG_INCR se pueden emitir en rápida sucesión. El modo libre el planificador acelera en el límite de aceleración conjunta al comienzo del mover, y desacelerará en el límite de aceleración conjunta para detenerse en el posición de objetivo.

2.9.21 JOG_ABS

El comando JOG_ABS inicia un desplazamiento absoluto en una sola articulación. Un trotar absoluto es un simple movimiento a una ubicación específica, en conjunto coordenadas Normalmente los trotes absolutos se detienen cuando alcanzan el deseado ubicación, sin embargo, también se detienen cuando alcanzan un límite, o en un ABORT mando.

2.9.21.1 Requisitos

El controlador de comandos rechazará silenciosamente el comando JOG_ABS si la máquina no está en modo libre, o si alguna junta está en movimiento (GET_MOTION_INPOS_FLAG () == FALSE), o si el movimiento no está habilitado. Eso también ignorará silenciosamente el comando si la articulación ya está en o más allá de su límite y el trote ordenado lo empeoraría.

2.9.21.2 Resultados

El planificador de trayectoria de modo libre para la articulación identificada por emcmotCommand \ -> el eje está activado, la posición de destino se establece en emcmotCommand \ -> offset, y el límite de velocidad se establece en emcmotCommand \ -> vel. El planificador de trayectoria de modo libre generará un movimiento trapezoidal suave desde la posición actual hasta el objetivo posición. El planificador puede manejar correctamente los cambios en el objetivo posición que sucede mientras el movimiento está en progreso. Si varios JOG_ABS los comandos se emiten en rápida sucesión, cada nuevo comando cambia el posición de destino y la máquina pasa a la posición final ordenada. El planificador de modo libre acelera en el límite de aceleración conjunta en el comienzo del movimiento, y se desacelerará en el límite de aceleración conjunta para detenerse en la posición de destino.

2.9.22 SET_LINE

El comando SET_LINE agrega una línea recta al planificador de trayectoria cola.

(Más tarde)

2.9.23 SET_CIRCLE

El comando SET_CIRCLE agrega un movimiento circular al planificador de trayectoria cola.

(Más tarde)

2.9.24 SET_TELEOP_VECTOR

El comando SET_TELEOP_VECTOR indica al controlador de movimiento que se mueva a lo largo de un vector específico en el espacio cartesiano.

(Más tarde)

2.9.25 PROBE

El comando PROBE indica al controlador de movimiento que se mueva hacia un punto específico en el espacio cartesiano, deteniendo y grabando su posición si se activa la entrada de la sonda.

(Más tarde)

2.9.26 CLEAR_PROBE_FLAG

El comando CLEAR_PROBE_FLAG se usa para restablecer la entrada de la sonda en preparación para un comando PROBE. (Pregunta: ¿por qué no debería la SONDA? comando restablecer automáticamente la entrada?)

(Más tarde)

2.9.27 SET_xxx

Hay aproximadamente 15 comandos SET_xxx, donde xxx es el nombre de algún parámetro de configuración. Se anticipa que habrá varios comandos SET más a medida que se agregan más parámetros. me gustaría encuentre una forma más limpia de establecer y leer los parámetros de configuración. Los métodos existentes requieren que se agreguen muchas líneas de código a múltiples archivos cada vez que se agrega un parámetro. Gran parte de ese código es idéntico o casi idéntico para cada parámetro.

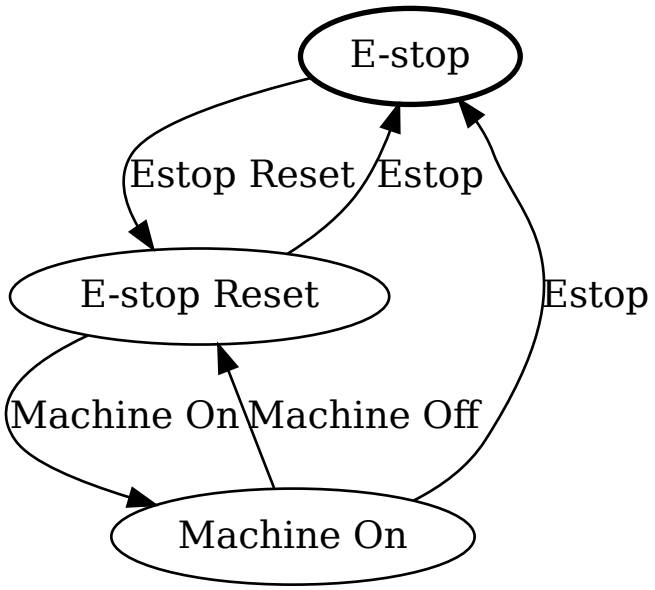
2.10 Compensación de error de tornillo y backlash

+

2.11 Controlador de tareas (EMCTASK)

2.11.1 Estado

Task tiene tres estados internos posibles: **E-stop**, **E-stop Reset**, y **Machine on**.



2.12 Controlador IO (EMCIO)

El controlador de E/S es un módulo separado que acepta comandos NML de TASK.
Interactúa con E/S externas utilizando pines HAL.
iocontrol.cc se carga a través del script linuxcnc antes de TASK.
Actualmente hay dos versiones de iocontrol. La segunda versión maneja los errores de hardware de cambio de herramienta

Actualmente ESTOP/Enable, el refrigerante, el lubricante y el cambio de herramienta se manejan con
iocontrol. Estos son eventos de velocidad relativamente baja; las E/S coordinadas de alta velocidad se manejan en motion.

emctaskmain.cc envía comandos de E/S a través de taskclass.cc
Las funciones de Taskclass envían mensajes NML a iocontrol.cc
taskclass usa los comandos definidos en c ++ en su archivo o,
si está definido, ejecuta comandos basados en python definidos en archivos proporcionados por el usuario.

Proceso del bucle principal de iocontrol:

- registros para señales SIGTERM y SIGINT del sistema operativo.
- comprueba si las entradas HAL han cambiado
- comprueba si read_tool_inputs() indica que el cambio de herramienta ha finalizado y establece emcioStatus.status
- busca mensajes NML relacionados con E/S

números de mensaje nml: de emc.hh:

```
#define EMC_IO_INIT_TYPE ((NMLTYPE) 1601)
#define EMC_TOOL_STAT_TYPE ((NMLTYPE) 1199)
#define EMC_TOOL_INIT_TYPE ((NMLTYPE) 1101)
#define EMC_TOOL_HALT_TYPE ((NMLTYPE) 1102)
#define EMC_TOOL_ABORT_TYPE ((NMLTYPE) 1103)
#define EMC_TOOL_PREPARE_TYPE ((NMLTYPE) 1104)
#define EMC_TOOL_LOAD_TYPE ((NMLTYPE) 1105)
#define EMC_TOOL_UNLOAD_TYPE ((NMLTYPE) 1106)
#define EMC_TOOL_LOAD_TOOL_TABLE_TYPE ((NMLTYPE) 1107)
#define EMC_TOOL_SET_OFFSET_TYPE ((NMLTYPE) 1108)
#define EMC_TOOL_SET_NUMBER_TYPE ((NMLTYPE) 1109)
#define EMC_TOOL_START_CHANGE_TYPE ((NMLTYPE) 1110)
```

2.13 Interfaces de usuario

+

2.14 Introducción a libnml

libnml se deriva de rcslib del NIST sin todos los apoyos para otras plataformas. Muchos de los contenedores del código específico de plataformas han sido eliminados, junto con gran parte del código que no es requerido por LinuxCNC. Está Se espera que quedara suficiente compatibilidad con rcslib para que las aplicaciones puedan implementarse en plataformas que no sean Linux y aún ser capaz de comunicarse con LinuxCNC.

Este capítulo no pretende ser una guía definitiva para usar libnml (o rcslib); en cambio proporcionará una visión general de cada clase C++ y sus funciones miembro. Inicialmente, la mayoría de estas notas se agregarán como comentarios aleatorios a medida que el código se analice y modifique.

2.15 LinkedList

Clase base para mantener una lista enlazada. Este es uno de los principales bloques utilizados para pasar mensajes NML y estructuras de datos internas variadas.

2.16 LinkedListNode

Clase base para producir una lista enlazada. Su propósito es mantener punteros a los nodos anteriores y siguientes, puntero a los datos y el tamaño de los datos.

No asigna memoria para el almacenamiento de datos.

2.17 SharedMemory

Proporciona un bloque de memoria compartida junto con un semáforo (heredado de la clase Semaphore). La creación y destrucción del semáforo es manejado por el constructor y destructor SharedMemory.

2.18 ShmBuffer

Clase para pasar mensajes NML entre procesos locales mediante memoria intermedia de uso compartido. Gran parte del funcionamiento interno se hereda de la clase CMS.

2.19 Timer

La clase Timer proporciona un temporizador periódico limitado solo por la resolución del reloj del sistema. Si, por ejemplo, un proceso necesita ser ejecutado cada 5 segundos, independientemente del tiempo que lleve ejecutar el proceso, el siguiente fragmento de código muestra cómo hacerlo:

```
main()
{
    timer = new Timer(5.0);    /* Inicializa un temporizador con un ciclo de 5 segundos */
    while(0) {
        / * Hacer algún proceso * /
        timer.wait(); /* Espera hasta el siguiente intervalo de 5 segundos */
    }
    delete timer;
}
```

2.20 Semaphore

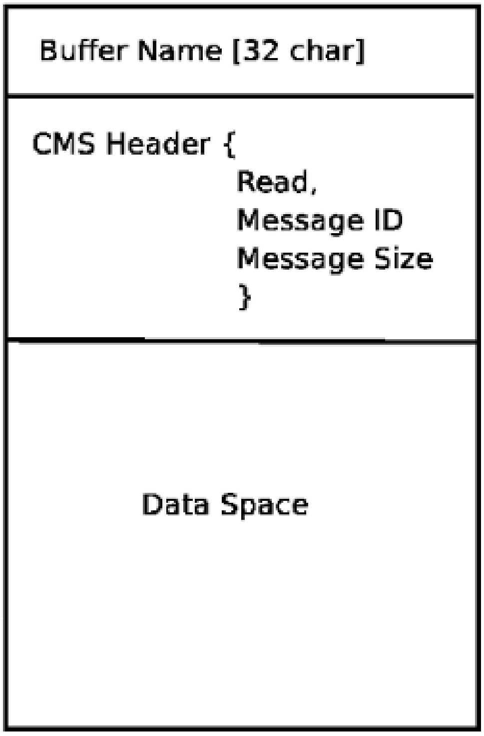
La clase Semaphore proporciona un método de exclusiones mutuas para acceder a un recurso compartido. La función para obtener un semáforo puede bloquear hasta que el acceso esté disponible, regresar después de un tiempo de espera o regresar inmediatamente con o sin obtener el semáforo. El constructor crear un semáforo o adjuntar a uno existente si la ID ya está en uso.

Semaphore::destroy() debe ser invocado solo por el último proceso.

2.21 CMS

En el corazón de libnml está la clase CMS. Contiene la mayor parte de funciones utilizadas por libnml y finalmente NML. Muchos de las funciones internas se sobrecargan para permitir métodos de paso de datos dependientes de hardware específico. En definitiva, todo gira en torno a un bloque central de memoria (denominado "búfer de mensajes" o simplemente *buffer*). Este búfer puede existir como un bloque de memoria compartida accedida por otros procesos CMS/NML, o un búfer local y privado para la transmisión de datos por red o interfaces seriales.

El búfer se asigna dinamicamente en tiempo de ejecución para permitir una mayor flexibilidad del subsistema CMS/NML. El tamaño del búfer debe ser suficientemente grande para acomodar el mensaje más grande, una pequeña cantidad para mensajes internos y permitir que el mensaje se codifique si se elige esta opción (los datos codificados se cubrirán más adelante). La siguiente figura es una vista interna del espacio del búfer.



CMS buffer La clase base de CMS es la principal responsable de crear las vías de comunicación e interfaz con el S.O.

// ////////////////////////////// == Notas NML /* FIX ME */

Una colección de notas y pensamientos al azar mientras estudias el código libnml y rcslib.

Gran parte de esto necesita ser editado y reescrito de manera coherente antes de su publicación // ///////////////////////

2.22 Formato del archivo de configuración

La configuración NML consta de dos tipos de formatos de línea. Uno para Buffers, y un segundo para Procesos que se conectan a los buffers.

2.22.1 Línea de búfer

El formato NIST original de la línea de búfer es:

- *B nombre tipo host tamaño neut RPC# buffer# max_procs key [configuraciones específicas por tipo]*
- *B*- identifica la línea como una configuración de búfer.
- *nombre*- es el identificador del búfer.
- *tipo*- describe el tipo de búfer: SHMEM, LOCMEM, FILEMEM, PHANTOM o GLOBMEM.
- *host*- es una dirección IP o un nombre de host para el servidor NML
- *tamaño*- es el tamaño del búfer
- *neut*- un booleano para indicar si los datos en el búfer están codificados en un formato independiente de la máquina, o sin formato.
- *RPC#*- Obsoleto - placeholder retenido solo para compatibilidad con versiones anteriores.

- *buffer#*- un número de ID único que se usa si un servidor controla varios buffers.
- *max_procs*- procesos máximos permitidos para conectarse a este búfer.
- *key* - es un identificador numérico para un búfer de memoria compartida

2.22.2 Configuraciones específicas por tipo

El tipo de búfer implica opciones de configuración adicionales mientras que el sistema operativo host impide ciertas combinaciones. En una tentativa de concretar la documentación publicada en un formato coherente, solo será cubierto el tipo de buffer **SHMEM**.

- *mutex=os_sem*- modo predeterminado para proporcionar el semáforo de bloqueo de la memoria intermedia.
- *mutex=none*- no utilizado
- *mutex=no_interrupts* - no aplicable en un sistema Linux
- *mutex=no_switching* - no aplicable en un sistema Linux
- *mutex=mao split*- divide el búfer en la mitad (o más) y permite que un proceso acceda a una parte del búfer mientras que un segundo proceso está escribiendo en la otra parte.
- *TCP=(número de puerto)*- especifica qué puerto de red utilizar.
- *UDP =(número de puerto)* - ídem
- *STCP =(número de puerto)* - ídem
- *serialPortDevName=(puerto serie)* - Sin documentar.
- *passwd=file_name.pwd*- agrega una capa de seguridad al búfer requiriendo que cada proceso proporcione una contraseña.
- *bsem*- la documentación del NIST implica una clave para un semáforo de bloqueo, y si bsem=-1, se evitan los bloqueos de lectura.
- *queue*- permite pasar mensajes en cola.
- *ascii* - Codifica mensajes en formato de texto plano
- *disp*- codifica los mensajes en un formato adecuado para mostrarlos (???)
- *xdr*- codifica mensajes en Representación de Datos Externos. (Ver rpc/xdr.h para más detalles).
- *diag*- habilita almacenado de diagnósticos en el búfer (¿temporizaciones y recuentos de bytes?)

2.22.3 Línea de Proceso

El formato NIST original de la línea de proceso es:

- P nombre buffer tipo host ops server timeout master c_num [configuraciones específicas por tipo] *
- *P*- identifica esta línea como una configuración de proceso.
- *nombre*- es el identificador del proceso.
- *buffer* - es uno de los buffers definidos en otra parte del archivo de configuración.
- *tipo*- define si este proceso es local o remoto en relación con el búfer.
- *host*- especifica en qué parte de la red se está ejecutando este proceso.
- *ops*- proporciona al proceso acceso de solo lectura, solo escritura o de lectura/escritura al búfer.

- *server*- especifica si este proceso ejecutará un servidor para este búfer.
- *timeout*: establece las características de tiempo de espera para los accesos al búfer.
- *master*: indica si este proceso es responsable de crear y destruir el búfer.
- *c_num*: un número entero entre cero y (max_procs -1)

2.22.4 Comentarios de configuración

Algunas de las combinaciones de configuración no son válidas, mientras que otras implican ciertas restricciones. En un sistema Linux, GLOBMEM es obsoleto, mientras que PHANTOM solo es realmente útil en la etapa de prueba de una aplicación. Igualmente para FILEMEM. LOCMEM es de poca utilidad para una aplicación multiproceso, y solo ofrece ventajas limitadas de rendimiento sobre SHMEM. Esto deja a SHMEM como el único tipo de búfer para usar con LinuxCNC.

La opción neut solo se usa en un sistema multiprocesador donde arquitecturas diferentes (e incompatibles) comparten un bloque de memoria. La probabilidad de ver un sistema de este tipo fuera de un museo o lugar de investigación es remoto y solo es relevante para buffers GLOBMEM.

El número RPC está documentado como obsoleto y solo se conserva por razones de compatibilidad.

Con un nombre de búfer único, tener una identidad numérica parece ser inútil. Es necesario revisar el código para identificar la lógica. Asimismo, el campo key parece ser redundante, y podría derivarse del nombre del búfer.

El propósito de limitar el número de procesos permitidos para conectarse a cualquier búfer no está claro a partir de la documentación existente y del código fuente original. Permitir un numero no especificado de procesos para conectarse a un búfer no es más difícil de implementar.

Los tipos mutex se reducen a uno de estos dos; el predeterminado "os_sem" o "mao split". La mayoría de los mensajes NML son relativamente cortos y se pueden copiar hacia o desde el búfer con retrasos mínimos, por lo que las lecturas divididas no son esenciales.

La codificación de datos solo es relevante cuando se transmite a un proceso remoto. Usar TCP o UDP implica codificación XDR. La codificación ASCII puede tener algún uso en diagnósticos o para pasar datos a un sistema integrado que no implementa NML.

Los protocolos UDP tienen menos chequeos en los datos y permiten descartar un porcentaje de paquetes. TCP es más confiable, pero es relativamente más lento.

Si LinuxCNC se va a conectar a una red, se esperaría que sea local y detrás de un firewall. La única razón para permitir el acceso a LinuxCNC a través de Internet sería para diagnósticos remotos. Esto puede ser logrado de manera mucho más segura utilizando otros medios, tal vez por una interfaz web.

El comportamiento exacto cuando timeout se establece en cero o un valor negativo no está claro de los documentos del NIST. Solo son mencionados valores INF y positivos. Sin embargo, dentro del código fuente de rcslib, es evidente que se aplica lo siguiente:

timeout > 0 - Bloqueo de acceso hasta que se alcanza el intervalo de tiempo de espera o el acceso al búfer esté disponible.

timeout = 0 - El acceso al búfer solo es posible si no hay otro proceso que esté leyendo o escribiendo en ese momento.

timeout < 0 o INF - El acceso está bloqueado hasta que el búfer esté disponible.

2.23 clase base NML

Expandir las listas y la relación entre NML, NMLmsg y el clases de cms de nivel inferior.

No debe confundirse con NMLmsg, RCS_STAT_MSG o RCS_CMD_MSG.

NML es responsable de analizar el archivo de configuración, configurar el búfer cms y es el mecanismo para enrutar mensajes a bufer(s) correcto(s). Para hacer esto, NML crea varias listas para:

- búferes cms creados o conectados.

- procesos y búferes a las que se conectan
- una larga lista de funciones de formato para cada tipo de mensaje

Este último elemento es probablemente el núcleo de gran parte de la desalineación de libnml/rcslib y NML en general. Cada mensaje que se pasa a través de NML requiere que se adjunte una cierta cantidad de información además de los datos reales. Para hacer esto, se invocan en secuencia varias funciones de formato para ensamblar fragmentos del mensaje general. las funciones de formato incluirán NML_TYPE, MSG_TYPE, además de los datos declarado en clases NMLmsg derivadas. Los cambios en el orden en que se llaman las funciones de formato y también las variables pasadas pueden romper la compatibilidad con rcslib si se hacen mal. Hay razones para mantener la compatibilidad con rcslib y buenas razones para alterar el código. La pregunta es, ¿qué conjunto de razones son primordiales?

2.23.1 Interioridades de NML

2.23.1.1 constructor NML

NML::NML() analiza el archivo de configuración y lo almacena en una lista enlazada para ser pasada a constructores cms en líneas simples. Es la función constructor NML para llamar al constructor cms relevante para cada búfer y mantener una lista de los objetos cms y los procesos asociados con cada búfer.

NML puede interactuar con cms desde los punteros almacenados en las listas y el por qué de que Doxygen no muestra las relaciones reales involucradas.

Note
La configuración se almacena en la memoria antes de pasar un puntero a una línea específica para el constructor cms. El constructor cms analiza luego la línea nuevamente para extraer un par de variables ... Tendría más sentido hacer TODO el análisis y guardar las variables en una estructura que sea pasada al constructor cms. Esto eliminaría el manejo de cadenas y reduciría el código duplicado en cms ...

2.23.1.2 Lectura/escritura NML

Las llamadas a NML::read y NML::write realizan tareas similares en el modo de procesar el mensaje; la única variación real está en el dirección del flujo de datos.

Una llamada a la función de lectura primero obtiene datos del búfer y luego llama a format_output(), mientras que una función de escritura llamaría a format_input() antes de pasar los datos al búfer. El trabajo de construir o deconstruir el mensaje está dentro de format_xxx(). Una lista de funciones variadas se llama a su vez para colocar varias partes del encabezado NML (que no debe confundirse con el encabezado cms) en el orden correcto. La última función llamada es emcFormat() en emc.cc.

2.23.1.3 Relaciones NMLmsg y NML

NMLmsg es la clase base de la que se derivan todas las clases de mensajes. Cada clase de mensaje debe tener un ID único definido (y pasado al constructor) y también una función update(*cms). update() será llamado por las funciones de lectura/escritura NML cuando se llama al formateador NML - El puntero al formateador habrá sido declarado en el constructor NML en algún momento. En virtud de las listas enlazadas que crea NML, puede seleccionar el puntero cms que se pasa al formateador y, por tanto, que búfer se utilizará.

2.24 Agregar comandos NML personalizados

LinuxCNC es bastante impresionante, pero algunas partes necesitan algunos ajustes. Como ya sabe, la comunicación se realiza a través de canales NML. Los datos enviados a través de tales canales es una de las clases definidas en emc.hh (implementado en emc.cc). Si alguien necesita un tipo de mensaje que no existe, debería seguir estos pasos para agregar uno nuevo. (El mensaje que se agrega en el ejemplo se llama EMC_IO_GENERIC (hereda EMC_IO_CMD_MSG (hereda RCS_CMD_MSG)))

1. agregar la definición de la clase EMC_IO_GENERIC a /src/emc/nml_intf/emc.hh
2. agregar el tipo: #define EMC_IO_GENERIC_TYPE ((NMLTYPE) 1605)

    a. (Se elige 1605 porque esta disponible) en /src/emc/nml_intf/emc.hh
3. agregar el caso EMC_IO_GENERIC_TYPE a emcFormat en /src/emc/nml_intf/emc.cc
4. agregar el caso EMC_IO_GENERIC_TYPE a emc_symbol_lookup en /src/emc/nml_intf/emc.cc
5. agregar la función EMC_IO_GENERIC::update a /src/emc/nml_intf/emc.cc

Al recompilar, el nuevo mensaje debería estar allí. La siguiente parte es enviar tales mensajes desde algún lugar y recibirlos en otro lugar, y hacer algunas cosas con eso.

2.25 La tabla de herramientas y el cambiador de herramientas

LinuxCNC interactúa con el hardware del cambiador de herramientas y tiene una abstracción interna del mismo. LinuxCNC gestiona la información de la herramienta con un archivo de tabla de herramientas

2.25.1 Abstracción del cambiador de herramientas en LinuxCNC

LinuxCNC admite dos tipos de hardware de cambiador de herramientas, llamados *nonrandom* y *random*. La entrada ini <<-section,[EMCIO]RANDOM_TOOLCHANGER controla cuál de estos tipos de hardware es con el que LinuxCNC considera que está conectado.

2.25.1.1 Cambiadores de herramientas no aleatorios

El hardware de cambiador de herramientas no aleatorio vuelve a colocar cada herramienta en la ranura desde la que fue originalmente cargada.

Ejemplos de hardware de cambiador de herramientas no aleatorio son el cambiador de herramientas "manual", torretas de herramientas de torno y cambiadores de herramientas en rack.

Cuando se configura para un cambiador de herramientas no aleatorio, LinuxCNC no cambia el número de ranura en el archivo de la tabla de herramientas a medida que las herramientas se cargan y descargan. Internamente, en el cambio de herramienta la información de la herramienta se **copia** de la ranura fuente de la tabla de herramientas a la ranura 0 (que representa el husillo), reemplazando cualquier información de herramienta que estaba allí anteriormente.

NOTA: Con LinuxCNC configurado para cambiador de herramientas no aleatorio, la herramienta 0 (T0) tiene significado especial: "sin herramienta". T0 puede no aparecer en el archivo de tabla de herramientas y cambiar a T0 dará como resultado que LinuxCNC piense que tiene el husillo vacío.

2.25.1.2 Cambiadores de herramientas aleatorios

El hardware de cambiador de herramientas aleatorio intercambia la herramienta en el husillo (si existe) con la herramienta solicitada a cambiar. Así, la ranura donde reside una herramienta cambia a medida que se intercambia dentro y fuera del husillo.

Un ejemplo de hardware de cambiador de herramientas aleatorio es un cambiador de herramientas de carrusel.

Cuando se configura para un cambiador de herramientas aleatorio, LinuxCNC intercambia el número de ranura de la herramienta antigua y la nueva en el archivo de tabla de herramientas cuando se cargan las herramientas. Internamente, en el cambio de herramienta la información de la herramienta se **intercambia** entre la ranura de origen de la tabla de herramientas y la ranura 0 (que representa el husillo). Por tanto, después de un cambio de herramienta, la ranura 0 en la tabla de herramientas tendrá la información de la herramienta para la nueva herramienta y la ranura de la que la nueva herramienta vino tendrá la información de la herramienta que estaba en el husillo antes del cambio de herramienta, si la había.

NOTA: En LinuxCNC configurado para cambiador de herramientas aleatorio, la herramienta 0 (T0) **no** tiene significado especial. Se trata exactamente como cualquier otra herramienta en la tabla de herramientas. Es habitual utilizar T0 para representar "sin herramienta" (es decir, una herramienta con TLO cero), de modo que el husillo se pueda vaciar convenientemente cuando sea necesario.

2.25.2 La tabla de herramientas

LinuxCNC realiza un seguimiento de las herramientas en un archivo llamado << sec:tool-table,tabla de herramientas>>. La tabla de herramientas registra la siguiente información para cada herramienta:

número de herramienta
Un entero que identifica de forma exclusiva esta herramienta. Los números de herramienta son manejados de manera diferente por LinuxCNC cuando se configuran cambiadores de herramientas no aleatorios o aleatorios:

- Cuando LinuxCNC está configurado para un cambiador de herramientas no aleatorio, el número debe ser positivo. T0 recibe un manejo especial y no está permitido que aparezca en la tabla de herramientas.
- Cuando LinuxCNC está configurado para un cambiador de herramientas aleatorio este número debe ser positivo o cero. T0 está permitido en la tabla de herramientas y generalmente se usa para representar "ninguna herramienta", es decir, ranura vacía.

número de ranura
Un entero que identifica la ranura en el hardware del cambiador donde reside la herramienta. Los números de ranura se manejan de manera diferente por LinuxCNC cuando está configurado para cambiadores de herramientas aleatorios y no aleatorio:

- Cuando LinuxCNC está configurado para un cambiador de herramientas no aleatorio, el número de ranura en el archivo de herramientas puede ser cualquier número entero positivo (ranura 0 no está permitida). LinuxCNC compacta en silencio los números de ranura cuando carga el archivo de herramienta, por lo que puede haber una diferencia entre los números en el archivo de herramientas y los números internos de ranura utilizados por LinuxCNC.
- Cuando LinuxCNC está configurado para un cambiador de herramientas aleatorio, los números de ranura en el archivo de herramientas deben estar entre 0 y 1000, ambos inclusive. Las ranuras 1-1000 están en el cambiador de herramientas; la ranura 0 es el husillo.

diámetro
Diámetro de la herramienta, en unidades de máquina.

offsets de longitud de herramienta
Desplazamiento de longitud de herramienta (también llamado TLO), en hasta 9 ejes, en unidades máquina. Los ejes que no tienen una TLO especificada, la rellenan con 0.

2.25.3 Códigos G que afectan a las herramientas

Los gcodes que usan o afectan la información de la herramienta son:

2.25.3.1 Txxx

Le dice al hardware del cambiador de herramientas que se prepare para cambiar a una determinada herramienta `xxx`.

Manejado por `Interp::convert_tool_select()`.

1. Se le pide a la máquina que se prepare para cambiar a la herramienta seleccionada llamando a la función Canonica `SELECT_POCKET()` con el número de ranura de la herramienta solicitada.

    a. (saicanon) No-op.

b. (emccanon) Crea un mensaje `EMC_TOOL_PREPARE` con el número de ranura solicitada y lo envía a Task, que lo envía a IO. IO recibe el mensaje y le pide a HAL que prepare la ranura configurando `iocontrol.0.tool-prep-pocket`, `iocontrol.0.tool-prep-number`, y `iocontrol.0.tool-prepare`. IO luego llama repetidamente a `read_tool_inputs()` para sondear el pin HAL `iocontrol.0.tool-ready`, que informa a IO, desde el hardware del cambiador de herramientas, a través de HAL, que la preparación de la herramienta solicitada está completa. Cuando ese pin se vuelve true, IO establece `emcioStatus.tool.pocketPrepped` con el número de ranura de la herramienta solicitada.

2. De vuelta a Interp, se le asigna a `settings->selected_pocket` el número de ranura de la herramienta solicitada *xxx*.

2.25.3.2 M6

Le dice al cambiador de herramientas que cambie a la herramienta seleccionada actualmente (seleccionada por el comando Txxx anterior).

Manejado por `Interp::convert_tool_change()`.

1. Se le pide a la máquina que cambie a la herramienta seleccionada llamando a la función Canónica `CHANGE_TOOL()` con `settings->selected_pocket`.

   a. (saicanon) Establece sai `_active_slot` en el número de ranura pasado. La información de la herramienta se copia de la ranura seleccionada de la tabla de herramientas (es decir, de sai's `_tools[_active_slot]`) al husillo (a sai `_tools[0]`).

   b. (emccanon) Envía un mensaje `EMC_TOOL_LOAD` a Task, que lo envía a IO. IO establece `emcioStatus.tool.toolInSpindle` al número de herramienta de la herramienta en el ranura identificado por `emcioStatus.tool.pocketPrepped` (establecido por `Txxx` alias `SELECT_POCKET()`). Luego solicita que el cambiador de herramientas hardware realice un cambio de herramienta, configurando el pin HAL `iocontrol.0.tool-change` a True. Más tarde, IO's `read_tool_inputs() + detectará que el pin HAL +iocontrol.0.tool_changed` se ha establecido en True, lo que indica que toolchanger ha completado el cambio de herramienta. Cuando esto pasa, llama a `load_tool()` para actualizar el estado de la máquina.

      i. `load_tool()` con un cambiador de herramientas no aleatorio, copia la información de la herramienta de la ranura seleccionada al husillo (ranura 0).

      ii. `load_tool()` con cambiador de herramientas aleatorio, intercambia información entre el ranura 0 (el husillo) y la ranura seleccionada, luego guarda la tabla de herramientas.

2. De vuelta en interp, `settings->current_pocket` se le asigna la nueva herramienta desde `settings->selected_pocket` (establecido por `Txxx`). Los parámetros numerados relevantes (<<sub:numbered-parameters, #5400- #5413) son actualizados con la nueva información de herramienta de la ranura 0 (husillo).

2.25.3.3 G43/G43.1/G49

Aplicar desplazamiento de longitud de herramienta. G43 usa el TLO de la herramienta cargada actualmente, o de una herramienta especificada si la palabra H se da en el bloque. G43.1 consigue el TLO de las palabras de eje en el bloque. G49 cancela el TLO (usa 0 para el desplazamiento de todos los ejes).

Manejado por `Interp::convert_tool_length_offset()`.

1. Comienza construyendo una `EmcPose` que contiene los desplazamientos a usar de 9 ejes. Para `G43.1`, estas compensaciones de herramientas provienen de palabras de eje en el bloque actual. Para `G43` estos desplazamientos provienen de la herramienta actual (la herramienta en la ranura 0), o de la herramienta especificada por la palabra H en el bloque. Para G49, los desplazamientos son todos 0.

2. Los desplazamientos se pasan a la función `USE_TOOL_LENGTH_OFFSET()` Canonica.

   a. (saicanon) Graba el TLO en `_tool_offset`.

b. (emccanon) Crea un mensaje `EMC_TRAJ_SET_OFFSET` que contiene los offsets y lo envía a Task, que copia las compensaciones en `emcStatus->task.toolOffset` y los envía a Motion a través de un comando `EMC MOT_SET_OFFSET`. Motion copia las compensaciones a `emcmotStatus->tool_offset`, donde se usa para compensar movimientos futuros

3. De vuelta en interp, los desplazamientos se registran en `settings->tool_offset`. La ranura efectiva se registra en `settings->tool_offset_index`, aunque este valor nunca se usa.

2.25.3.4 G10 L1/L10/L11

Modifica la tabla de herramientas.

Manejado por `Interp::convert_setup_tool()`.

1. Selecciona el número de herramienta de la palabra P en el bloque y encuentra la ranura para esa herramienta:

   a. Con una configuración de cambiador de herramientas no aleatorio, este es siempre el número de ranura en el cambiador de herramientas (incluso cuando la herramienta está en el husillo).

   b. Con una configuración de cambiador de herramientas aleatorio, si la herramienta está actualmente cargada utiliza la ranura 0 (ranura 0 significa "el husillo"), y si la herramienta no está cargada, usa el número de ranura en el cambiador de herramientas. (Esta diferencia es importante).

2. Averigua cuáles deberían ser las nuevas compensaciones.

3. La nueva información de la herramienta (diámetro, desplazamientos, ángulos y orientación), junto con el número de herramienta y el número de ranura, se pasan al Canon llame a SET_TOOL_TABLE_ENTRY ().

   a. (saicanon) Copie la información de la nueva herramienta en el ranura especificado (en la tabla de herramientas interna de sai, + _tools +).

   b. (emccanon) Cree un mensaje + EMC_TOOL_SET_OFFSET + con el nuevo información de la herramienta y enviarla a Tarea, que la pasa a IO. IO actualiza el ranura especificado en su interno copia de la tabla de herramientas (+ emcioStatus.tool.toolTable `), y si la herramienta especificada está cargada actualme nte (se compara con +emcioStatus.tool.toolInSpindle`) luego la información de la nueva herramienta se copia en el ranura 0 (el eje) también. (FIXME: eso es un buglet, solo debe copiarse en máquinas no aleatorias). Finalmente IO guarda la nueva tabla de herramientas.

4. De vuelta en interp, si la herramienta modificada está cargada actualmente en el husillo, y si la máquina es un cambiador de herramientas no aleatorio, entonces la nueva información de la herramienta se copia del ranura de inicio de la herramienta al ranura 0 (el huso) en la copia de interp de la tabla de herramientas, `configuración-> tool_table`. (Esta copia no es necesaria en una herramienta aleatoria máquinas de cambio porque allí, las herramientas no tienen un ranura en casa y en su lugar, acabamos de actualizar la herramienta en el ranura 0 directamente).

5. Los parámetros numerados relevantes (<< sub: parámetros numerados, # 5400- # 5413 >>) se actualizan desde la herramienta información en el huso (copiando la información de interp's `configuración-> tool_table` a `configu ración-> parámetros`). (FIXME: esto es un buglet, los params solo deberían actualizarse si era el actual herramienta que fue modificada).

6. Si la herramienta modificada está cargada actualmente en el eje, y si la configuración es para un cambiador de herramientas no aleatorio, entonces el la nueva información de herramienta también se escribe en el ranura 0 de la tabla de herramientas, a través de una segunda llamada a SET_TOOL_TABLE_ENTRY (). (Esta segunda tabla de herramientas la actualización no es necesaria en máquinas de cambio de herramientas aleatorias porque allí, las herramientas no tienen un ranura de casa y en su lugar acabamos de actualizar la herramienta en el ranura 0 directamente.)

2.25.3.5 M61

Establecer el número de herramienta actual. Esto cambia la representación interna de LinuxCNC de qué herramienta está en el eje, sin mover realmente el cambiador de herramientas o intercambiando cualquier herramienta.

Manejado por + Interp
convert_tool_change () +.

Canon: + CHANGE_TOOL_NUMBER ()

settings-> current_pocket tiene asignado el número de ranura actualmente sosteniendo la herramienta especificada por el argumento Q-word.

2.25.3.6 G41 / G41.1 / G42 / G42.1

Habilite la compensación del radio de corte (generalmente se llama *cutter comp*).

Manejado por + Interp
convert_cutter_compensation_on () +.

No hay llamada de Canon, la composición del cortador ocurre en el intérprete. Usa la herramienta tabla de la manera esperada: si se proporciona un número de herramienta D-word se ve arriba el número de ranura del número de herramienta especificado en la tabla, y si no se suministra ninguna palabra D, utiliza la cavidad 0 (el eje).

2.25.3.7 G40

Cancele la compensación del radio de corte.

Manejado por + Interp
convert_cutter_compensation_off () +.

No hay llamada de Canon, la composición del cortador ocurre en el intérprete. No se usa la mesa de herramientas.

2.25.4 Variables de estado interno

¡Esta no es una lista exhaustiva! La información de la herramienta se difunde a través de fuera LinuxCNC.

2.25.4.1 IO

`emcioStatus` es de tipo `EMC_IO_STAT`

emcioStatus.tool.pocketPrepped
Cuando IO recibe la señal de HAL de que la preparación del cambiador de herramientas es completa (después de un comando + Txxx +), esta variable se establece en ranura de la herramienta solicitada. Cuando IO recibe la señal de HAL que el cambio de herramienta en sí está completo (después de un comando + M6 +), esta variable se restablece a -1.

emcioStatus.tool.toolInSpindle
Número de herramienta de la herramienta instalada actualmente en el eje. Exportado en el pin HAL + iocontrol.0.tool-number + (s32).

emcioStatus.tool.toolTable []
Una matriz de + CANON_TOOL_TABLE + estructuras, + CANON_POCKETS_MAX + long. Cargado desde el archivo de la tabla de herramientas al inicio y mantenido allí después El índice 0 es el huso, los índices 1- (CANON_POCKETS_MAX-1) son los ranuras en el cambiador de herramientas. Esta es una copia completa de la información de la herramienta, mantenida por separado de la de Interp `settings.tool_table`.

2.25.4.2 interp

`settings` es de tipo `settings`, que es `struct setup_struct`. Definido en `src /emc /rs274ngc /interp_internal.hh`.

settings.selected_pocket
ranura de la herramienta seleccionada más recientemente por + Txxx +.

settings.current_pocket
ranura original de la herramienta actualmente en el husillo. En otras palabras: qué cambiador de herramientas guarda la herramienta que está actualmente en el eje fue cargado desde.

settings.tool_table []
Un conjunto de información de herramientas. El índice en la matriz es el "ranura número "(también conocido como" número de ranura "). El ranura 0 es el eje, los ranuras 1 hasta (CANON_POCKETS_MAX-1) son los ranuras del cambiador de herramientas.

settings.tool_offset_index
No usado. FIXME: Probablemente debería eliminarse.

settings.toolchange_flag
Interp establece esto en verdadero cuando llama a Canon CHANGE_TOOL ()

función. Está marcado en + Interp
convert_tool_length_offset ()
para decidir qué ranura usar para G43 (sin palabra H): `configuración-> current_pocket` si el cambio de herramienta aún está en progreso, ranura 0 (el husillo) si se completa el cambio de herramienta.

settings.random_toolchanger
Establecer desde la variable ini + [EMCIO] RANDOM_TOOLCHANGER + al inicio. Controla varias herramientas de lógica de manejo de tablas. (IO también lee esto ini variable y cambia su comportamiento en función de ella. Por ejemplo, al guardar la tabla de herramientas, el cambiador de herramientas aleatorio guarda la herramienta en el husillo (ranura 0), pero el cambiador de herramientas no aleatorio guarda cada herramienta en su "ranura de casa".)

settings.tool_offset
Esta es una variable + EmcPose +.

- Se utiliza para calcular la posición en varios lugares.
- Enviado a Motion a través del mensaje `EMCMOT_SET_OFFSET`. Todo el movimiento que se hace con los desplazamientos es exportarlos a los pines HAL `motion.0.tooloffset.[xyzabcuvw]`. FIXME: exportarlos desde algún lugar más cercano a la mesa de herramientas (io o interp, probablemente) y elimine el mensaje EMCMOT_SET_OFFSET.

settings.pockets_max
Se usa de manera intercambiable con + CANON_POCKETS_MAX + (una constante #definida, establecido en 1000 a partir de abril de 2020). FIXME: esta variable de configuración actualmente no es útil y probablemente debería eliminarse.

settings.tool_table
Esta es una matriz de `CANON_TOOL_TABLE` estructuras (definidas en `src /emc /nml_intf /emctool.h`), con `CANON_POCKETS_MAX` entradas. Indizado por "número de ranura", también conocido como "número de ranura". El índice 0 es el husillo, los índices 1- (CANON_POCKETS_MAX-1) son los ranuras de la herramienta cambiador En un cambiador de herramientas al azar, los números de ranura son significativos. En un cambiador de herramientas no aleatorio, los ranuras no tienen sentido; el ranura los números en el archivo de la tabla de herramientas se ignoran y las herramientas se asignan para `tool_table` ranuras secuencialmente.

settings.tool_change_at_g30 , settings.tool_change_quill_up , settings.tool_change_with_spindle_on
Estos se establecen a partir de variables ini en la sección `[EMCIO]`, y controlar cómo se realizan los cambios de herramienta.

2.26 Recuento de juntas y ejes

2.26.1 En el búfer de estado

El buffer de estado es utilizado por Task y las UI.

FIXME: `axis_mask` y` axes` especifican en exceso el número de ejes

`status.motion.traj.axis_mask`
Una máscara de bits con un "1" para los ejes que están presentes y un "0" para los ejes que no están presentes. X es el bit 0, Y es el bit 1, etc. Por ejemplo, una máquina con ejes X y Z tendría una `axis_mask` de 0x5, una máquina XYZ tendría 0x7, y una máquina XYZB tener un `axis_mask` de 0x17.

`status.motion.traj.axes` (en desuso)
El valor de esta variable es uno más que el índice de eje con el número más alto presente en la máquina. Como en la `axis_mask`, el índice de X en 0, Y es 1, etc. Una máquina XZ tiene un valor de *ejes* de 3, al igual que una máquina XYZ. Una máquina XYZW tiene un valor de *ejes* 9. Esta variable no es muy útil y su uso está en desuso. Use `axis_mask` en su lugar.

`status.motion.traj.joints`
Un recuento del número de juntas que tiene la máquina. Un torno normal tiene 2 articulaciones; uno manejando el eje X y otro manejando el eje Z. Un molino de pórtico XYYZ tiene 4 articulaciones; uno manejando X, uno manejando un lado de la Y, uno manejando el otro lado de la Y, y uno manejando Z. Un molino XYZA también tiene 4 articulaciones.

`status.motion.axis [EMCMOT_MAX_AXIS]`
Una matriz de estructuras de eje `EMCMOT_MAX_AXIS`. `axis [n]` es válido si `(axis_mask & (1 << n))` es True. Si `(axis_mask & (1 << n))` es Falso, entonces `axis [n]` no existe en esta máquina y debe ser ignorado.

`status.motion.joint [EMCMOT_MAX_JOINTS]`
Una matriz de estructuras de unión `EMCMOT_MAX_JOINTS`. `conjunta [0]` a través `joint [joint-1]` son válidas, las otras no existen en esta máquina y debe ser ignorado.

Las cosas no están así actualmente en la rama de ejes articulados, pero Las desviaciones de este diseño se consideran errores. Por un ejemplo de tal un error, vea el tratamiento de los ejes en src/emc/ini/initraj.cc: loadTraj (). Indudablemente hay más, y necesito tu ayuda para encontrarlos y arreglalos.

2.26.2 En motion

El componente en tiempo real del controlador de movimiento obtiene primero el número de uniones del parámetro de tiempo de carga `num_joints`. Esto determina cuantos Se crean juntas por valor de pines HAL al inicio.

El número de articulaciones de Motion se puede cambiar en tiempo de ejecución utilizando el Comando `EMCMOT_SET_NUM_JOINTS` de la tarea.

El controlador de movimiento siempre funciona en los ejes `EMCMOT_MAX_AXIS`. Siempre crea nueve conjuntos de pines `axis.*.*`.

Chapter 3

Mensajes NML

Lista de mensajes NML Para mas detalles, vea src/emc/nml_intf/emc.hh

OPERATOR

```
    EMC_OPERATOR_ERROR_TYPE
    EMC_OPERATOR_TEXT_TYPE
    EMC_OPERATOR_DISPLAY_TYPE
```

JOINT

```
    EMC_JOINT_SET_JOINT_TYPE
    EMC_JOINT_SET_UNITS_TYPE
    EMC_JOINT_SET_MIN_POSITION_LIMIT_TYPE
    EMC_JOINT_SET_MAX_POSITION_LIMIT_TYPE
    EMC_JOINT_SET_FERROR_TYPE
    EMC_JOINT_SET_HOMING_PARAMS_TYPE
    EMC_JOINT_SET_MIN_FERROR_TYPE
    EMC_JOINT_SET_MAX_VELOCITY_TYPE
    EMC_JOINT_INIT_TYPE
    EMC_JOINT_HALT_TYPE
    EMC_JOINT_ABORT_TYPE
    EMC_JOINT_ENABLE_TYPE
    EMC_JOINT_DISABLE_TYPE
    EMC_JOINT_HOME_TYPE
    EMC_JOINT_ACTIVATE_TYPE
    EMC_JOINT_DEACTIVATE_TYPE
    EMC_JOINT_OVERRIDE_LIMITS_TYPE
    EMC_JOINT_LOAD_COMP_TYPE
    EMC_JOINT_SET_BACKLASH_TYPE
    EMC_JOINT_UNHOME_TYPE
    EMC_JOINT_STAT_TYPE
```

AXIS

```
    EMC_AXIS_STAT_TYPE
```

JOG

```
    EMC_JOG_CONT_TYPE
    EMC_JOG_INCR_TYPE
    EMC_JOG_ABS_TYPE
    EMC_JOG_STOP_TYPE
```

TRAJ

```
    EMC_TRAJ_SET_AXES_TYPE
    EMC_TRAJ_SET_UNITS_TYPE
    EMC_TRAJ_SET_CYCLE_TIME_TYPE
    EMC_TRAJ_SET_MODE_TYPE
    EMC_TRAJ_SET_VELOCITY_TYPE
    EMC_TRAJ_SET_ACCELERATION_TYPE
    EMC_TRAJ_SET_MAX_VELOCITY_TYPE
    EMC_TRAJ_SET_MAX_ACCELERATION_TYPE
    EMC_TRAJ_SET_SCALE_TYPE
    EMC_TRAJ_SET_RAPID_SCALE_TYPE
    EMC_TRAJ_SET_MOTION_ID_TYPE
    EMC_TRAJ_INIT_TYPE
    EMC_TRAJ_HALT_TYPE
    EMC_TRAJ_ENABLE_TYPE
    EMC_TRAJ_DISABLE_TYPE
    EMC_TRAJ_ABORT_TYPE
    EMC_TRAJ_PAUSE_TYPE
    EMC_TRAJ_STEP_TYPE
    EMC_TRAJ_RESUME_TYPE
    EMC_TRAJ_DELAY_TYPE
    EMC_TRAJ_LINEAR_MOVE_TYPE
    EMC_TRAJ_CIRCULAR_MOVE_TYPE
    EMC_TRAJ_SET_TERM_COND_TYPE
    EMC_TRAJ_SET_OFFSET_TYPE
    EMC_TRAJ_SET_G5X_TYPE
    EMC_TRAJ_SET_HOME_TYPE
    EMC_TRAJ_SET_ROTATION_TYPE
    EMC_TRAJ_SET_G92_TYPE
    EMC_TRAJ_CLEAR_PROBE_TRIPPED_FLAG_TYPE
    EMC_TRAJ_PROBE_TYPE
    EMC_TRAJ_SET_TELEOP_ENABLE_TYPE
    EMC_TRAJ_SET_SPINDLESYNC_TYPE
    EMC_TRAJ_SET_SPINDLE_SCALE_TYPE
    EMC_TRAJ_SET_FO_ENABLE_TYPE
    EMC_TRAJ_SET_SO_ENABLE_TYPE
    EMC_TRAJ_SET_FH_ENABLE_TYPE
    EMC_TRAJ_RIGID_TAP_TYPE
    EMC_TRAJ_STAT_TYPE
```

MOTION

```
    EMC_MOTION_INIT_TYPE
    EMC_MOTION_HALT_TYPE
    EMC_MOTION_ABORT_TYPE
    EMC_MOTION_SET_AOUT_TYPE
    EMC_MOTION_SET_DOUT_TYPE
    EMC_MOTION_ADAPTIVE_TYPE
    EMC_MOTION_STAT_TYPE
```

TASK

```
    EMC_TASK_INIT_TYPE
    EMC_TASK_HALT_TYPE
    EMC_TASK_ABORT_TYPE
    EMC_TASK_SET_MODE_TYPE
    EMC_TASK_SET_STATE_TYPE
    EMC_TASK_PLAN_OPEN_TYPE
```

```
    EMC_TASK_PLAN_RUN_TYPE
    EMC_TASK_PLAN_READ_TYPE
    EMC_TASK_PLAN_EXECUTE_TYPE
    EMC_TASK_PLAN_PAUSE_TYPE
    EMC_TASK_PLAN_STEP_TYPE
    EMC_TASK_PLAN_RESUME_TYPE
    EMC_TASK_PLAN_END_TYPE
    EMC_TASK_PLAN_CLOSE_TYPE
    EMC_TASK_PLAN_INIT_TYPE
    EMC_TASK_PLAN_SYNCH_TYPE
    EMC_TASK_PLAN_SET_OPTIONAL_STOP_TYPE
    EMC_TASK_PLAN_SET_BLOCK_DELETE_TYPE
    EMC_TASK_PLAN_OPTIONAL_STOP_TYPE
    EMC_TASK_STAT_TYPE
```

TOOL

```
    EMC_TOOL_INIT_TYPE
    EMC_TOOL_HALT_TYPE
    EMC_TOOL_ABORT_TYPE
    EMC_TOOL_PREPARE_TYPE
    EMC_TOOL_LOAD_TYPE
    EMC_TOOL_UNLOAD_TYPE
    EMC_TOOL_LOAD_TOOL_TABLE_TYPE
    EMC_TOOL_SET_OFFSET_TYPE
    EMC_TOOL_SET_NUMBER_TYPE
    EMC_TOOL_START_CHANGE_TYPE
    EMC_TOOL_STAT_TYPE
```

AUX

```
    EMC_AUX_ESTOP_ON_TYPE
    EMC_AUX_ESTOP_OFF_TYPE
    EMC_AUX_ESTOP_RESET_TYPE
    EMC_AUX_INPUT_WAIT_TYPE
    EMC_AUX_STAT_TYPE
```

SPINDLE

```
    EMC_SPINDLE_ON_TYPE
    EMC_SPINDLE_OFF_TYPE
    EMC_SPINDLE_INCREASE_TYPE
    EMC_SPINDLE_DECREASE_TYPE
    EMC_SPINDLE_CONSTANT_TYPE
    EMC_SPINDLE_BRAKE_RELEASE_TYPE
    EMC_SPINDLE_BRAKE_ENGAGE_TYPE
    EMC_SPINDLE_SPEED_TYPE
    EMC_SPINDLE_ORIENT_TYPE
    EMC_SPINDLE_WAIT_ORIENT_COMPLETE_TYPE
    EMC_SPINDLE_STAT_TYPE
```

COOLANT

```
    EMC_COOLANT_MIST_ON_TYPE
    EMC_COOLANT_MIST_OFF_TYPE
    EMC_COOLANT_FLOOD_ON_TYPE
    EMC_COOLANT_FLOOD_OFF_TYPE
    EMC_COOLANT_STAT_TYPE
```

LUBE

```
    EMC_LUBE_ON_TYPE
    EMC_LUBE_OFF_TYPE
    EMC_LUBE_STAT_TYPE
```

IO (INPUT/OUTPUT)

```
    EMC_IO_INIT_TYPE
    EMC_IO_HALT_TYPE
    EMC_IO_ABORT_TYPE
    EMC_IO_SET_CYCLE_TIME_TYPE
    EMC_IO_STAT_TYPE
    EMC_IO_PLUGIN_CALL_TYPE
```

OTROS

```
    EMC_NULL_TYPE
    EMC_SET_DEBUG_TYPE
    EMC_SYSTEM_CMD_TYPE
    EMC_INIT_TYPE
    EMC_HALT_TYPE
    EMC_ABORT_TYPE
    EMC_STAT_TYPE
    EMC_EXEC_PLUGIN_CALL_TYPE
```

Chapter 4

Estilo de codificación

Este capítulo describe el estilo de código fuente preferido por el equipo LinuxCNC.

4.1 No provoque daños

Al realizar pequeñas ediciones en el código en un estilo diferente al descrito a continuación, observe el estilo de codificación local. Cambios rápidos de un estilo de codificación a otro disminuye la legibilidad del código.

Nunca verifique el código después de ejecutar "sangría" en él. Los cambios de espacios en blanco introducidos por la sangría hacen que sea más difícil seguir el historial de revisión del archivo.

No use un editor que realice cambios innecesarios en los espacios en blanco (por ejemplo, reemplaza 8 espacios con un tabulador en una línea no modificada, o ajuste de texto en líneas no modificadas de otra manera)

4.2 Tabulaciones

Una tabulación siempre corresponde a 8 espacios. No escriba código que se muestra correctamente solo con una configuración de tabulación diferente.

4.3 Sangría

Use 4 espacios por nivel de sangría. Combinando 8 espacios en un tabulador es aceptable pero no requerido.

4.4 Colocación de llaves

Ponga la llave de apertura al final de la línea y la llave de cierre al principio:

```
if (x) {
    // hacer algo apropiado
}
```

La llave de cierre está en una línea propia, excepto en los casos en que sea seguida por una continuación de la misma declaración, es decir, un *while* en una declaración do o un *else* en una declaración if, así:

```
do {
    // algo importante
} while (x > 0);
```

y

```
if (x == y) {
    // Haz una cosa
} else if (x < y) {
    // haz otra cosa
} else {
    // hacer una tercera cosa
}
```

Esta colocación de llaves también minimiza el número de líneas vacías (o casi vacias), lo que permite una mayor cantidad de código o comentarios visible a la vez en un terminal de un tamaño fijo.

4.5 Nombrado

C es un lenguaje espartano, y así debería ser tu nombrado. A diferencia de Modula-2 y Pascal, los programadores C no usan nombres pomposos como ThisVariableIsATemporaryCounter. Un programador de C llamaría eso variable *tmp*, que es mucho más fácil de escribir y no mas difícil de comprender.

Sin embargo, los nombres descriptivos para las variables globales son imprescindibles. Llamar a una función global *foo* es un crimen.

Las variables GLOBALES (para usarlas solo si **realmente** se precisan) necesitan tener nombres descriptivos, al igual que las funciones globales. Si tiene una función que cuenta el número de usuarios activos, debe llame a eso *count_active_users()* o similar, **no** debe llamarla *cntusr()*.

Codificar el tipo de una función en el nombre (llamada notación húngara) no tiene sentido; el compilador conoce los tipos de todos modos y puede verificarlos, y solo confunde al programador. No es de extrañar que Microsoft haga programas con errores.

Los nombres de variables LOCALES deben ser cortos y escuetos. Si usted tiene algún contador entero de bucles aleatorio, probablemente debería llamarse *i*. Llamarlo *loop_counter* no es productivo, si no hay posibilidad de ser mal entendido. Del mismo modo, *tmp* puede ser casi cualquier tipo de variable que se utiliza para mantener un valor temporal.

Si tiene miedo de mezclar sus nombres de variables locales, tiene otro problema, que se llama síndrome de desequilibrio de la hormona del crecimiento funcional. Ver el siguiente capítulo.

4.6 Funciones

Las funciones deben ser cortas y fáciles, y hacer una sola cosa. Deben caber en una o dos pantallas de texto (el tamaño de pantalla ISO/ANSI es 80x24, como todos sabemos), hacer una cosa y hacerla bien.

La longitud máxima de una función es inversamente proporcional a la complejidad y nivel de sangría de esa función. Por tanto, si tiene una función conceptualmente simple que es solo un largo (pero simple) switch-case, donde tienes que hacer muchas cosas pequeñas para muchos casos diferentes, está bien tener una función más larga.

Sin embargo, si tiene una función compleja y sospecha que un estudiante de primer año de secundaria no superdotado no podría siquiera entender de qué se trata la función, debe respetar los límites máximos más de cerca. Use funciones de ayuda con nombres descriptivos (puede pedirle al compilador que los incorpore si piensa que es crítico para el rendimiento, y probablemente hará un mejor trabajo que el suyo).

Otra medida de la función es el número de variables locales. No deben exceder de 5-10, o algo estás haciendo mal. Repensar la función, y dividirla en pedazos más pequeños. Un cerebro humano generalmente puede realizar un seguimiento de aproximadamente 7 cosas diferentes, cualquier cosa más y se confunde. Sabes que eres brillante, pero tal vez te gustaría entender lo que hiciste dentro de 2 semanas.

4.7 Comentarios

Los comentarios son buenos, pero también existe el peligro de comentar en exceso. NUNCA intente explicar CÓMO funciona su código en un comentario; es mucho mejor escribir el código para que el **funcionamiento** sea obvio, y es un desperdicio de tiempo explicar un código mal escrito.

En general, busque que sus comentarios digan QUÉ hace su código, no CÓMO. Un comentario en recuadro que describe la función, el valor de retorno y quién lo llama colocado encima del cuerpo es bueno. Además, trate de evitar poner comentarios dentro del cuerpo de una función; si la función es tan compleja que necesita comentar partes por separado, probablemente debería volver a leer la sección de funciones de nuevo. Puede hacer pequeños comentarios para anotar o advertir sobre algo particularmente inteligente (o feo), pero trate de evitar el exceso. En su lugar, coloque los comentarios al frente de la función, diciéndoles a las personas lo que hace, y posiblemente POR QUÉ lo hace.

Si se utilizan comentarios /* Fix me */ en la línea, por favor, por favor, decir por qué algo necesita ser reparado. Cuando se ha realizado un cambio en la parte afectada del código, elimine el comentario o modifíquelo para indicar que se ha realizado un cambio y necesita pruebas.

4.8 Scripts de Shell y Makefiles

No todos tienen las mismas herramientas y paquetes instalados. Algunas personas usan vi, otros emacs: algunos incluso evitan tener cualquiera de estos paquetes instalados, prefiriendo un editor de texto liviano como nano o el construido en Midnight Commander.

gawk versus mawk - Nuevamente, no todos tendrán gawk instalado, mawk es casi una décima parte del tamaño y, sin embargo, se ajusta al Posix AWK estándar. Si se necesita algún comando específico de gawk oscuro que mawk no proporciona, el script se romperá para algunos usuarios. Lo mismo se aplicaría a mawk. En resumen, use la invocación genérica awk en preferencia a gawk o mawk.

4.9 Convenciones de C++

Los estilos de codificación de C++ siempre terminan en acalorados debates (un poco como los argumentos de emacs versus vi). Una cosa es cierta, sin embargo; el estilo común utilizado por todos los que trabajan en un proyecto conduce a la uniformidad y a código legible

Convenciones de nomenclatura: las constantes #define o enumeraciones debe estar en mayúscula. Justificación: hace que sea más fácil detectar constantes de tiempo de compilación en el código fuente. p.ej. EMC_MESSAGE_TYPE

Las clases y los espacios de nombres deben poner en mayúscula la primera letra de cada palabra y evitar guiones bajos. Justificación: identifica clases, constructores y destructores. p.ej. GtkWidget

Los métodos (o nombres de funciones) deben seguir las recomendaciones C anteriores y no debe incluir el nombre de la clase. Justificación: mantiene un estilo común en fuentes C y C++. p.ej. get_foo_bar()

Sin embargo, los métodos booleanos son más fáciles de leer si evita los guiones bajos y use un prefijo *is* (no debe confundirse con los métodos que manipulan un booleano). Justificación: identifica el valor de retorno como VERDADERO o FALSO y nada más. p.ej. isOpen, isHomed

NO use *Not* en un nombre booleano, solo conduce a confusión al hacer pruebas lógicas. p.ej. isNotOnLimit o is_not_on_limit son MALOS.

Los nombres de las variables deben evitar el uso de mayúsculas y guiones bajos a excepción de nombres locales o privados. El uso de variables globales debería evitarse tanto como sea posible. Justificación: aclara cuáles son variables y cuales son métodos. Ej. Público: axislimit. Ej. Privado: maxvelocity_

Convenciones de nomenclatura de métodos específicos

Los términos get y set deben usarse donde se accede a un atributo directamente. Justificación: indica el propósito de la función o método. p.ej. get_foo set_bar

Para los métodos que involucran atributos booleanos, se prefiere set y reset. Justificación: como arriba. p.ej. set_amp_enable reset_amp_fault

Los métodos intensivos en matemáticas deben usar el cálculo como prefijo. Razón fundamental: mostrar que es computacionalmente intensivo y acaparará la CPU. p.ej. compute_PID

Las abreviaturas en los nombres deben evitarse siempre que sea posible. la excepción es para nombres de variables locales. Justificación: claridad del código. p.ej. se prefiere pointer a ptr, compute a cmp, o compare a (de nuevo) cmp.

Las enumeraciones y otras constantes pueden tener como prefijo un nombre de tipo común p.ej. enum COLOR{COLOR_RED, COLOR_BLUE};

Se debe evitar el uso excesivo de macros y defines. Se prefieren métodos o funciones. Justificación: mejora el proceso de depuración.

Los declaraciones include de archivos de encabezado deben incluirse en la parte superior de un archivo fuente y no disperso por todo el cuerpo. Deberían estar ordenados y agrupados por su posición jerárquica dentro del sistema con los archivos de bajo nivel incluidos primero. Las rutas de archivos include NUNCA deben ser absolutas; use el flag del compilador -I en su lugar. Razón fundamental: Los encabezados pueden no estar en el mismo lugar en todos los sistemas.

Los punteros y las referencias deben tener su símbolo de referencia junto al nombre de la variable y no junto al nombre del tipo. Justificación: Reduce la confusión. p.ej. float *x o int &i

Las pruebas implícitas para cero no deben usarse excepto para variables booleanas. p.ej. if (spindle_speed != 0) NO if (spindle_speed)

Solo las declaraciones de control de bucle deben incluirse en una construcción for(). p.ej. sum = 0; for (i = 0; i < 10; i++) { sum += value[i]; }

NO for (i = 0, sum =0; i < 10; i++) sum += value[i];

Del mismo modo, deben evitarse las declaraciones ejecutables en condicionales. p.ej. if (fd = open(nombre_archivo) es malo.

Deben evitarse las declaraciones condicionales complejas - Introducir variables booleanas temporales en su lugar.

Los paréntesis deben usarse en abundancia en las expresiones matemáticas. No confíe en la precedencia del operador cuando un paréntesis adicional aclararía las cosas.

Nombres de archivo: las fuentes y los encabezados de C++ usan la extensión .cc y .hh. El uso de .c y .h están reservados para C. Los encabezados son para clases, métodos, y declaraciones de estructuras, no para código (a menos que se declaren las funciones inline).

4.10 Estándares de codificación de Python

Utilice el estilo PEP 8 para Código de Python

4.11 Estándares de codificación comp

En la parte de declaración de un archivo .comp, comience cada declaración en la primera columna. Insertar líneas en blanco adicionales cuando ayuden a agrupar items relacionados.

En la parte del código de un archivo .comp, siga el estilo de codificación C normal.

Chapter 5

Construyendo LinuxCNC

5.1 Introducción

Este documento describe cómo construir el software LinuxCNC y la documentación desde las fuentes. Esto es útil principalmente si eres un desarrollador que está modificando LinuxCNC. También puede ser útil si eres un usuario que está probando ramas de desarrollo, aunque también se tiene la opción de instalar paquetes unicos Debian desde buildboot: http://buildbot.linuxcnc.org

5.1.1 Inicio rápido

Si usted es un impaciente, intente esto (suponiendo que ha instalado git):

```
> git clone git://github.com/linuxcnc/linuxcnc.git linuxcnc-dev
> cd linuxcnc-dev/src
> ./autogen.sh
> ./configure --with-realtime = uspace
> make
```

¡Eso probablemente fallará!. No es culpa suya; solo significa que debe leer todo este documento para averiguar cómo solucionar sus problemas. Especialmente la sección sobre Satisfacer Dependencias de Construcción.

Si está manejando un sistema con capacidad de tiempo real (como una instalación desde la imagen de LinuxCNC Live/Install, vea la sección Tiempo Real mas abajo); se necesita un paso de compilación adicional:

```
> sudo make setuid
```

Después de haber compilado con éxito LinuxCNC, es hora de ejecutar las pruebas:

```
> source ../scripts/rip-environment
> runtests
```

¡Esto también podría fallar!. Lea todo este documento, pero especialmente la sección. en Configuración del entorno de prueba.

5.2 Plataformas compatibles

El proyecto LinuxCNC apunta a distribuciones modernas basadas en Debian, que incluyen Debian, Ubuntu y Mint.

Probamos continuamente en las plataformas enumeradas en http://buildbot.linuxcnc.org.

LinuxCNC compila en la mayoría de las otras distribuciones de Linux, aunque la gestión de dependencias será más manual. Siempre son bienvenidos parches para mejorar la portabilidad a nuevas plataformas .

5.2.1 Realtime

LinuxCNC es un controlador de máquina herramienta, y requiere una plataforma en tiempo real para hacer este trabajo. Esta versión de LinuxCNC admite tres plataformas en tiempo real.

RTAI
: De https://www.rtai.org. En el archivo de Debian en http://www.linuxcnc.org hay disponible un kernel de Linux con el parche RTAI. Ver Obtener LinuxCNC para las instrucciones de instalación.

Xenomai
: De https://xenomai.org. Tendrá que compilar u obtener un kernel Xenomai por usted mismo

Preempt-RT
: De https://rt.wiki.kernel.org. Un kernel de Linux con el parche Preempt-RT, que está disponible ocasionalmente en el archivo Debian en https://www.debian.org, y desde la "máquina wayback" en https://snapshot.debian.org.

Para hacer uso de las capacidades en tiempo real de LinuxCNC, ciertas partes necesitan ejecutarse con privilegios root. Para habilitar root para estas partes, ejecute este comando adicional después del `make` que construye LinuxCNC:

```
> sudo make setuid
```

5.2.2 Sin Tiempo Real

LinuxCNC también se puede construir y ejecutar en plataformas que no son de tiempo real, como una instalación normal de Debian o Ubuntu sin ningún kernel especial en tiempo real.

En este modo, LinuxCNC no es útil para controlar máquinas herramientas, pero es útil para simular la ejecución de código G y para probar las partes del sistema que no son en tiempo real (como las interfaces de usuario y algunos tipos de componentes y controladores de dispositivos).

5.3 Modos de compilación

Hay dos formas de tener LinuxCNCen una maquina: "ejecución en el sitio" (run-in-place o RIP, el modo de mayor libertad) y el modo "empaquetado Debian", fácil de usar para el usuario (pero difícilmente modificable).

5.3.1 Compilacion Run-In-Place

En una compilación Run-In-Place, los programas LinuxCNC se compilan desde las fuentes y luego se ejecuta directamente desde el directorio de compilación. Nada queda instalado fuera del directorio de compilación.

El conjunto de pruebas LinuxCNC se ejecuta solo en una compilación Run-In-Place.

La mayoría de los desarrolladores de LinuxCNC compilan principalmente con este modo.

Para una compilacion Run-In-Place, sigua los pasos en la sección Inicio rápido en la parte superior de este documento, posiblemente con diferentes argumentos para `src/configure` y` make`.

5.3.1.1 Argumentos `src/configure`

El script `src/configure` configura cómo será compilado el código fuente. Admite muchos argumentos opcionales.

Enumere todos los argumentos para `src/configure` ejecutando:

```
> cd linuxcnc-dev/src
> ./configure --help
```

Los argumentos más utilizados son:

`--with-realtime=uspace`
: Compilar para cualquier plataforma en tiempo real, o no. Los ejecutables LinuxCNC resultantes se ejecutarán tanto en un kernel de Linux con parches Preempt-RT (que proporcionan control de la máquina en tiempo real) o en un núcleo Linux original (sin parches) (que proporciona simulación de código G pero sin control de máquina en tiempo real). Si los archivos de desarrollo están instalados para Xenomai (típicamente del paquete libxenomai-dev) o RTAI (típicamente desde un paquete con un nombre que comienza por "rtai-modules"), también estarán habilitado soporte para estos núcleos en tiempo real.

`--with-realtime=/usr/realtime-$VERSION`
: Compilación para la plataforma RTAI en tiempo real utilizando el antiguo modelo "kernel realtime". Esto requiere tener un kernel RTAI y los módulos RTAI instalados en `/usr/realtime-$VERSION`. Los ejecutables LinuxCNC resultantes solo se ejecutarán en el kernel RTAI especificado. A partir de LinuxCNC 2.7, esto produce el mejor rendimiento en tiempo real.

`--enable-build-documentation`
: Crear la documentación, además de los ejecutables. Esta opción aumenta significativamente el tiempo requerido para la compilación, ya que construir los documentos consumen bastante tiempo. Si no está trabajando activamente en la documentación es posible que desee omitir este argumento.

5.3.1.2 Argumentos `make`

El comando `make` admite dos argumentos opcionales útiles.

Compilación Paralela
: `make` admite un argumento opcional `-jN` (donde N es un número). Esto permite la compilación paralela con N procesos simultáneos, que puede acelerar significativamente tu construcción.

  Un valor útil para N es la cantidad de CPU's en su sistema de compilación. Usted puede averiguar el número de CPU ejecutando `nproc`.

Compilar un solo objetivo específico
: Si desea compilar solo una parte específica de LinuxCNC, puede nombrar lo que quiere construir en la línea de comando `make`. Por ejemplo, si está trabajando en un componente llamado `froboz`, puede construir su ejecutable con los comandos:

```
> cd linuxcnc-dev/src
> make ../bin/froboz
```

5.3.2 Construyendo paquetes Debian

Al crear paquetes Debian, los programas LinuxCNC se compilan a partir de fuente y luego se almacenan en un paquete Debian completo, con información de dependencias. Esto lleva más tiempo y los programas no se pueden usar hasta que el paquete Debian se instala en una máquina de destino.

Este modo de compilación es principalmente útil cuando se empaqueta el software para entrega a usuarios finales, o para construir el software para una máquina que no tiene instalado el entorno de compilación, o que no tiene acceso a Internet.

La creación de paquetes Debian requiere la herramienta `dpkg-buildpackage`, del paquete `dpkg-dev`, que puede ser instalado con:

```
> sudo apt-get install dpkg-dev
```

La creación de paquetes Debian también requiere que todas las dependencias de compilación estén instaladas, tal como se describe en la sección Satisfacer Dependencias de Construcción.

Una vez que se cumplen esos requisitos previos, la construcción de los paquetes Debian consiste en dos pasos.

El primer paso es generar los scripts y metadatos del paquete Debian desde su clon del repositorio git ejecutando esto:

```
> cd linuxcnc-dev/debian
> ./configure uspace
> cd ..
```

Note
El script `debian/configure` es diferente del script `src/configure`!
El script `debian/configure` necesita diferentes argumentos dependiendo de la plataforma en/para la que está compilando; vea la sección argumentos debian/configure.

Una vez que los scripts del paquete Debian y los metadatos estén configurados, cree el paquete ejecutando `dpkg-buildpackage` (tenga en cuenta que debe ejecutarse desde el directorio `linuxcnc-dev`, **no** desde `linuxcnc-dev/debian`):

```
> dpkg-buildpackage -b -uc
```

5.3.2.1 Argumentos `debian/configure`

El script `debian/configure` configura el paquete Debian. Debe ejecutarse antes de que se puedan ejecutar `dpkg-checkbuilddeps` y `dpkg-buildpackage`.

Admite un solo argumento que especifica la plataforma de tiempo real o no tiempo real. Los valores normales para este argumento son:

`uspace`
: Configura el paquete Debian para Preempt-RT en tiempo real o para no tiempo real (estos dos son compatibles).

`noauto,rtai,xenomai`
: Normalmente, se detectan automáticamente las listas de RTOS para uspace en tiempo real soportados. Sin embargo, si lo desea, puede especificar uno o más de estos RTOS después de `uspace` para habilitar el soporte para estos RTOS. Para deshabilitar la autodetección, especifique `noauto`.

```
Si solo desea el tradicional "módulo de kernel" RTAI en tiempo real, use
`-r` o `$KERNEL_VERSION` en su lugar.
```

`rtai=<nombre del paquete>`
: Si el paquete de desarrollo para rtai lxrt no comienza con "rtai-modules", o si el primer paquete de este tipo aparece en la búsqueda de apt-cache no es el deseado, especifique explícitamente el nombre del paquete.

`-r`
: Configura el paquete Debian para el kernel RTAI actualmente en ejecución. Debe estar ejecutando un kernel RTAI en su máquina de compilación para que esto trabaje!

`$KERNEL_VERSION`
: Configura el paquete debian para la versión de kernel RTAI especificada (por ejemplo, "3.4.9-rtai-686-pae"). Los encabezados del kernel del paquete debian coincidente debe estar instalado en su máquina de compilación (por ejemplo "linux-headers-3.4.9-rtai-686-pae"). Tenga en cuenta que puede *construir* LinuxCNC en esta configuración, pero si no está ejecutando el kernel RTAI coincidente, no podrá ejecutar LinuxCNC, incluyendo el conjunto de pruebas.

5.4 Satisfacer Dependencias de Construcción

En las plataformas basadas en Debian, proporcionamos metadatos de empaquetado que saben qué paquetes de software externos deben instalarse para construir LinuxCNC. Esto se llama dependencias de compilación de LinuxCNC. Usted puede usar estos metadatos para enumerar fácilmente los paquetes requeridos que faltan en su sistema de construcción.

Los sistemas Debian proporcionan un programa llamado `dpkg-checkbuilddeps` que analiza los metadatos del paquete y compara los paquetes enumerados como dependencias de compilación contra la lista de paquetes instalados, y le dice lo que falta.

Primero, instale el programa `dpkg-checkbuilddeps` ejecutando:

```
> sudo apt-get install dpkg-dev
```

Luego, solicite a su clon git LinuxCNC para la generación de metadatos de su paquete Debian:

```
> cd linuxcnc-dev/debian
> ./configure uspace
> cd ..
```

Finalmente, solicite a `dpkg-checkbuilddeps` que haga su trabajo (tenga en cuenta que necesita ejecutarlo desde el directorio `linuxcnc-dev`, **no** desde `linuxcnc-dev/debian`):

```
> dpkg-checkbuilddeps
```

Esto emitirá una lista de paquetes necesarios para construir LinuxCNC en su sistema, pero que aún no están instalados. Instalelos todos con `sudo apt-get install`, seguido de los nombres de los paquetes.

Puede volver a ejecutar `dpkg-checkbuilddeps`, en el momento que desee, para enumerar cualquier paquete faltante.

5.5 Configuración del entorno

Esta sección describe los pasos especiales necesarios para configurar una máquina para ejecutar los programas LinuxCNC, incluidas las pruebas.

5.5.1 Aumentar el límite de memoria bloqueada

LinuxCNC intenta mejorar su latencia en tiempo real bloqueando la memoria que utiliza en la RAM. Hace esto para evitar que el sistema operativo intercambie LinuxCNC al disco, lo que tendría malos efectos sobre la latencia.

Normalmente, el bloqueo de memoria en la RAM está mal visto y el sistema operativo establece un límite estricto sobre la cantidad de memoria que un usuario puede tener bloqueada.

Cuando se utiliza la plataforma de tiempo real Preempt-RT, LinuxCNC se ejecuta con suficiente privilegio para aumentar su límite de bloqueo de memoria. Cuando use la plataforma RTAI en tiempo real, no tiene suficientes privilegios, y el usuario debe elevar el límite de bloqueo de memoria.

Si LinuxCNC muestra el siguiente mensaje al inicio, el problema es el límite de memoria bloqueada configurado de su sistema:

```
RTAPI: ERROR: failed to map shmem
RTAPI: Locked memory limit is 32KiB, recommended at least 20480KiB.
```

Para solucionar este problema, agregue un archivo llamado `/etc/security/limits.d/linuxcnc.conf` (como root) con su editor de texto favorito (por ejemplo, `sudo gedit/etc/security/limits.d/linuxcnc.conf`). El archivo debe contener la siguiente línea:

```
* - memlock 20480
```

Cierre la sesión y vuelva a iniciar sesión para que los cambios surtan efecto. Verificar que el límite de bloqueo de memoria se aumentó con el siguiente comando:

```
> ulimit -l
```

5.6 Opciones para ver el repositorio de git

Las instrucciones Inicio rápido en la parte superior de este documento hablan de hacer un clon local anónimo desde nuestro repositorio git en http://github.com/linuxcnc/linuxcnc.git. Esta es la manera más rápida y fácil de empezar. Sin embargo, hay otras opciones a considerar.

5.6.1 Bifurcación en Github (fork)

El repositorio git del proyecto LinuxCNC está en http://github.com/LinuxCNC/linuxcnc. github es un popular servicio de alojamiento git y un sitio web para compartir código. Puede crear fácilmente (y sin costo) una bifurcacion de nuestro repositorio de git en github, y usarlo para rastrear y publicar sus cambios.

Después de crear su propia bifurcación github de LinuxCNC, clónela en su máquina de desarrollo y proceda con sus modificaciones.

Nosotros, el proyecto LinuxCNC, esperamos que comparta sus cambios, para que la comunidad pueda beneficiarse de su trabajo. Github hace que compartir sea muy fácil; después de pulir sus cambios y añadirlos a su bifurcacion github, envíenos una solicitud de extracción.

Chapter 6

Agregar elementos al Selector de Configuraciones

Se pueden agregar configuraciones de ejemplo al Selector de configuración por dos métodos:

- Aplicaciones auxiliares: aplicaciones instaladas independientemente con un paquete deb que pueden colocar subdirectorios de configuración en un directorio del sistema especificado. El nombre del directorio se especifica utilizando el script de shell linuxcnc_var:

```
$ linuxcnc_var LINUXCNC_AUX_EXAMPLES
/usr/share/linuxcnc/aux_examples
```

- Configuración de tiempo de ejecución: el selector de configuración también puede ofrecer subdirectorios de configuración especificados en tiempo de ejecución utilizando una variable de entorno exportada (LINUXCNC_AUX_CONFIGS). Esta variable debe ser una lista de rutas de uno o más directorios de configuraciones separados por (:). Normalmente, esta variable se establecería en un shell de inicio de linuxcnc o en un script de inicio de usuario ~/.profile . Ejemplo:

```
export LINUXCNC_AUX_CONFIGS=~/myconfigs:/opt/otherconfigs
```

Chapter 7

Contribuir a LinuxCNC

7.1 Introducción

Este documento contiene información para desarrolladores sobre la infraestructura de LinuxCNC, y describe las mejores prácticas para contribuir con código y actualizaciones de documentación para el proyecto LinuxCNC.

En este documento, "fuente" significa tanto el código fuente de programas y bibliotecas, como el texto fuente para la documentación.

7.2 Comunicación entre desarrolladores de LinuxCNC

Las dos formas principales en que los desarrolladores de proyectos se comunican entre sí son:

- Vía IRC, en #linuxcnc-devel en FreeNode.
- Por correo electrónico, en la lista de correo de los desarrolladores: https://lists.sourceforge.net/lists/listinfo/emc-developers

7.3 El proyecto LinuxCNC Source Forge

Utilizamos Source Forge para las listas de correo: http://sourceforge.net/p/emc/mailman/

7.4 El Sistema de Control de Revisiones git

Todas las fuentes LinuxCNC se mantiene en el sistema de control de revisión git [1].

7.4.1 LinuxCNC repositorio oficial de git

El repositorio oficial git de LinuxCNC está en https://github.com/linuxcnc/linuxcnc/

Cualquiera puede obtener una copia de solo lectura del árbol fuente LinuxCNC a través de git:

```
git clone https://github.com/linuxcnc/linuxcnc linuxcnc-dev
```

Si usted es un desarrollador con acceso push, entonces siga las instrucciones de github para configurar un repositorio desde el que pueda hacer push.

[1] http://git-scm.com/

Tenga en cuenta que el comando clone coloca el repositorio local de LinuxCNC en un directorio llamado `linuxcnc-dev`, en lugar del predeterminado` linuxcnc`. Esto se debe a que el software LinuxCNC por defecto espera configuraciones y Programas de código G en un directorio llamado `$ HOME /linuxcnc`, y tener el repositorio git en el mismo sitio es confuso.

Los problemas y las solicitudes pull son bienvenidos en github: https://github.com/LinuxCNC/linuxcnc/issues https://github.com/-LinuxCNC/linuxcnc/pulls

7.4.2 Uso de git en el proyecto LinuxCNC

Utilizamos los flujos de trabajo git "merging upwards" y "topic branches" descritos aquí:

https://www.kernel.org/pub/software/scm/git/docs/gitworkflows.html

Tenemos una rama de desarrollo llamada `master`, y una o más ramas estables con nombres como `2.6` o `2.7` que indican el número de versión de los lanzamientos que hacemos.

Las correcciones de errores van en la rama estable aplicable más antigua, y esa rama se fusiona con la siguiente rama estable más nueva, y así sucesivamente hasta `master`. El committer, o confirmador de la corrección de error, puede hacer las fusiones por sí mismo, o dejarlas a otra persona.

Las nuevas características generalmente van en la rama `master`, pero algunos tipos de características (específicamente controladores de dispositivo y documentación bien aislados) pueden (a discreción de los gerentes de ramas estables) entrar en una rama estable y fusionarse como lo hacen las correcciones de errores.

7.4.3 tutoriales git

Hay muchos tutoriales gratuitos excelentes de git en Internet.

El primer lugar para buscar es probablemente la página de manual "gittutorial". Se puede acceder a esta página de manual ejecutando "man gittutorial" en un terminal (si tiene instaladas las páginas de manual de git). gittutorial y su documentación de seguimiento también está disponible en línea aquí:

- tutorial de git: https://www.kernel.org/pub/software/scm/git/docs/gittutorial.html
- tutorial git 2: https://www.kernel.org/pub/software/scm/git/docs/gittutorial-2.html
- Everyday git with 20 commands or so: https://www.kernel.org/pub/software/scm/git/docs/giteveryday.html
- Manual del usuario de Git: https://www.kernel.org/pub/software/scm/git/docs/user-manual.html

Para una documentación más completa de git, vea el libro "Pro Git": http://git-scm.com/book donde hay versiones en varios idiomas, incluido Español.

Otro tutorial en línea que se ha recomendado es "Git for the Lazy": http://wiki.spheredev.org/Git_for_the_lazy

7.5 Descripción general del proceso

La descripción general de alto nivel de cómo contribuir con cambios a la fuente es la siguiente:

- Comunicarse con los desarrolladores del proyecto e informarles en que está trabajando.
- Clonar el repositorio git
- Realizar sus cambios en una rama local, asegurándose de "cerrar" sus commits de acuerdo con nuestra política de aprobación (ver más abajo).
- Agregar documentación y pruebas es una parte importante al agregar una nueva característica. De lo contrario, otros no sabrán cómo usar su función, y si otros cambios corrompen su función, puede pasar desapercibidos sin una prueba.

- Comparta sus cambios con los otros desarrolladores de proyectos de una de estas maneras:
  - Haga push de su rama a github y cree una solicitud de extracción pull de github a https://github.com/linuxcnc/linuxcnc (esto requiere una cuenta github)
  - Haga push de su rama a un repositorio de git visible públicamente (como github, bitbucket, su propio servidor de acceso público, etc.) y comparta su ubicación en la lista de correo de emc-developers, o
  - Envíe sus confirmaciones por correo electrónico a la lista de correo de emc-developers (use `git format-patch` para crear parches)
- Defienda su parche
  - Explique qué problema aborda y por qué debería incluirse en LinuxCNC
  - Sea receptivo a las preguntas y comentarios de la comunidad de desarrolladores.
  - No es raro que un parche pase por varias revisiones antes de ser aceptado

7.6 Configuración de git

Para ser considerada la inclusión en las fuentes de LinuxCNC, los commits deben tener campos de Autor correctos que identifiquen al autor del commit. Una buena manera de garantizar esto es establecer su configuración global de git:

```
git config --global user.name "Su nombre completo"
git config --global user.email "SuCorreo@example.com"
```

Use su nombre real (no un identificador) y una dirección de correo electrónico clara.

7.7 Uso efectivo de git

7.7.1 Contenidos de Commits

Mantenga sus commits pequeños y directos. Cada commit debe aportar un cambio lógico al repositorio.

7.7.2 Escribir buenos mensajes con los commits

Mantenga los mensajes de commits alrededor de 72 columnas de ancho (de modo que en un tamaño predeterminado de ventana de terminal, no se partan cuando se muestren con `git log`).

Use la primera línea como un resumen de la intención del cambio (casi como la línea de asunto de un correo electrónico). Sígalo con una línea en blanco, y luego un mensaje más largo explicando el cambio. Ejemplo:

```
Deshacerse de RTAPI_SUCCESS, usar 0 en su lugar

La prueba "retval < 0" debería ser familiar; es el mismo tipo de
prueba que se utiliza en el espacio de usuario (devuelve -1 para error) y
en el espacio de kernel (devuelve -ERRNO para error)
```

7.7.3 Commit a la rama adecuada

Las correcciones de errores deben ir en la rama aplicable más antigua. Las nuevas funciones deberían ir a la rama maestra. Si no está seguro de dónde pertenece un cambio, pregunte en el irc o en la lista de correo.

7.7.4 Use múltiples commits para organizar los cambios

Cuando sea apropiado, organice sus cambios en una rama (una serie de commits) donde cada commit es un paso lógico hacia su objetivo máximo. Por ejemplo, primero factorice un código complejo en una nueva función. Luego, en un segundo commit, corrija algún error subyacente. Después, en un tercer commit, agregue una nueva característica que sea fácil para la refactorización y que no hubiera funcionado sin arreglar aquel error.

Esto es útil para los revisores, porque es más fácil ver que el paso "factorizar el código en una nueva función" era correcto, sin otras ediciones mezcladas; es más fácil ver que el error se corrige cuando el cambio que lo arregla es independiente de la nueva característica; y así sucesivamente.

7.7.5 Siga el estilo del código circundante

Haga un esfuerzo por seguir el estilo de sangría predominante en el código. En particular, los cambios en los espacios en blanco hacen que sea más difícil para otros desarrolladores rastrear cambios a lo largo del tiempo. Cuando se debe reformatear código, hágalo como un commit separado de cualquier cambio semántico.

7.7.6 Simplifique historias complicada antes de compartirla con otros desarrolladores

Con git, es posible grabar cada edición y falso comienzo como un commit separado. Esto es muy conveniente como una forma de crear puntos de control durante el desarrollo, pero a menudo no se quiere compartir estos falsos comienzos con otros.

Git proporciona dos formas principales de limpiar el historial, las cuales se pueden hacer libremente antes de compartir el cambio:

`git commit --amend` le permite hacer cambios adicionales a su último commit, modificando opcionalmente también el mensaje del mismo. Utilizar esto si se dio cuenta de inmediato de que dejó algo fuera del commit o para reescribir el mensaje.

`git rebase --interactive upstream-branch` le permite volver a través de cada commit realizado desde que bifurco su rama de características desde la rama superior, posiblemente editando, descartando o comprimiendo (combinando) commits con otros. Rebase también se puede usar para dividir commits individuales en múltiples commits nuevos.

7.7.7 Asegúrese de que cada commit compila

Si su cambio consta de varios parches, `git rebase -i` puede usarse para reordenarlos en una secuencia de commits que establezca más claramente los pasos de su trabajo. Una consecuencia potencial de reordenar parches es que podrían aparecer dependencias incorrectas, por ejemplo, introducir el uso de una variable y, en un parche posterior, la declaración de esa variable.

Si bien la rama HEAD se construirá, no todos los commits podrán compilarse en tal caso. Eso rompe `git bisect`, algo que se podría usar más tarde para encontrar el commit que introdujo un error. Así que más allá de asegurarse que su rama compila es importante asegurar que cada commit compila también.

Hay una forma automática de verificar una rama para que con cada commit siga siendo compilable. - ver http://dustin.sallings.org/-2010/03/28/git-test-sequence.html y el código en https://github.com/dustin/bindir/blob/master/git-test-sequence. Úselo de la siguiente manera (en este caso probando cada confirmación desde origen/maestro a HEAD, incluida la ejecución de pruebas de regresión):

```
cd linuxcnc-dev
git-test-sequence origin/master..  '(cd src && make && ../scripts/runtests)'
```

Esto informará *Todo bien* o *Se rompió en <commit>*

7.7.8 Renombrar archivos

Utilice la capacidad de cambiar el nombre de los archivos con mucho cuidado. Al igual que correr sangría en archivos individuales, los cambios de nombre hacen que sea más difícil de seguir cambios en el tiempo. Como mínimo, debe buscar consenso en IRC o La lista de correo de que el cambio de nombre es una mejora.

7.7.9 Prefiera "rebase"

Utilice `git pull --rebase` en lugar de` git pull` para mantener un buen historial lineal. Con rebase, siempre se retiene el trabajo como revisiones delante de origen/maestro, para que pueda hacer cosas como `git format-patch` para compartir con otros sin push al repositorio central.

7.8 Otras formas de contribuir

Hay muchas formas de contribuir a LinuxCNC, que no se abordan en este documento. Estas formas incluyen:

- Responder preguntas en el foro, listas de correo y en IRC
- Informar errores en el seguidor de errores, foro, listas de correo o en IRC
- Ayudando a probar características experimentales

Chapter 8

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.

Chapter 9

Seccion Legal

N.T. Estos textos legales solo se muestra en Inglés ya que las traducciones no se reconocen oficialmente.

9.1 Terminos de Copyright

9.2 Licencia GNU de Documentation Libre

GNU Free Documentation License Version 1.1, March 2000

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.

Chapter 10

Index

V