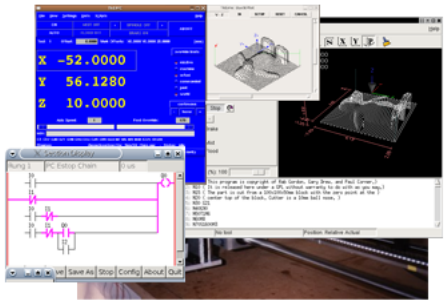
EMC²

The Enhanced Machine Controller



www.linuxcnc.org

Manuel de l'utilisateur V2.2

The EMC Team

26 janvier 2009

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.

Table des matières

Première partie

Introduction et Installation d'EMC2

Chapitre 1

Installer le logiciel EMC2

1.1 Introduction

Un des problèmes les plus souvent évoqués par les utilisateurs à propos d'EMC a été qu'il ne s'installait pas de lui-même. Il fallait qu'ils récupèrent les sources, les compilent eux-mêmes, essaient d'appliquer un noyau Linux patché RT, etc. Les développeurs d'EMC2 ont donc choisi une distribution standard appelée Ubuntu[1].

Ubuntu a été choisi parce-qu'il est parfaitement dans l'esprit Open Source d'EMC2 :

- Ubuntu restera toujours gratuit, il n'y a pas de frais pour l'édition "enterprise edition", nous faisons de notre mieux pour rendre notre travail disponible à tous dans les mêmes termes de gratuité.
- Ubuntu fournit un support professionnel commercial à des centaines de sociétés dans le monde, vous aurez peut être besoin de ces services. Chaque nouvelle version d'Ubuntu reçoit des mises à jour de sécurité gratuites pendant 18 mois après sa publication, certaines versions sont supportées plus longtemps.
- Ubuntu utilise le meilleur en termes de traduction et d'accessibilité à ses infrastructures parmi ce que la communauté du logiciel libre peut offrir et pour faire qu'Ubuntu soit aprécié par autant d'utilisateurs que possible.
- Ubuntu est publié régulièrement selon un calendrier précis ; Une nouvelle version est publiée tous les six mois. Vous pouvez utiliser la dernière version stable ou aider à stabiliser la version en cours de développement.
- La communauté Ubuntu est entièrement dévouée aux principes de développement du logiciel libre ; elle encourage tout le monde à utiliser des logiciels libres et open source, les améliorer et les distribuer.

1.2 La page de téléchargement d'EMC

Vous pouvez trouver l'annonce des plus récentes versions publiées d'EMC2 sur le site www.linuxcnc.org. Les versions d'EMC2 sont fournies de deux manières (sources et paquets binaires). Les sources (décrites dans le manuel du développeur) consistent en un tarball (emc2-<version>.tar.gz), que vous devez charger et décompacter dans votre répertoire home.

Le présent document (plus orienté utilisateur final) expliquera seulement comment installer les paquets binaires sur une distribution Ubuntu[2].

[1] Le mot "Ubuntu" est un ancien mot Africain, signifiant "humanité aux autres". Ubuntu signifie aussi "Je suis ce que je suis à cause de ce que nous sommes tous". La distribution Ubuntu Linux amène l'esprit d'Ubuntu au monde du logiciel. Vous pouvez en lire plus à ce propos ici : http://www.ubuntu-fr.org/

[2] Pour plus d'informations sur les autres variantes Linux, lisez le Manuel du développeur ou demandez de l'aide sur la Liste de diffusion http://sourceforge.net/mail/?group_id=6744.

1.3 Le live CD d'EMC2

Les développeurs d'EMC2 ont créé un Live-CD basé sur Ubuntu 6.06 qui vous permet d'essayer EMC2 avant de l'installer, c'est également une manière facile d'installer ensemble Ubuntu et EMC2.

Téléchargez l'image image ISO http://linuxcnc.org/iso/emc2-ubuntu6.06-desktop-i386.iso (Miroir EU http://dsplabs.utt.ro/~juve/emc/) et gravez la sur un CD. (la somme MD5 du CD est vérifiable)

Quand vous bootez avec ce CD dans le lecteur de votre machine, vous pouvez voir et expérimenter un environnement identique à celui d'EMC2 qui sera le vôtre si vous choisissez de l'installer.

Si cette démonstation vous a convaicu, cliquez sur l'icône Install du bureau, répondez à quelques questions (votre nom, fuseau horaire, mot de passe) et l'installation terminera en quelques minutes.

Cette installation vous apportera tout les avantages du support de la communauté Ubuntu avec la configuration automatique d'EMC2. Quand une mise à jour d'EMC2 sera publiée, le gestionnaire de paquets vous le fera savoir et vous permettra une mise à jour aisée.

1.4 Script d'installation d'EMC2

Il est également possible d'utiliser un simple script d'installation d'emc2 sur Ubuntu pour les utilisateurs ayant déjà une installation existante d'Ubuntu. Il lance la commande expliquée dans 1.5.

Pour l'utiliser vous devez :

- Charger le script depuis http://linuxcnc.org/dapper/emc2-install.sh (Pour Ubuntu 6.06)
- Le sauver sur votre bureau. Faire un clic droit sur son icône, sélectionner Propriétés. Choisir l'onglet Permissions et cocher Propriétaire : Exécuter. Fermer la fenêtre des propriétés.
- Maintenant double-cliquez sur l'icône emc2-install.sh, et choisissez "Run in Terminal". Une console va apparaître et vous demander votre mot de passe.
- Quand l'installation vous demande si vous voulez installer les paquets d'EMC2, pressez Entrée pour accepter. Laissez ensuite l'installation se poursuivre jusqu'à la fin.
- Quand elle est terminée, éjectez le CD puis vous devrez redémarrer votre machine (Système > Quitter > Redémarrer l'ordinateur). Quand vous aurez redémarré vous pourrez alors lancer EMC2 via le menu Applications > CNC.
- Si vous n'êtes pas prêt pour configurer votre machine, essayez la configuration sim-AXIS ; elle démarre en mode "machine simulée" qui ne requiert le raccordement d'aucun matériel.
- Maintenant que l'installation est terminée, Ubuntu vous avertira quand des mises à jour d'EMC2 seront disponibles. Quand ça arrivera, vous pourrez mettre à jour facilement et automatiquement avec le gestionnaire de mises à jour.

1.5 Installation manuelle par apt-get.

Cette petite section décrira comment installer EMC2 sur Ubuntu 6.06 “Dapper Dreake” en utilisant les commandes apt dans une console. Si vous connaissez un peu Linux et Debian, ça va être facile. Sinon, vous devriez peut être lire 1.4.

Premièrement, ajoutez le dépôt à /etc/apt/sources.list :

```
$ sudo sh -c 'echo "deb http ://www.linuxcnc.org/emc2/ dapper emc2.2" >>/etc/apt/sources.list;'
$ sudo sh -c 'echo "deb-src http ://www.linuxcnc.org/emc2/ dapper emc2.2" >>/etc/apt/sources.list'
```

Puis faites les mises à jour et l'installation d'emc2 avec :

```
$ sudo apt-get update
$ sudo apt-get install emc2
```

Ces commandes vont installer correctement les paquets emc2 avec toutes leurs dépendances[3].

Vous pourriez avoir une alarme indiquant que les paquets proviennent d'une source non vérifiée (ce qui voudrait dire que votre ordinateur ne reconnaît pas la signature GPG des paquets). Pour corriger cette situation, appliquez les commandes suivantes :

```
$ gpg --keyserver pgpkeys.mit.edu --recv-key BC92B87F
$ gpg -a --export BC92B87F | sudo apt-key add -
```

[3]Les dépendances sont un des atouts majeurs des distributions basées sur Debian. Elles assurent que vous avez la totalité de ce qui doit être installé. Même dans un cas comme emc2 qui nécessite un noyau de Linux patché pour travailler en temps réel, ainsi que toutes les librairies indispensables.

Chapitre 2

Compiler EMC2 depuis les sources

2.1 Introduction

Quelques difficultés sont à surmonter quand vous commencez à installer EMC2, son téléchargement et l'installation du software proprement dit. L'ensemble des fichiers d'EMC2 sont placés dans le dépôt cvs.linuxcnc.org, c'est un dépôt avec gestion des versions (CVS). EMC2 est également disponible en paquets pré-compilés (pour différentes plateformes) pour téléchargement depuis ce site.

L'installation peut être une tâche compliquée pour quelqu'un de nouveau sous Linux. La partie la plus dure étant d'appliquer le patch temps réel (Real Time Linux) au noyau. Après ça, installer EMC2 est assez facile. Celà dit, il est dorénavant possible aux utilisateurs de profiter d'une possibilité totalement nouvelle, il leur suffit d'installer Ubuntu (une distribution Linux vraiment conviviale), puis d'exécuter un simple script d'installation, et ils auront alors un EMC2 directement en état de marche sur un noyau temps réel. Les informations pour accéder à cette solution sont diponibles sur www.linuxcnc.org à la page Download.

2.2 Page de téléchargement EMC

Vous pouvez trouver l'annonce des versions les plus récentes d'EMC2 sur www.linuxcnc.org. Les versions d'EMC2 sont fournies de deux manières, sources et paquets binaires. Les sources (described furtheron) sont sous forme de fichiers tarball (emc2-version.tar.gz), que vous devez télécharger et décompacter dans votre répertoire home.

2.3 Gestion des versions d'EMC2

EMC2 utilise un modèle de versions similaire (bien que simplifié) à celui utilisé par Debian. Il y a tout le temps trois versions d'EMC2. Debian utilise "stable", "testing" et "unstable". Nous utilisons "Released", "Testing" et "Head". Pour les dernières informations, cliquez sur la version qui vous intéresse.

Released est exactement ça, une version publiée d'EMC2 avec un numéro de version. Elle a été testée par beaucoup de développeurs et de béta testeurs avant d'être publiée, elle est utilisable par la moyenne des utilisateurs. Les développeurs et réguliers des IRC/mailing list sont prêts à aider ceux qui démarrent avec une version "released". **"Released"** est disponible sous plusieurs formes, incluant .debs pour Ubuntu et tarballs de sources pour une compilation locale. Il y a un dépôt Debian qui a toujours la dernière version "released" (elle permet donc de faciliter les mises à jour d'une version stable).

Testing est une version d'EMC2 qui est prête pour le "beta testing" mais pas pour une publication générale. Avant qu'une version soit labellisée **testing** elle doit d'abord être compilée et doit démarrer sur différentes plateformes, mais il y aura probablement des limitations et divers problèmes. La page **Testing** du wiki est prévue pour lister les problèmes connus et leurs solutions, mais il reste probablement aussi des bugs non découverts. Puisque la version **Testing** est un software "beta", il ne doit pas être utilisé pour tout ce qui est critique. Les utilisateurs de la version **Testing** doivent comprendre qu'il s'agit d'un software en beta et qu'ils doivent être disposés à donner des rapports de bugs détaillés si quelque chose ne va pas. **Testing** est disponible principalement comme une balise en CVS, toutefois pour la commodité des testeurs, un dépôt "testing" debian et/ou des tarballs peuvent aussi être disponibles. C'est le conseil d'administration d'EMC qui décide quand une version "Testing" est digne de devenir "Released". C'est une décision formelle, présentée par voix de motion aux votes du conseil d'administration ou votes par la mailing liste de l'IRC.

TRUNK est un terme CVS pour indiquer l'emplacement des versions en début de développement. Une version **TRUNK** peut souvent être non fonctionnelle. Lorsque la version **TRUNK** sera réputée digne par de nombreux testeurs soit un grand nombre de personnes, la balise **"Testing"** lui sera appliquée. C'est une décision informelle, prise par consensus à la tête des développeurs, habituellement sur l'IRC. Le développement continue immédiatement et un autre **TRUNK** diverge de cette nouvelle version **Testing**. **TRUNK** n'a pas de numéro de version, au cours d'un veek-end chargé il peut changer littéralement toutes les 10 minutes.

2.4 Téléchargement et compilation des sources.

Les quelques sections suivantes décriront comment se procurer les sources d'EMC2 et les compiler.

Pour les télécharger, allez simplement sur www.linuxcnc.org à la page “Download” et prenez les tarballs de la dernière version “release” ou “testing”.

Quand vous les avez dans votre répertoire home, il faut les extraire, ouvrez une console et faites :

```
$ cd ~/
$ tar xzvf emc2-version.tar.gz
```

Puis vous devez décider quel type d'installation vous voulez. Il y a deux possibilités pour essayer EMC2 :

Installed Comme la plupart des autres logiciels sous Linux, les fichiers sont placés dans des répertoires système, ils sont automatiquement disponibles à tous les utilisateurs de l'ordinateur.[1]

Run-in-place Tous les dossiers sont conservés à l'intérieur du répertoire EMC2. Cette option est utile pour essayer EMC2, surtout quand il existe déjà une autre version d'EMC2 installée sur le système..

2.4.1 Télécharger une version CVS

Si vous souhaitez utiliser la version TRUNK d'EMC2, veuillez suivre les instructions de notre wiki pour obtenir le code source : : http://wiki.linuxcnc.org/cgi-bin/emcinfo.pl?CVS

2.5 Installed

EMC2 suit la manière standard de la compilation de logiciel sous linux. Pour compiler il suffit de se rendre dans le répertoire des sources :

[1] Le paquet pré-installé pour Ubuntu Linux utilise la méthode “installé”

```
$ cd ~/emc2/src
```

et d'y lancer ces commandes :

```
$ ./configure
$ make && sudo make install
```

Pour le lancer, tapez 'emc'.

2.6 Run-in-place

Si vous voulez seulement tester le logiciel avant de l'installer, ou si vous avez peur d'écraser une version déjà existante, vous pouvez essayer le mode Run-In-Place (RIP). Dans ce mode, il n'y a aucune installation et aucun fichier ne sera placé en dehors du répertoire `~/emc2`.

Faites juste :

```
$ cd ~/emc2/src
```

puis tapez ces commandes :

```
$ ./configure --enable-run-in-place
$ make && sudo make setuid
```

Dans une console, où vous voulez utiliser EMC2, tapez :[2]

```
$ . ~/emc2/scripts/emc-environment
```

Jusqu'à ce que vous fermiez la console, il sera mis en place afin que les programmes et les pages de manuel soient disponibles sans avoir à se référer au chemin à chaque fois. Ensuite vous pouvez lancer EMC2 en faisant :

```
$ emc
```

2.7 Simulateur

Pour installer EMC2 sur un système sans noyau temps réel, ajoutez `--enable-simulator` à la ligne de commande `configure`. Dans ce mode, seule la partie purement programme d'EMC2 démarrera. Aucun matériel n'aura à être contrôlé, les timmings ne sont pas garantis, mais les autres fonctionnalités de HAL, EMC2 et ses diverses interfaces sont disponibles. Pour utiliser ce mode ajoutez `--enable-run-in-place` à la commande configure, l'étape du `sudo make setuid` n'est pas nécessaire.

2.8 Editer et recompiler

Vous pouvez avoir besoin de recompiler le code d'EMC2 pour diverses raisons. Vous pouvez avoir à modifier le code source, ou vous pouvez avoir seulement téléchargé quelques nouveaux fichiers. Pour recompiler, tapez les commandes suivantes :

[2] En tapant cette commande dans dans le script de démarrage de la console, comme `~/.bash_profile`, vous n'aurez plus à la taper manuellement dans la fenêtre de chaque console.

```
$ cd ~/emc2/src
$ make && sudo make install # pour le run-installed
$ make && sudo make setuid  # pour le run-in-place
$ make                      # pour le run-in-place en simulateur
```

Le processus de compilation est suffisamment performant pour ne recompiler que ce qui est affecté par vos changements.

Part II

Configuring EMC2

Chapitre 3

Démarrage d'EMC

3.1 Lancer emc

Après installation, emc2 se lance comme un autre programme Linux : depuis un terminal en passant la commande **emc**, ou depuis le sous-menu **Applications -> CNC**.

3.2 Sélecteur de configuration

Par défaut, le dialogue Sélecteur de configuration s'affichera la première fois que vous lancerez emc. Vos propres configurations personnalisées s'affichent dans le haut de la liste, suivies par les différentes configurations fournies en exemple.[1] Figure 3.1 montre l'apparence de la fenêtre du sélecteur de configuration.

Cliquez dans la liste, sur les différentes configurations pour afficher les informations les concernant. Double-cliquez sur une configuration ou cliquez OK pour démarrer EMC2 avec cette configuration. Cochez la case "Créer un raccourci sur le bureau" puis cliquez OK pour ajouter une icône sur le bureau d'Ubuntu. Cette icône vous permettra par la suite de lancer directement EMC2 avec cette configuration, sans passer par le sélecteur de configuration.

Quand vous choisissez un exemple de configuration dans le sélecteur, un dialogue vous demandera si vous voulez en faire une copie dans votre répertoire home. Si vous répondez oui, un dossier "emc2" autorisé en écriture sera créé, il contiendra un jeu de fichiers que vous pourrez éditer pour les adapter à vos besoins. Si vous répondez "non", emc démarrera mais pourra se comporter de façon étrange, par exemple, les décalages d'origine pièce entrés avec la commande "Toucher" ne seront pas pris en compte, ce comportement est lié à ce moment, à l'absence de répertoire autorisé en écriture sans lequel les paramètres ne peuvent être enregistrés.

3.3 Les étapes suivantes dans la configuration

Si votre machine est pilotée avec des signaux de type : "pas et direction" (step/dir) via le port parallèle du PC, lisez le chapitre "Stepconf". Pour les autres types, trouvez l'exemple de configuration qui utilise la même interface matériel que votre machine et personnalisez la pour l'adapter exactement à votre matériel. Pour cela, faites en une copie dans votre répertoire home (voir chapitre : 1.2) et passez à la lecture du chapitre "INI Configuration".

[1] Etant donné que chaque exemple de configuration utilise un type différent d'interface matérielle, la plupart ne fonctionneront pas sur votre système. Les configurations listées dans la catégorie "Sim" (simulation) fonctionneront toutes, même sans matériel raccordé.

FIG. 3.1 – Sélecteur de configuration pour EMC2

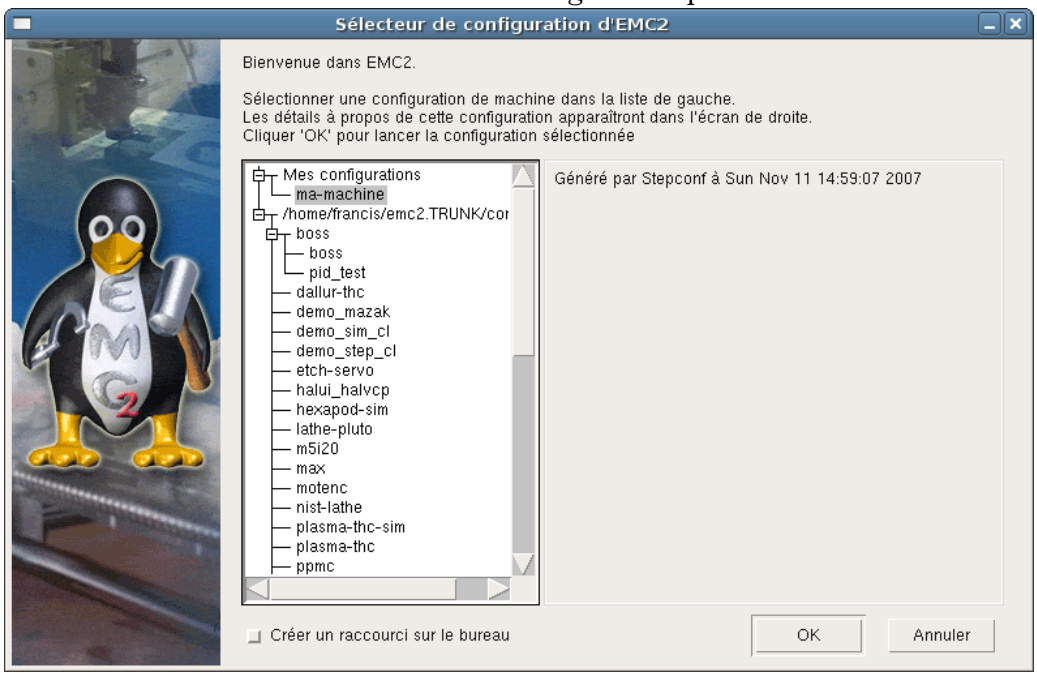
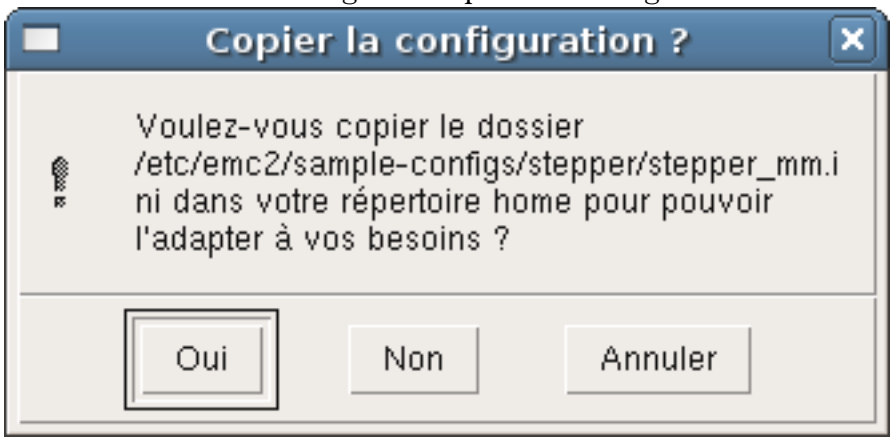


FIG. 3.2 – Dialogue de copie de la configuration



Chapitre 4

Stepconf : Assistant à la configuration d'EMC2 pour machines CNC de type "Step & Direction"

Ce chapitre décrit certains des paramètres les plus communs que les utilisateurs doivent adapter lors d'une nouvelle installation. En raison des multiples possibilités de configuration d'EMC2, il est très difficile de les documenter toutes et de garder ce document relativement court. EMC2 est capable de contrôler une large gamme de machines utilisant différentes interfaces matérielles. Stepconf est un programme qui génère des fichiers de configuration EMC pour une classe spécifique de machine CNC : celles qui sont pilotées via un **port parallèle** standard et contrôlées par des signaux de type **step & direction**.

4.1 Instructions pas à pas

4.1.1 Informations sur la machine

Nom de la machine Choisissez un nom pour votre machine. Utilisez uniquement des lettres, des chiffres ou "-" et "_".

Configuration des axes Choisissez votre type de machine : XYZ (fraiseuse), XYZA (machine 4-axes) ou XZ (tour).

L'unité utilisée par la machine Choisissez entre pouce ou millimètre. Toutes les questions suivantes (telles que la longueur des courses, le pas de la vis, etc) devront obtenir des réponses dans cette unité.

Caractéristiques du pilote Si vous avez un des pilotes énumérés dans la liste déroulante, cliquez sur lui. Sinon, cherchez dans la documentation du pilote de votre matériel, les 4 valeurs de timing et entrez les. Si la fiche donne des valeurs en microsecondes, multipliez les par 1000. Par exemple, pour 4.5µs entrez 4500.

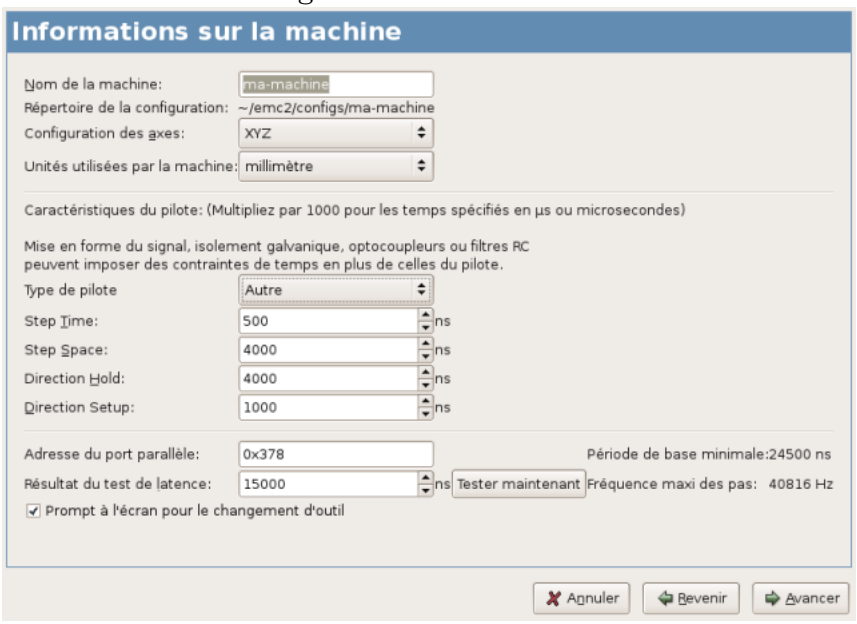
D'éventuels traitements des signaux, une opto-isolation ou des filtres RC, peuvent imposer des contraintes de temps supplémentaires aux signaux, il convient de les ajouter à celles du pilote.

Résultat du test de latence Entrez ici, le résultat du test de latence. (voir la section 4.5)

Fréquence maxi des pas Affiche la valeur calculée de la fréquence maximum des pas que la machine devrait atteindre avec les paramètres de cette configuration.

Période de base minimum En se basant sur les caractéristiques du pilote et sur le résultat du test de latence, Stepconf détermine automatiquement la période de base la plus petite utilisable (BASE_PERIOD). La fréquence maxi des pas est calculée sur la même BASE_PERIOD.

FIG. 4.1 – Page d'informations sur la machine



Prompt à l'écran au changement d'outil Si cette case est cochée, EMC va faire une pause et demander de changer l'outil lorsque M6 est rencontré. Laissez cette case cochée, sauf si vous envisagez d'ajouter la gestion personnalisées d'un changeur automatique d'outils dans un fichier hall.

4.1.2 Réglage du port parallèle

Pour chacune des pins, choisir le signal qui correspond au brochage de votre port parallèle. Cochez la case "inverser" si le signal est inversé (0V pour vrai / actif, 5V pour faux / inactif).

Sorties préselectionnées Réglage automatique des pins 2 à 9 selon le standard Sherline (Direction sur les pins 2, 4, 6, 8) ou selon le standard Xylotex (Direction sur les pins 3, 5, 7, 9).

Inclure une configuration de HAL personnalisée Vous permet d'ajouter une personnalisation de hal dans le fichier "custom.hal" après l'exécution de Stepconf.

Inclure un panneau PyVCP personnalisé Si coché, le panneau de contrôle PyVCP "`panel.xml`" sera affiché dans la partie droite de la fenêtre principale d'AXIS.

4.1.3 Configuration des axes

Nombre de pas par tour Nombre de pas entiers par tour de moteur. Si vous connaissez l'angle d'un pas en degrés (par exemple, 1.8 degrés), divisez 360 par cet angle pour obtenir le nombre de pas par tour du moteur.

Micropas du pilote Le nombre de micropas produits par le pilote. Entrez "2" pour le demipas.

Ratio des poulies Si votre machine a des poulies entre le moteur et la vis, entrez le ratio ici. Pour un entraînement direct, entrez "1 :1".

Pas de la vis Entrez ici le pas de la vis. Si vous avez choisi le "pouce" comme unité, entrez le nombre de filets par pouce (exemple, entrez 8 pour 8TPI). Si vous avez choisi le "mm", entrez la taille du filet en millimètres (exemple, entrez 2 pour un pas de 2mm). Si la machine se déplace dans la mauvaise direction, entrez une valeur négative au lieu d'une positive.

FIG. 4.2 – Page de réglage du port parallèle

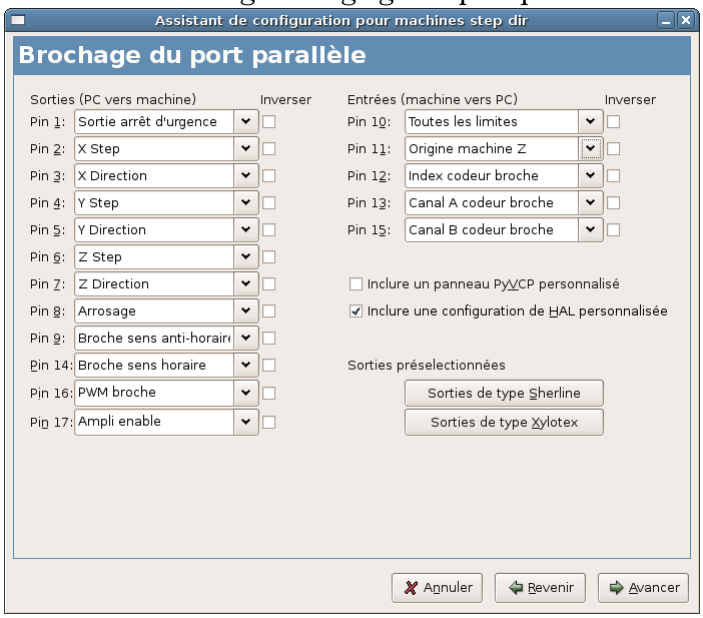
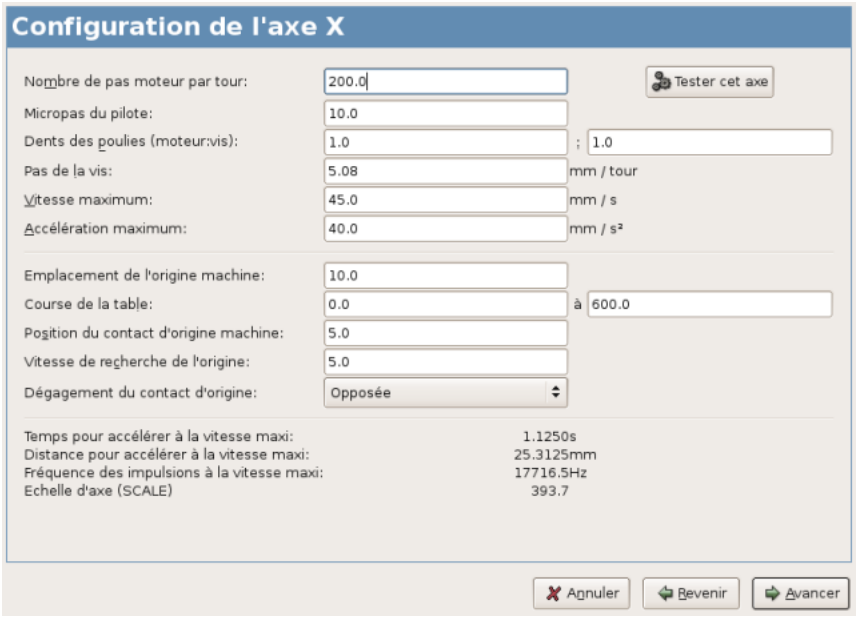


FIG. 4.3 – Page de configuration des axes



Vitesse maximum

Accélération maximum Les valeurs correctes pour ces deux entrées ne peuvent être déterminées que par l'expérimentation. Consultez 4.2.1 pour définir la vitesse et 4.2.2 pour définir l'accélération.

Emplacement de l'origine machine La position sur laquelle la machine se place après avoir terminé la procédure de prise d'origine de cet axe. Pour les machines sans contact placé au point d'origine, c'est la position à laquelle l'opérateur place la machine en manuel, avant de presser le bouton de **`POM des axes`**.

Course de la table Étendue de la course que le programme en gcode ne doit jamais dépasser. L'origine machine doit être située à l'intérieur de cette course. En particulier, avoir un point d'origine exactement égal à une de ces limites de course est une configuration incorrecte.

Position du contact d'origine machine Position à laquelle le contact d'origine machine est activé ou relâché pendant la procédure de prise d'origine machine. Ces entrées et les deux suivantes, n'apparaissent que si les contacts d'origine ont été sélectionnés dans le réglage des broches du port parallèle.

Vitesse de recherche de l'origine Vitesse utilisée pendant le déplacement vers le contact d'origine machine. Si le contact est proche d'une limite de déplacement de la table, cette vitesse doit être suffisamment basse pour permettre de décélérer et de s'arrêter avant d'atteindre la butée mécanique. Si le contact est fermé par la came sur une faible longueur de déplacement (au lieu d'être fermé depuis son point de fermeture jusqu'au bout de le course), cette vitesse doit être réglée pour permettre la décélération et l'arrêt, avant que le contact ne soit dépassé et ne s'ouvre à nouveau. La prise d'origine machine doit toujours commencer du même côté du contact.
Si la machine se déplace dans la mauvaise direction au début de la procédure de prise d'origine machine, rendez négative la valeur de **Vitesse de recherche de l'origine**.

Dégagement du contact d'origine Choisissez "Identique" pour que la machine reparte en arrière pour dégager le contact, puis revienne de nouveau vers lui à très petite vitesse. La seconde fois que le contact se ferme, la position de l'origine machine est acquise.
Choisissez "Opposition" pour que la machine reparte en arrière à très petite vitesse jusqu'au dégagement du contact. Quand le contact s'ouvre, la position de l'origine machine est acquise.

Temps pour accélérer à la vitesse maxi Temps calculé.

Distance pour accélérer à la vitesse maxi Distance calculée.

Fréquence des impulsions à la vitesse maxi Informations calculées sur la base des informations entrées précédemment. Il faut rechercher la plus haute **Fréquence des impulsions à la vitesse maxi** possible, elle détermine la période de base : **BASE_PERIOD**. Des valeurs supérieures à 20000Hz peuvent toutefois provoquer des ralentissements importants de l'ordinateur, voir même son bloquage (La plus grande fréquence utilisable variera d'un ordinateur à un autre)

4.1.4 Configuration de la broche

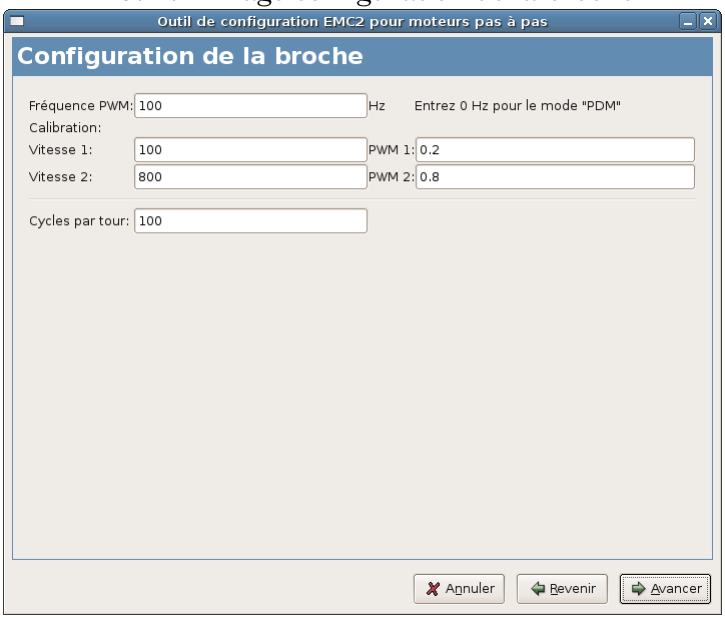
Ces options ne sont accessibles que quand "PWM broche", "Phase A codeur broche" ou " index broche" sont configurés sur le **brochage du port parallèle**.

4.1.4.1 Contrôle de la vitesse de broche

Si "PWM broche" apparaît dans le réglage du port parallèle, les informations suivantes doivent être renseignées :

Fréquence PWM La fréquence porteuse du signal PWM (modulation de largeur d'impulsions) du moteur de broche. Entrez " 0" pour le mode PDM (modulation de densité d'impulsions), qui est très utile pour générer une tension de consigne analogique. Reportez-vous à la documentation de votre variateur de broche pour la valeur appropriée.

FIG. 4.4 – Page configuration de la broche



Vitesse 1 et 2, PWM 1 et 2 Le fichier de configuration généré utilise une simple relation linéaire pour déterminer la valeur PWM correspondante à une vitesse de rotation. Si les valeurs ne sont pas connues, elles peuvent être déterminées. Voir la section "Ajuster la vitesse de broche" ci-dessous.

4.1.4.2 Mouvement avec broche synchronisée (filetage sur tour, taraudage rigide)

Lorsque les signaux appropriés, provenants d'un codeur de broche, sont connectés au port parallèle, EMC peut être utilisé pour le filetage avec broche synchronisée sur un tour. Ces signaux sont :

Index codeur broche également appelé "PPR broche", c'est une impulsion produite à chaque tour de broche.

Phase A codeur broche C'est une suite d'impulsions carrées générées sur la voie A du codeur pendant la rotation de la broche. La quantité d'impulsions pour un tour correspond à la résolution du codeur.

Phase B codeur broche (optionnel) C'est une seconde suite d'impulsions, générées sur la voie B du codeur et décalées par rapport à celle de la voie A. L'utilisation de ces deux signaux permet d'accroître l'immunité au bruit et la résolution.

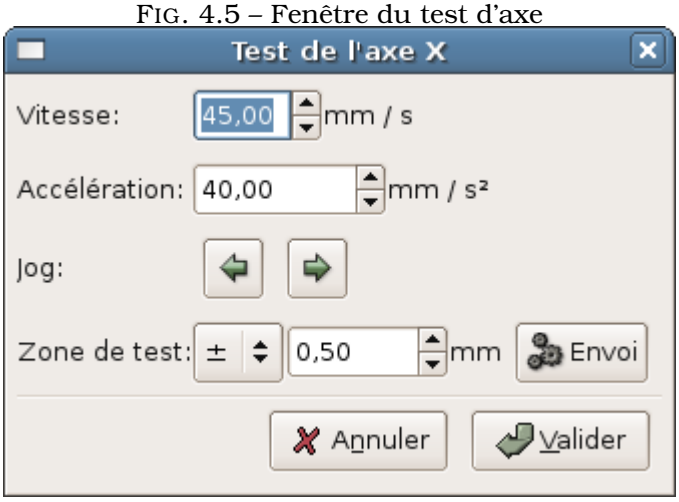
Si "Phase A codeur broche" et "Index broche" apparaissent dans le réglage des broches du port, l'information suivante doit être renseignée :

Cycles par tour Le nombre d'impulsions par tour sur la pin **Phase A codeur broche.**

4.1.5 Configuration machine complète

Cliquez "Appliquer" pour enregistrer les fichiers de configuration. Ensuite, vous pourrez relancer ce programme et ajuster les réglages entrés précédemment.

FIG. 4.5 – Fenêtre du test d'axe



4.2 Définir la vitesse et l'accélération

Avec Stepconf il est facile de définir certaines valeurs comme celles de l'accélération et de la vitesse. Entrez d'abord les éléments corrects pour les **Pas moteur**, les **Micropas**, les **Poulies**, et le **pas de vis**. Puis entrez une valeur aproximative de **Vitesse**. **Enfin,** cliquez sur **Tester cet axe**.

4.2.1 Définir la vitesse maximum

Commencez avec une faible valeur d'accélération (par exemple, 2 in/s^2 ou 50mm/s^2) et la vitesse que vous espérez atteindre. À l'aide des boutons de jog, positionnez l'axe vers son centre. Soyez prudent, car avec peu d'accélération, l'axe peut prendre une surprenante longueur pour ralentir et s'arrêter.

Après avoir mesuré les longueurs de déplacement disponibles dans chaque direction, entrez la moyenne de ces distances dans la zone test, en gardant à l'esprit que, après un décrochage le moteur peut repartir dans la direction inattendue. Cliquez sur **`Envoi`**. La machine va commencer à faire des allers et retours sur cet axe. Dans cet essai, il est important que la combinaison entre l'accélération et la zone de test, permette à la machine d'atteindre la vitesse sélectionnée et de s'y déplacer sur, au moins, une courte distance (plus cette distance sera longue, meilleur sera le test). La formule : distance=0.5*vitesse*vitesse/accélération donne la distance minimum requise avec les vitesse et accélération spécifiées. Il est souhaitable de pousser sur la table dans la direction inverse du mouvement pour simuler les efforts de coupe. Si la machine cale, réduire la vitesse et relancer le test.

Si la machine ne présente aucun décrochage, cliquez sur le bouton **`Exécuter`**. L'axe revient alors à sa position de départ. Si cette position est incorrecte, c'est que l'axe a calé ou a perdu des pas au cours de l'essai. Réduire la vitesse et relancer le test. Si la machine ne se déplace pas, cale, ou perd des pas même à faible vitesse, vérifiez les éléments suivants :

- Timings des impulsions de pas corrects
- Brochage des sorties correct, inclus les cases **`Inverser`** des pins du port
- Câbles blindés et correctement raccordés
- Problème physique avec le moteur, le couplage du moteur, l'écrou de la vis, point dur des glissières, etc.

Une fois que vous avez trouvé une vitesse à laquelle l'axe ne perd plus de pas et à laquelle les mesures sont exactes pendant le test, réduisez la de 10% et utilisez la comme vitesse maximum pour cet axe.

4.2.2 Définir l'accélération maximum

Avec la vitesse maximale que vous avez trouvé à l'étape précédente, entrez une valeur d'accélération approximative. Procédez comme pour la vitesse, en ajustant la valeur d'accélération en plus ou en moins selon le résultat. Dans cet essai, il est important que la combinaison de l'accélération et de la zone test permette à la machine d'atteindre la vitesse sélectionnée. Une fois que vous avez trouvé une valeur à laquelle l'axe ne perd plus de pas pendant le test, la réduire de 10% et l'utiliser comme accélération maximale pour cet axe.

4.3 Définir la calibration de la broche

Entrez les valeurs suivantes dans la page de configuration de la broche :

Vitesse 1 :	**0**	PWM 1 :	**0**
Vitesse 2 :	**1000**	PWM 1 :	**1**

Finissez les étapes suivantes de la configuration, puis lancez EMC2 avec votre configuration. Mettez la machine en marche et allez dans l'onglet **`Données manuelles`**. démarrez le moteur de broche en entrant : **`M3 S100`**. Modifiez la vitesse de broche avec différentes valeur comme : **`S800`**. Les valeurs permises vont de 1 à 1000.

Pour deux différentes valeurs de **`S`**xxx, mesurez la vitesse de rotation réelle de la broche en tours/mn. Enregistrez ces vitesses réelles de la broche. Relancez Stepconf. Pour les **`Vitesses`**, entrez les valeurs réelles mesurées et pour les **`PWM`**, entrez la valeur **`S`**xxx divisée par 1000.

Parce que la plupart des interfaces ne sont pas linéaires dans leurs courbes de réponse, il est préférable de :

- S'assurer que les deux points de mesure des vitesses en tr/mn ne soient pas trop rapprochés
- S'assurer que les deux vitesses utilisées sont dans la gamme des vitesses utilisées généralement par la machine.

Par exemple, si votre broche tourne entre 0tr/mn et 8000tr/mn, mais que vous l'utilisez généralement entre 400tr/mn et 4000tr/mn, prenez alors des valeurs qui donneront 1600tr/mn et 2800tr/mn.

4.4 Fin de course des axes, position de l'origine machine et emplacements des contacts d'origine machine

La course de chaque axe est bien délimitée. Les extrémités physiques d'une course sont appelées **les butées mécanique**s.

Avant la **butée mécanique** se trouve un **contact de fin de course**. Si ce contact est rencontré pendant les opérations normales, EMC coupe la puissance du moteur. La distance entre **le fin de course** et **la butée mécanique** doit être suffisante pour permettre au moteur, dont la puissance a été coupée, de s'arrêter malgré l'inertie du mobile.

Avant le **contact de fin de course** se trouve une **limite soft**. Cette limite logicielle est introduite après la prise d'origine machine. Si une commande manuelle ou un programme gcode dépasse cette limite, ils ne seront pas éxécutés. Si un mouvement en jog cherche à dépasser la limite logicielle, il sera interrompu sur cette limite.

Le **contact d'origine machine** peut être positionné n'importe où, le long d'une course (entre les butées mécaniques).

Si aucun mécanisme externe ne désactive la puissance moteur quand un contact de limite est enfoncé, un des contacts de fin de course peut être utilisé comme contact d'origine machine.

La **position zéro** est la position correspondante au 0 de l'axe dans le système de coordonnées machine. Habituellement la **position zéro** doit se trouver entre les deux **limites soft**. Sur les tours, le mode vitesse de surface constante requiert que la coordonnée **X=0** corresponde au centre de rotation de la broche quand aucun correcteur d'outil n'est actif.

La **position de l'origine** est la position, située le long de l'axe, sur laquelle le mobile sera déplacé à la fin de la séquence de prise d'origine. Cette position doit se situer entre les **limites soft**. En particulier, la **position de l'origine** ne doit jamais être égale à une **limite soft**.

4.4.1 Travail sans contact de fin de course

Une machine peut être utilisée sans contact de fin de course. Dans ce cas, seules les **limites soft** empêcheront la machine d'atteindre les **butées mécaniques**. Les **limites soft** n'opérent qu'après que la POM soit faite sur la machine. Puisqu'il n'y a pas de contact, la machine doit être déplacée à la main et à l'oeil, à sa position d'origine machine avant de presser le bouton **`POM des axes`** ou le sous-menu **`Machine->Prises d'origines machine->POM de l'axe`**.

4.4.2 Travail sans contact d'origine machine

Un machine peut être utilisée sans contact d'origine machine. Si la machine dispose de contacts de fin de courses, mais pas de contact d'origine machine, il est préférable d'utiliser le contact de fin de course comme contact d'origine machine (exemple, choisissez **Limite mini + origine X** dans le réglage du port). Si la machine ne dispose d'aucun contact, ou que le contact de fin de course n'est pas utilisable pour une autre raison, alors la prise d'origine machine peut être réalisée à la main. Faire la prise d'origine à la main n'est certes pas aussi reproductible que sur des contacts, mais elle permet tout de même aux **limites soft** d'être utilisables.

4.5 Test de latence

Faire générer les impulsions de pas au logiciel présente un grand avantage, c'est gratuit. Quasiment chaque PC dispose d'un port parallèle capable de sortir sur ses broches les signaux de pas générés par le logiciel. Cependant, les générateurs d'impulsions logiciels ont aussi quelques désavantages :

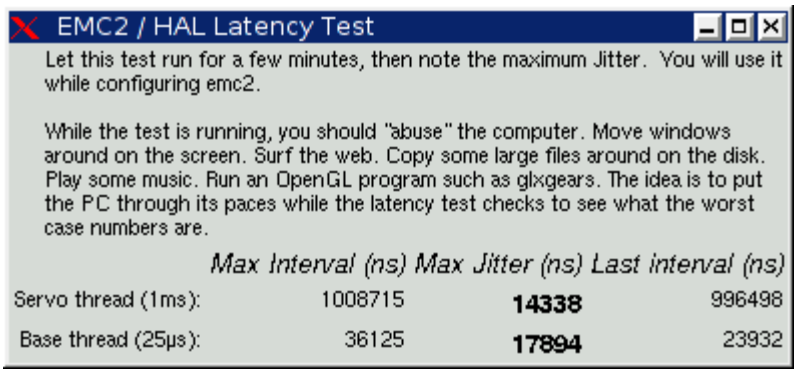
– Leur fréquence maximum est limitée
– Les trains d'impulsions générés sont irréguliers
– Ils chargent le processeur

Le temps de latence est le temps nécessaire au PC pour arrêter ce qu'il est en train de faire pour répondre à une requête externe. Dans notre cas, la requête est le "battement de coeur" périodique qui sert de référence pour les impulsions de pas. Plus la latence est basse, plus le coeur pourra battre vite et donc, plus rapides et plus douces seront les impulsions de pas.

Le temps de latence est beaucoup plus important que la vitesse du µP. Un vieux Pentium II qui répond aux interruptions avec 10 microsecondes entre chacune peut donner de meilleurs résultats qu'un rapide P4 en Hyperthreading.

Le CPU n'est pas le seul facteur déterminant le temps de latence. Les cartes mères, les cartes vidéo, les ports USB et de nombreuses autres choses peuvent détériorer le temps de latence. La meilleure façon de savoir ce qu'il en est sur votre PC est de lancer un **`latency test`** de HAL.

Pour exécuter le test, il suffit d'ouvrir une console et de taper : **`latency-test`** . Vous devriez voir quelque chose comme ceci :



Pendant que le test est en cours d'exécution, il faut "abuser" de l'ordinateur. Déplacez les fenêtres sur l'écran. Connectez vous à l'Internet. Copiez quelques gros fichiers sur le disque dur. Jouer de la musique. Lancez une démo OpenGL telle que **glxgears**. L'idée est de charger le PC au maximum pour que le temps de latence soit mesuré dans le pire des cas. **Ne pas exécuter EMC2 ou Stepconf pendant que latency-test est en cours d'exécution.**

Le chiffre **max jitter** dans cet exemple est de 17894 nanosecondes, soit 17.9 microsecondes. Enregistrer ce chiffre et entrez le dans Stepconf quand il le demande.

Dans cet exemple de test de latence il n'a fallu que quelques secondes pour afficher cette valeur. Vous devrez peut être lancer le test pendant plusieurs minutes. Parfois même, dans le pire des cas, rien ne provoque de latence ou seulement des actions particulières. Par exemple, une carte mère Intel marchait très bien la plupart du temps, mais toutes les 64 secondes elle avait une très mauvaise latence de 300µs. Heureusement, il existe un correctif (voir "Fixing Dapper SMI Issues http://wiki.linuxcnc.org/cgi-bin/emcinfo.pl?FixingDapperSMIIssues)

Alors, que signifient les résultats ? Si le résultat de votre Max Jitter est en dessous d'environ 15-20 microsecondes (15000-20000 nanosecondes), l'ordinateur pourra donner d'excellents résultats avec la génération logicielle des pas. Si le temps de latence est à plus de 30-50 microsecondes, vous aurez de bons résultats, mais la vitesse maximum sera un peu décevante, spécialement si vous utilisez des micropas ou si le pas de votre vis est fin. Si les résultats sont de 100uS ou plus (100,000 nanosecondes), alors le PC n'est pas un bon candidat à la génération des pas. Les résultats supérieurs à 1 milliseconde (1,000,000 nanosecondes) éliminent, dans tous les cas, ce PC pour faire tourner EMC, en utilisant des micropas ou pas.

Notez que si vous obtenez une latence élevée, il peut être possible de l'améliorer. Un PC avait une très mauvaise latence (plusieurs millisecondes) en utilisant la carte graphique interne. Un carte graphique Matrox d'occasion à $5US a résolu le problème. EMC n'exige pas de matériel de pointe.

4.6 Câblage des contacts de fin de course et d'origine machine

Le câblage idéal des contacts externes serait une entrée par contact. Toutefois, un seul port parallèle d'ordinateur offre un total de 5 entrées, alors qu'il n'y a pas moins de 9 contacts sur une machine 3 axes. Au lieu de cela, plusieurs contacts seront câblés ensemble de diverses façons afin de nécessiter un plus petit nombre d'entrées.

Les figures ci-dessous montrent l'idée générale du câblage de plusieurs contacts à une seule broche d'entrée. Dans chaque cas, lorsqu'un contact est actionné, la valeur vue sur l'entrée va passer d'une logique **haute** à une logique **basse**. Cependant, EMC s'attend à une valeur **VRAIE** quand un contact est fermé, de sorte que les cases **Inverser** correspondantes devront être cochées sur la page de réglage du port parallèle.

Les combinaisons suivantes sont permises dans Stepconf :

- Les contacts d'origine machine de tous les axes combinés
- Les contacts de fin de course de tous les axes combinés

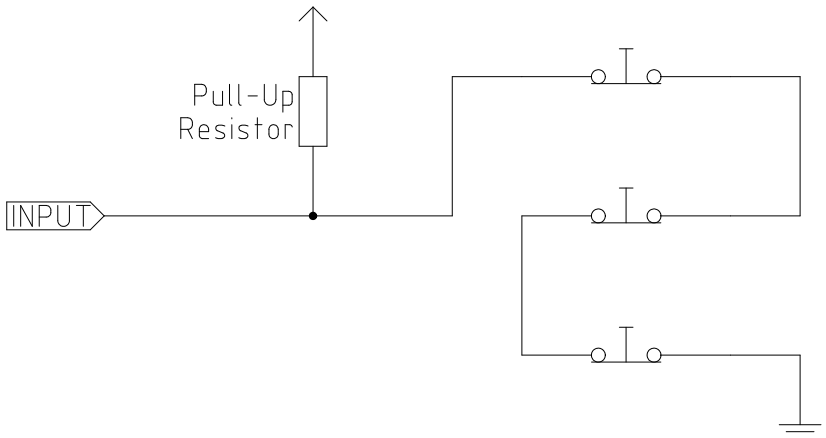


FIG. 4.6 – Câblage de contacts NC en série (schéma simplifié)

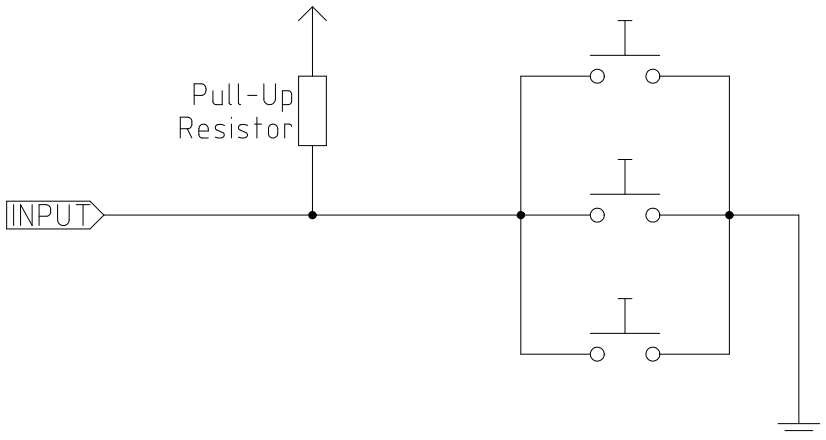


FIG. 4.7 – Câblage de contacts NO en parallèle (schéma simplifié)

– Les contacts de fin de course d'un seul axe combinés
– Les contacts de fin de course et le contact d'origine machine d'un seul axe combinés
– Un seul contact de fin de course et le contact d'origine machine d'un seul axe combinés

Les deux dernières combinaisons sont également appropriées quand le type **`contact + origine`** est utilisé.

Chapitre 5

Configuration, fichier ini

5.1 Fichiers utilisés pour la configuration

EMC est entièrement configuré avec des fichiers textes classiques. Tous ces fichiers peuvent être lus et modifiés dans n'importe quel éditeur de texte disponible dans toute distribution Linux[1] . Soyez prudent lorsque vous modifierez ces fichiers, certaines erreurs pourraient empêcher le démarrage d'EMC. Ces fichiers sont lus à chaque fois que le logiciel démarre. Certains d'entre eux sont lus de nombreuses fois pendant l'exécution d'CNC.

Les fichiers de configuration inclus :

INI Le fichier ini écrase les valeurs par défaut compilées dans le code d'EMC. Il contient également des sections qui sont lues directement par le HAL (Hardware Abstraction Layer, couche d'abstraction matérielle).

HAL Les fichiers hal installent les modules de process, ils créent les liens entre les signaux d'EMC et les broches spécifiques du matériel.

VAR Ce fichier contient une suite de numéros de variables. Ces variables contiennent les paramètres qui seront utilisés par l'interpréteur. Ces valeurs sont enregistrées d'une exécution à l'autre.

TBL Ce fichier contient les informations relatives aux outils.

NML Ce fichier configure les canaux de communication utilisés par EMC. Il est normalement réglé pour lancer toutes les communications avec un seul ordinateur, peut être modifié pour communiquer entre plusieurs ordinateurs.

.emcrc Ce fichier enregistre des informations spécifiques à l'utilisateur, il a été créé pour enregistrer le nom du répertoire lorsque l'utilisateur choisit sa première configuration d'EMC.[2]

Les éléments avec le repère **(HAL)** sont utilisés seulement pour les fichiers de HAL en exemples. C'est une bonne convention. D'autres éléments sont utilisés directement par EMC et doivent toujours avoir la section et le nom donné à l'item.

5.2 Organisation du fichier INI

Un fichier INI typique suit une organisation simple ;

[1] Ne confondez pas un éditeur de texte et un traitement de texte. Un éditeur de texte comme gedit ou kwrite produisent des fichiers uniquement en texte. Les lignes de textes sont séparées les unes des autres. Un traitement de texte comme Open Office produit des fichiers avec des paragraphes, des mises en formes des mots. Ils ajoutent des codes de contrôles, des polices de formes et de tailles variées etc. Un éditeur de texte n'a rien de tout cela.

[2] Habituellement, ce fichier est dans le répertoire home de l'utilisateur (ex : /home/user/)

– commentaires.
– sections,
– variables.

Chacun de ces éléments est séparé, sur une seule ligne. Chaque fin de ligne ou retour chariot crée un nouvel élément.

5.2.1 Commentaires

Une ligne de commentaires débute avec un ; ou un #. Si le logiciel qui analyse le fichier ini rencontre l'un ou l'autre de ces caractères, le reste de la ligne est ignorée. Les commentaires peuvent être utilisés pour décrire ce que font les éléments du fichier INI.

```
; Ceci est le fichier de configuration de ma petite fraiseuse.
; Je l'ai ajusté le 12 janvier 2006
```

Des commentaires peuvent également être utilisés pour choisir entre plusieurs valeurs d'une seule variable.

```
# DISPLAY = tkemc
DISPLAY = axis
# DISPLAY = mini
# DISPLAY = keystick
```

Dans cette liste, la variable DISPLAY est positionnée sur axis puisque toutes les autres sont commentées. Si quelqu'un édite une liste comme celle-ci et par erreur, décommente deux lignes, c'est la première rencontrée qui sera utilisée.

Notez que dans une ligne de variables, les caractères "**#**" et "**;**" n'indiquent pas un commentaire.

```
INCORRECT = value     # and a comment
```

5.2.2 Sections

Les différentes parties d'un fichier .ini sont regroupées dans des sections. Une section commence par son nom en majuscules entre crochets [UNE_SECTION]. L'ordre des sections est sans importance. Les sections suivantes sont utilisées par emc :

– [EMC] informations générales (5.3.1)
– [DISPLAY] sélection du type d'interface graphique (5.3.2)
– [RS274NGC] ajustements utilisés par l'interpréteur de g-code
– [EMCMOT] Réglages utilisés par le contrôleur de mouvements temps réel (5.3.3)
– [HAL] spécifications des fichiers .hal (5.3.5)
– [TASK] Réglages utilisés par le contrôleur de tâche (5.3.4)
– [TRAJ] Réglages additionnels utilisés par le contrôleur de mouvements temps réel (5.3.6)
– [AXIS_0] ... [AXIS_n] Groupes de variables pour AXIS (5.3.7)
– [EMCIO] Réglages utilisés par le contrôleur d'entrées/sorties (5.3.8)

5.2.3 Variables

Une ligne de variables est composée d'un nom de variable, du signe égal (=) et d'une valeur. Tout, du premier caractère non blanc qui suit le signe = jusqu'à la fin de la ligne, est passé comme valeur à la variable. Vous pouvez donc intercaler des espaces entre les symboles si besoin. Un nom de variable est souvent appelé un mot clé.

Les paragraphes suivants détaillent chaque section du fichier de configuration, en utilisant des exemples de variables dans les lignes de configuration.

Certaines de ces variables sont utilisées par EMC. Elles doivent toujours utiliser le nom de section et le nom de variable dans leur appellation. D'autres variables ne sont utilisées que par HAL. Les noms des sections et les noms des variables indiquées sont celles qui sont utilisées dans les exemples de fichiers de configuration.

5.3 Définition des variables du fichier INI

5.3.1 Section [EMC]

VERSION = $Revision : 1.3 $ Le numéro de version du fichier INI. La valeur indiquée ici semble étrange, car elle est automatiquement mise à jour lors de l'utilisation du système de contrôle de révision. C'est une bonne idée de changer ce numéro à chaque fois que vous modifiez votre fichier. Si vous voulez le modifier manuellement, il suffit de changer le numéro sans toucher au reste.

MACHINE = ma machine C'est le nom du contrôleur, qui est imprimé dans le haut de la plupart des fenêtres. Vous pouvez insérer ce que vous voulez ici tant que ça reste sur une seule ligne.

RS274NGC_STARTUP_CODE = G21 G90 Une chaine de codes NC qui sera utilisée pour initialiser l'interpréteur. Elle ne se substitue pas à la spécification des gcodes modaux du début de chaque fichier ngc. Les codes modaux des machines diffèrent, ils pourraient être modifiés par les gcodes interprétés plus tôt dans la session.

5.3.2 Section [DISPLAY]

Les différentes interfaces du programme utilisent différentes options. Toutes les options ne sont pas supportées par toutes les interfaces.

DISPLAY = tkemc Le nom de l'interface utilisateur à utiliser. Les options disponibles sont les suivantes :

- axis
- keystick
- mini
- tkemc
- xemc

POSITION_OFFSET = RELATIVE Le système de coordonnées (RELATIVE ou MACHINE) à utiliser au démarrage de l'interface utilisateur. Le système de coordonnées RELATIVE reflète le G92 et le décalage d'origine G5x actuellement actifs.

POSITION_FEEDBACK = ACTUAL Valeur de la position (COMMANDED ou ACTUAL) à afficher au démarrage de l'interface utilisateur. La position COMMANDED est la position exacte requise par emc. La position ACTUAL est la position retournée par l'électronique des moteurs.

MAX_FEED_OVERRIDE = 1.2 La correction de vitesse maximum que l'opérateur peut utiliser. 1.2 signifie 120% de la vitesse programmée.

MIN_SPINDLE_OVERRIDE = 0.5 Correction de vitesse minimum de broche que l'opérateur pourra utiliser. 0.5 signifie 50% de la vitesse de broche programmée. (utile si il est dangereux de démarrer un programme avec une vitesse de broche trop basse).

MAX_SPINDLE_OVERRIDE = 1.0 Correction de vitesse maximum de broche que l'opérateur pourra utiliser. 1.0 signifie 100% de la vitesse de broche programmée.

DEFAULT_LINEAR_VELOCITY = .25 Vitesse minimum par défaut pour les jogs linéaires, en unités machine par seconde. Seulement utilisé dans l'interface AXIS.

MAX_LINEAR_VELOCITY = 1.0 Vitesse maximum par défaut pour les jogs linéaires, en unités machine par seconde. Seulement utilisé dans l'interface AXIS.

DEFAULT_ANGULAR_VELOCITY = .25 Vitesse minimum par défaut pour les jogs angulaires, en unités machine par seconde. Seulement utilisé dans l'interface AXIS.

MAX_ANGULAR_VELOCITY = 1.0 Vitesse maximum par défaut pour les jogs angulaires, en unités machine par seconde. Seulement utilisé dans l'interface AXIS.

PROGRAM_PREFIX = ~/emc2/nc_files Répertoire par défaut des fichiers de g-codes et emplacement des M-codes définis par l'utilisateur.

INCREMENTS = 1 mm, .5 mm, . . . Définit les incréments disponibles pour le jog incremental. Voir la section 5.3.2 pour plus d'informations. Seulement utilisé dans l'interface AXIS.

INTRO_GRAPHIC = emc2.gif L'image affichée sur l'écran d'accueil.

INTRO_TIME = 5 Durée d'affichage de l'écran d'accueil.

OPEN_FILE = /full/path/to/file.ngc Le fichier NC à utiliser au démarrage d'AXIS.

5.3.3 Section [EMCMOT]

BASE_PERIOD = 50000 (HAL) "Période de base" des tâches, exprimée en nanosecondes. C'est la plus rapide des horloges de la machine.
Avec un système à servomoteurs, il n'y a généralement pas de raison pour que **BASE_PERIOD** soit plus petite que **SERVO_PERIOD**.
Sur une machine de type "step&direction" avec génération logicielle des impulsions de pas, c'est **BASE_PERIOD** qui détermine le nombre maximum de pas par seconde. Si de longues impulsions de pas ou de longs espaces entre les impulsions ne sont pas requis par l'électronique, la fréquence maximum absolue est de un pas par **BASE_PERIOD**. Ainsi, la **BASE_PERIOD** utilisée ici donnera une fréquence de pas maximum absolue de 20000 pas par seconde. 50000ns est une valeur assez large. La plus petite valeur utilisable est liée au résultat du test de latence (4.5), à la longueur des impulsions de pas nécessaire et à la vitesse du µP.
Choisir une BASE_PERIOD trop basse peut amener à des messages "Unexpected realtime delay", des bloquages ou des reboots spontanés.

SERVO_PERIOD = 1000000 (HAL) Période de la tâche "Servo", exprimée également en nanosecondes. Cette valeur sera arrondie à un multiple entier de **BASE_PERIOD**. Elle est utilisée aussi sur des systèmes basés sur des moteurs pas à pas
C'est la vitesse avec laquelle la nouvelle position des moteurs est traitée, les erreurs de suivi vérifiées, les valeurs des sorties PID sont rafraichies etc.
Sur la plupart des systèmes **cette** valeur n'est pas à modifier. Il s'agit du taux de mise à jour du planificateur de mouvement de bas niveau.

TRAJ_PERIOD = 1000000 (HAL) Période du planificateur de trajectoire, exprimée en nanosecondes. Cette valeur sera arrondie à un multiple entier de **SERVO_PERIOD**.
Excepté pour les machines avec une cinématique particulière (ex : hexapodes) Il n'y a aucune raison de rendre cette valeur supérieure à **SERVO_PERIOD**.

5.3.4 Section [TASK]

CYCLE_TIME = 0.001 Période exprimée en secondes, à laquelle EMCTASK va tourner. Ce paramètre affecte l'intervalle de polling lors de l'attente de la fin d'un mouvement, lors de l'exécution d'une pause d'instruction et quand une commande provenant d'une interface utilisateur est acceptée. Il n'est généralement pas nécessaire de modifier cette valeur.

5.3.5 Section [HAL]

HALFILE = example.hal Exécute le fichier 'example.hal' au démarrage. Si **HALFILE** est spécifié plusieurs fois, les fichiers sont exécutés dans l'ordre de leur apparition dans le fichier ini.

Presque toutes les configurations auront au moins un **HALFILE**. Les systèmes à moteurs pas à pas ont généralement deux de ces fichiers, un qui spécifie la configuration générale des moteurs (`core_stepper.hal`) et un qui spécifie le brochage des sorties (`xxx_pinout.hal`)

HAL = command Exécute 'command' comme étant une simple commande hal. Si **HAL** est spécifié plusieurs fois, les commandes sont exécutées dans l'ordre où elles apparaissent dans le fichier ini. Les lignes **HAL** sont exécutées après toutes les lignes **HALFILE**.

SHUTDOWN = shutdown.hal Exécute le fichier 'shutdown.hal' quand emc s'arrête. Selon les pilotes de matériel utilisés, il est ainsi possible de positionner les sorties sur des valeurs définies quand emc s'arrête normalement. Cependant, parce qu'il n'y a aucune garantie que ce fichier sera exécuté (par exemple, dans le cas d'une panne de l'ordinateur), il ne remplace pas une véritable chaîne physique d'arrêt d'urgence ou d'autres logiciels de protection des défauts de fonctionnement.

POSTGUI_HALFILE = example2.hal *(Seulement avec l'interface AXIS)* Exécute 'example2.hal' après que l'interface graphique ait créé ses HAL pins.

5.3.6 Section [TRAJ]

La section [TRAJ] contient les paramètres généraux du module planificateur de trajectoires d' EMC-MOT. Vous n'aurez pas à modifier ces valeurs si vous utilisez EMC avec une machine à trois axes en provenance des USA. Si vous êtes dans une zone métrique, utilisant des éléments matériels métriques, vous pourrez utiliser le fichier stepper_mm.ini dans lequel les valeurs sont déjà configurées dans cette unité.

COORDINATES = X Y Z Les noms des axes à contrôler. X, Y, Z, A, B, C, U, V, et W sont valides. Seuls les axes nommés dans **COORDINATES** seront acceptés dans le g-code. Cela n'a aucun effet sur l'ordonnancement des noms d'axes depuis le G-code (X- Y- Z-) jusqu'aux numéros d'articulations. Pour une "cinématique triviale", X est toujours l'articulation 0, A est toujours l'articulation 4, U est toujours l'articulation 7 et ainsi de suite. Il est permis d'écrire les noms d'axe par paire (ex : X Y Y Z pour une machine à portique) mais cela n'a aucun effet.

AXES = 3 Une unité de plus que le plus grand numéro d'articulation du système. Pour une machine XYZ, les articulations sont numérotées 0, 1 et 2. Dans ce cas, les AXES sont 3. Pour un système XYUV utilisant une "cinématique triviale", l'articulation V est numérotée 7 et donc les AXES devraient être 8. Pour une machine à cinématique non triviale (ex : scarakins) ce sera généralement le nombre d'articulations contrôlées.

HOME = 0 0 0 Coordonnées de l'origine machine de chaque axe. De nouveau, pour une machine 4 axes, vous devrez avoir 0 0 0 0. Cette valeur est utilisée uniquement pour les machines à cinématique non triviale. Sur les machines avec cinématique triviale, cette valeur est ignorée.

LINEAR_UNITS=<units> Le nom des unités utilisées dans le fichier INI. Les choix possibles sont 'in', 'inch', 'imperial', 'metric', 'mm'.
Cela n'affecte pas les unités linéaires du code NC (pour cela il y a les mots G20 et G21).

ANGULAR_UNITS=<units> Le nom des unités utilisées dans le fichier INI. Les choix possibles sont 'deg', 'degree' (360 pour un cercle), 'rad', 'radian' (2pi pour un cercle), 'grad', ou 'gon' (400 pour un cercle).
Cela n'affecte pas les unités angulaires du code NC. Dans le code RS274NGC, les mots A-, B- et C- sont toujours exprimés en degrés.

DEFAULT_VELOCITY = 0.0167 La vitesse initiale de jog des axes linéaires, en unités par seconde. La valeur indiquée ici correspond à une unité par minute.

DEFAULT_ACCELERATION = 2.0 Dans les machines à cinématique non triviale, l'accélération utilisée pour "teleop" jog (espace cartésien), en unités machine par seconde par seconde.

MAX_VELOCITY = 5.0 Vitesse maximale de déplacement pour les axes, exprimée en unités machine par seconde. La valeur indiquée est égale à 300 unités par minute.

MAX_ACCELERATION = 20.0 Accélération maximale pour les axes, exprimée en unités machine par seconde par seconde.

POSITION_FILE = position.txt Si réglée à une valeur non vide, les positions des articulations sont enregistrées dans ce fichier. Cela permet donc de redémarrer avec les mêmes coordonnées que lors de l'arrêt[3] . Si vide, les positions ne seront pas enregistrées et commenceront à 0 à chaque fois qu'EMC démarrera.

5.3.7 Section [AXIS_<num>]

Les sections [AXIS_0], [AXIS_1], etc. contiennent les paramètres généraux des composants individuels du module de contrôle. La numérotation des sections axis commencent à 0 et vont jusqu'au nombre d'axes spécifié dans la variable [TRAJ] AXES, moins 1.

TYPE = LINEAR Type des axes, soit LINEAR, soit ANGULAR.

UNITS = inch Ce réglage écrase celui des variables [TRAJ] UNITS si il est spécifié. (ex : [TRAJ]LINEAR_UNITS si le TYPE de cet axe est LINEAR, [TRAJ]ANGULAR_UNITS si le TYPE de cet axe est ANGULAR)

MAX_VELOCITY = 1.2 Vitesse maximum pour cet axe en unités machine par seconde.

MAX_ACCELERATION = 20.0 Accélération maximum pour cet axe en unités machine par seconde au carré.

BACKLASH = 0.000 Valeur de compensation du jeu en unités machine. Peut être utilisée pour atténuer de petites déficiences du matériel utilisé pour piloter cet axe.

COMP_FILE = file.extension Fichier dans lequel est enregistrée une structure de compensation spécifique à cet axe. Les valeurs internes sont des triplets représentant les positions suivantes :

1. Positions nominales
2. Positions en marche positive
3. Positions en marche négative.

La position nominale est celle où devrait être le mobile. La position en marche positive signifie, où se trouve le mobile pendant le déplacement dans le sens positif. La position en marche négative signifie, où se trouve le mobile pendant le déplacement dans le sens négatif. Un triplet par ligne. Actuellement la limite d'EMC2 est de 256 triplets par axe. Si COMP_FILE est spécifié, BACKLASH est ignoré. Les valeurs sont en unités machine.

COMP_FILE_TYPE = 1 En spécifiant une valeur non nulle, le format des triplets du fichier COMP_FILE sera différent. Pour COMP_FILE_TYPE = 0, les valeurs des triplets seront : position nominale, position en marche positive, position en marche négative. Pour COMP_FILE_TYPE différent de 0, les valeurs dans COMP_FILE seront : position nominale, écart sens positif, écart sens négatif. Comparées au valeurs définies au dessus elles correspondent à, nominale, nominale-position en marche positive, nominal-position en marche négative.

Exemple de triplet avec COMP_FILE_TYPE = 0 : 1.00 1.01 0.99.

Le même exemple de triplet avec COMP_FILE_TYPE = 1 : 1.00 -0.01 0.01

MIN_LIMIT = -1000 Limite minimum des mouvements de cet axe (limite soft), en unités machine. Quand cette limite tend à être dépassée, le contrôleur arrête le mouvement.

MAX_LIMIT = 1000 Limite maximum des mouvements de cet axe (limite soft), en unités machine. Quand cette limite tend à être dépassée, le contrôleur arrête le mouvement.

MIN_FERROR = 0.010 Valeur indiquant, en unités machine, de combien le mobile peut dévier à très petite vitesse de la position commandée. Si MIN_FERROR est plus petit que FERROR, les deux produisent une rampe de points de dérive. Vous pouvez imaginer un graphe sur lequel une dimension représente la vitesse et l'autre, l'erreur tolérée. Quand la vitesse augmente, la quantité d'erreurs de suivi augmente également et tend vers la valeur FERROR.

[3]Cela suppose, que hors puissance, la machine ne fera aucun mouvement pendant tout son arrêt. C'est utile pour les petites machines sans contact d'origine machine.

FERROR = 1.0 FERROR est le maximum d'erreurs de suivi tolérable, en unités machine. Si la différence entre la position commandée et la position retournée excède cette valeur, le contrôleur désactive les calculs des servomoteurs, positionne toutes les sorties à 0.0 et coupe les amplis des moteurs. Si MIN_FERROR est présent dans le fichier .ini, une vitesse proportionnelle aux erreurs de suivi est utilisée. Ici, le maximum d'erreur de suivi est proportionnel à la vitesse, quand FERROR est appliqué à la vitesse rapide définie dans [TRAJ]MAX_VELOCITY et proportionnel aux erreurs de suivi pour les petites vitesses. L'erreur maximale admissible sera toujours supérieure à MIN_FERROR. Cela permet d'éviter que de petites erreurs de suivi sur les axes stationnaires arrêtent les mouvements de manière impromptue. Des petites erreurs de suivi seront toujours présentes à cause des vibrations, etc. La polarité des valeurs de suivi détermine comment les entrées sont interprétées et comment les résultats sont appliqués aux sorties. Elles peuvent généralement être réglées par tâtonnement car il n'y a que deux possibilités. Le programme utilitaire USRMOT peut être utilisé pour les ajuster intéractivement et vérifier les résultats, de sorte que les valeurs puissent être mises dans le fichier INI avec un minimum de difficultés.

5.3.7.1 Variables relatives aux prises d'origine

Les cinq prochains paramètres sont relatifs aux prises d'origine.

HOME_OFFSET = 0.0 Position du contact d'origine machine de l'axe ou impulsion d'index, en unités machine.

HOME_SEARCH_VEL = 0.0 Vitesse du mouvement inital de prise d'origine, en unités machine par seconde. Une valeur de zéro suppose que la position courante est l'origine machine. Si votre machine n'a pas de contact d'origine, laissez cette valeur à zéro.

HOME_LATCH_VEL = 0.0 Vitesse du mouvement final de prise d'origine pour le dégagement du contact d'origine, en unités machine par seconde.

HOME_USE_INDEX = NO Si l'encodeur utilisé pour cet axe fournit une impulsion d'index et qu'elle est gérée par la carte contrôleur, vous pouvez mettre sur Yes. Quand il est sur yes, il aura une incidence sur le type de séquence de prise d'origine utilisé.

HOME_IGNORE_LIMITS = NO Certaines machines utilisent une limite d'axe comme contact d'origine machine. Cette variable devra être positionnée sur yes si c'est le cas de votre machine.

5.3.7.2 Variables relatives aux servo

Les entrées suivantes concernent les systèmes à servomoteurs, comme la carte du système univstep de Pico Systems.[4] Cette description suppose que les unités en sortie du composant PID sont des Volts.

P = 50 (HAL) La composante **p**roportionnelle du gain de l'ampli moteur de cet axe. Cette valeur multiplie l'erreur entre la position commandée et la position actuelle en unités machine, elle entre dans le calcul de la tension appliquée à l'ampli moteur. Les unités du gain **P** sont des Volts sur des unités machine, exemple : $\frac{volt}{mm}$ si l'unité machine est le millimètre.

I = 0 (HAL) La composante **i**ntégrale du gain de l'ampli moteur de cet axe. Cette valeur multiplie l'erreur cumulative entre la position commandée et la position actuelle en unités machine, elle entre dans le calcul de la tension appliquée à l'ampli moteur. Les unités du gain **I** sont des Volts sur des unités machine par seconde, exemple : $\frac{volt}{mm\,s}$ si l'unité machine est le millimètre.

D = 0 (HAL) La composante **d**érivée du gain de l'ampli moteur de cet axe. Cette valeur multiplie la différence entre l'erreur courante et les précédentes, elle entre dans le calcul de la tension appliquée à l'ampli moteur. Les unités du gain **D** sont des Volts sur des unités machine sur des secondes, exemple : $\frac{volt}{mm/s}$ si l'unité machine est le millimètre.

[4] Référez vous au "Manuel de l'intégrateur d'EMC2" pour des informations complémentaires sur les systèmes à servomoteurs et leur contrôle en PID.

FF0 = 0 (HAL) Gain à priori (feedforward) d'orde 0. Cette valeur est multipliée par la position commandée, elle entre dans le calcul de la tension appliquée à l'ampli moteur. Les unités du gain FF0 sont des Volts sur des unités machine, exemple : $\frac{volt}{mm}$ si l'unité machine est le millimètre.

FF1 = 0 (HAL) Gain à priori (feedforward) de premier orde. Cette valeur est multipliée par l'écart de la position commandée par seconde, elle entre dans le calcul de la tension appliquée à l'ampli moteur. Les unités du gain FF1 sont des Volts sur des unités machine par seconde, exemple : $\frac{volt}{mm\,s}$ si l'unité machine est le millimètre.

FF2 = 0 (HAL) Gain à priori (feedforward) de second orde. Cette valeur est multipliée par l'écart de la position commandée par seconde au carré, elle entre dans le calcul de la tension appliquée à l'ampli moteur. Les unités du gain FF2 sont des Volts sur des unités machine par des secondes au carré, exemple : $\frac{volt}{mm\,s^2}$ si l'unité machine est le millimètre.

OUTPUT_SCALE = 1.000

OUTPUT_OFFSET = 0.000 (HAL) Ces deux valeurs sont, l'échelle et le facteur d'offset de l'ampli moteur de cet axe. La seconde valeur (offset) est soustraite de la valeur de sortie calculée (en Volts) puis divisée par la première valeur (facteur d'échelle), avant d'être écrite dans le convertisseur D/A. Les unités du facteur d'échelle sont des Volts réels par Volts en sortie de DAC. Les unités de la valeur d'offset sont en Volts. Ces valeurs peuvent être utilisées pour linéariser un DAC.
Plus précisément, quand les sorties sont écrites, EMC converti d'abord les unités quasi-SI des sorties concernées en valeurs brutes, exemple : Volts pour un amplificateur DAC. Cette mise à l'échelle ressemble à cela :

$$raw = \frac{output - offset}{scale}$$

La valeur d'échelle peut être obtenue par analyse des unités, exemple : les unités sont [unités SI en sortie]/[unités de l'actuateur]. Par exemple, sur une machine sur laquelle une tension de consigne de l'ampli de 1 Volt donne une vitesse de 250 mm/sec :

$$amplifier[volts] = (output[\frac{mm}{sec}] - offset[\frac{mm}{sec}])/250\frac{mm}{sec\,volt}$$

Notez que les unités d'offset sont en unités machine, exemple : mm/sec et qu'elles sont déjà soustraites depuis la sonde de lecture. La valeur de cet offset est obtenue en prenant la valeur de votre sortie qui donne 0,0 sur la sortie de l'actuateur. Si le DAC est linéarisé, cet offset est normallement de 0.0.
L'échelle et l'offset peuvent être utilisés pour linéariser les DAC, d'où des valeurs qui reflètent les effets combinés du gain de l'ampli, de la non linéarité du DAC, des unités du DAC, etc. Pour ce faire, suivez cette procédure :

1. Construire un tableau de calibrage pour la sortie, piloter le DAC avec la tension souhaitée et mesurer le résultat. Voir le tableau 5.3.7.2 pour un exemple de mesures de tension.
2. Effectuer un "least squares" linéaire pour obtenir les coefficients a, b tels que :

$$meas = a * raw + b$$

3. Notez que nous voulons des sorties brutes de sorte que nos résultats mesurés soient identiques à la sortie commandée. Ce qui signifie :

   (a)
   $$cmd = a * raw + b$$

   (b)
   $$raw = (cmd - b)/a$$

4. En conséquence, les coefficients a et b d'ajustement linéaire peuvent être directement utilisés comme valeurs d'échelle et d'offset pour le contrôleur.

MAX_OUTPUT = 10 (HAL) La valeur maximale pour la sortie de la compensation PID pouvant être envoyée sur l'ampli moteur, en Volts. La valeur calculée de la sortie sera fixée à cette valeur limite. La limite est appliquée avant la mis à l'échelle de la sortie en unités brutes.

MIN_OUTPUT = -10 (HAL) La valeur minimale pour la sortie de la compensation PID pouvant être envoyée sur l'ampli moteur, en Volts. La valeur calculée de la sortie sera fixée à cette valeur limite. La limite est appliquée avant la mis à l'échelle de la sortie en unités brutes.

Mesure des tensions de sortie

Raw (brutes)	Mesurées
-10	-9.93
-9	-8.83
0	-0.03
1	0.96
9	9.87
10	10.87

INPUT_SCALE = 40000 (HAL) Spécifie le nombre d'impulsions qui correspond à un mouvement d'une unité machine. Un second chiffre, si spécifié, sera ignoré.
La valeur de l'échelle peut être obtenue par l'analyse des unités, exemple : les unités sont :

$$\frac{sensor\ units}{desired\ input\ SI\ units}$$

Par exemple, sur un codeur de 2000 top par tour, un réducteur de 10 tours/pouce et des unités demandées en mm, nous avons :

$$\begin{aligned} input\_scale &= 2000\frac{counts}{rev} * 10\frac{rev}{inch} * \frac{1\,inch}{25.4\,mm} \\ &= 787.40157\frac{counts}{mm} \end{aligned}$$

5.3.7.3 Variables relatives aux moteurs pas à pas

SCALE = 40000 (HAL) Spécifie le nombre d'impulsions qui correspond à un mouvement d'une unité machine. Pour les sytèmes à moteurs pas à pas, c'est le nombre d'impulsions de pas nécessaires pour avancer d'une unité machine. Pour les systèmes à servo, c'est le nombre d'impulsions de retour signifiant que le mobile a avancé d'une unité machine. Un second chiffre, si spécifié, sera ignoré.
La valeur de l'échelle peut être obtenue par l'analyse des unités, exemple : les unités sont :

$$\frac{step\ units}{desired\ input\ SI\ units}$$

Par exemple, un pas moteur de 1.8 degré, en mode demipas, avec une réduction de 10 tours/pouce et des unités souhaitées en mm, nous avons :

$$\begin{aligned} input\_scale &= \frac{2\,count}{1.8\,degree} * 10\frac{rev}{inch} * \frac{1\,inch}{25.4\,mm} \\ &= 157.48031\frac{counts}{mm} \end{aligned}$$

D'anciens fichiers de configuration .ini et .hal utilisaient INPUT_SCALE pour cette valeur.

STEPGEN_MAXACCEL = 21.0 (HAL) Limite d'accélération pour le générateur de pas. Elle doit être 1% à 10% supérieure à celle de l'axe MAX_ACCELERATION. Cette valeur améliore les réglages de la "boucle de position" de stepgen.

STEPGEN_MAXVEL = 1.4 (HAL) Les anciens fichiers de configuration avaient également une limite de vitesse du générateur de pas. Si spécifiée, elle doit auss être 1% à 10% supérieure à celle de l'axe MAX_VELOCITY. Des test ultérieurs ont montré que l'utilisation de STEPGEN_MAXVEL n'améliore pas le réglage de la boucle de position de stepgen.

5.3.8 Section [EMCIO]

CYCLE_TIME = 0.100 La période en secondes, à laquelle EMCIO va tourner. La mettre à 0.0 ou à une valeur négative fera qu'EMCIO tournera en permanence. Il est préférable de ne pas modifier cette valeur.

TOOL_TABLE = tool.tbl Ce fichier contient les informations des outils.

TOOL_CHANGE_POSITION = 0 0 2 Spécifies la position XYZ ou le mobile se déplacera lors d'un appel d'outil.

5.4 Prise d'origine

5.4.1 Vue d'ensemble

La prise d'origine semble assez simple, il suffit de déplacer chaque axe à un emplacement connu et de positionner l'ensemble des variables internes d'EMC en conséquence. Toutefois, les machines sont différentes les unes des autres et la prise d'origine est maintenant devenue assez complexe.

5.4.2 Séquence de prise d'origine

La figure 5.4.2 montre les quatre séquences de prise d'origine possibles, avec les variables de configuration associées. Une description détaillée de ces paramètres sera faite au chapitre suivant.

5.4.3 Configuration

Il y a six éléments d'information qui définissent le déroulement de la séquence de prise d'origine. Ils sont définis dans la section [AXIS] du fichier ini.

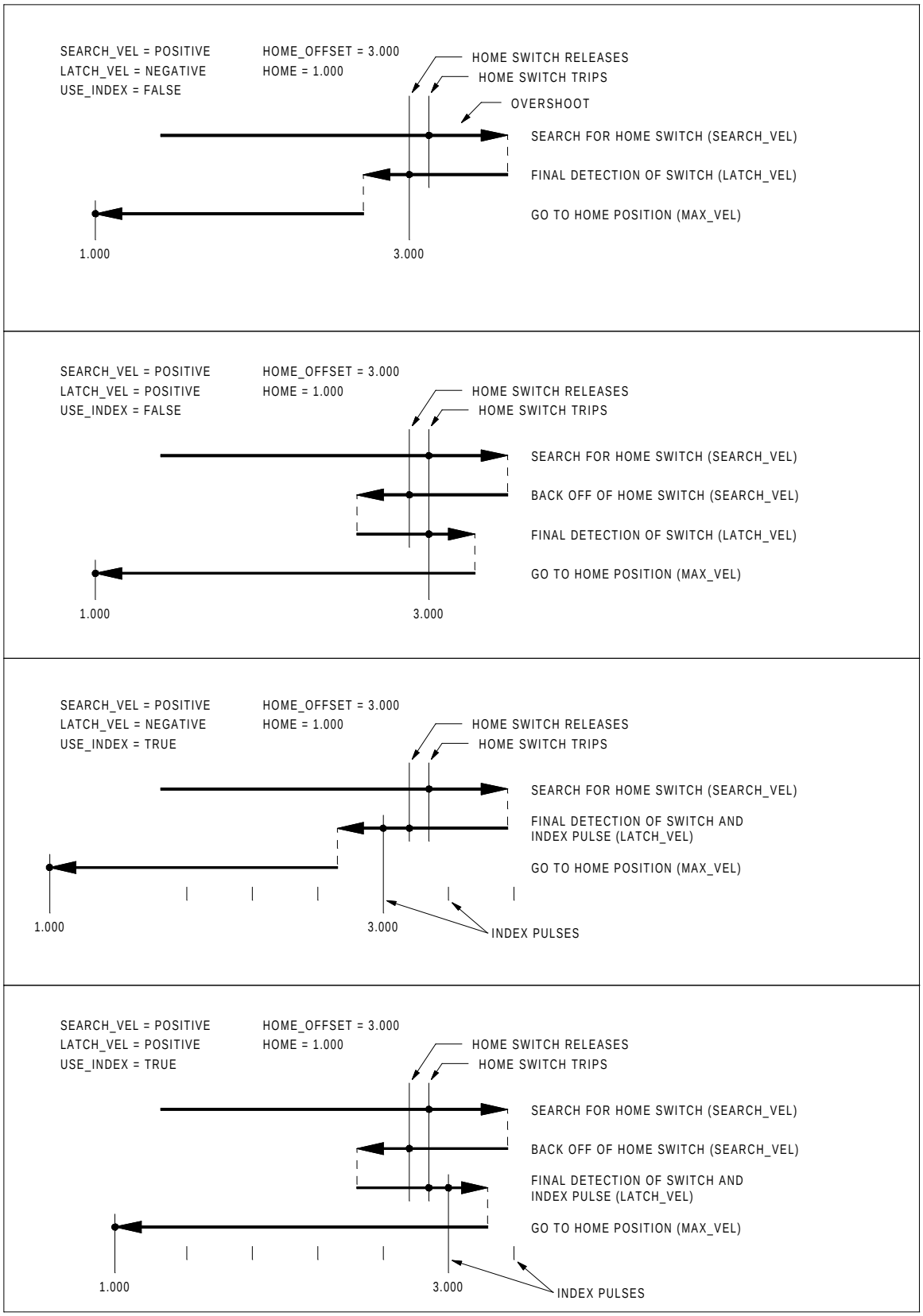
5.4.3.1 HOME_SEARCH_VEL = 0

Vitesse de la phase initiale de prise d'origine, c'est la recherche du contact d'origine machine. Une valeur différente de zéro indique à EMC la présence d'un contact d'origine machine. EMC va alors commencer par vérifier si ce contact est déjà pressé. Si oui, il le dégagera à la vitesse établie par "HOME_SEARCH_VEL", la direction du dégagement sera de signe opposé à celui de "HOME_SEARCH_VEL". Puis, il va revenir vers le contact en se déplaçant dans la direction spécifiée par le signe de "HOME_SEARCH_VEL" et à la vitesse déterminée par sa valeur absolue. Quand le contact d'origine machine est détecté, le mobile s'arrête aussi vite que possible, il y aura cependant toujours un certain dépassement dépendant de la vitesse. Si celle-ci est trop élevée, le mobile peut dépasser suffisamment le contact pour aller attaquer un fin de course de limite d'axe, voir même aller se crasher dans une butée mécanique. À l'opposé, si "HOME_SEARCH_VEL" est trop basse, la prise d'origine peut durer très longtemps.

Une valeur égale à zéro indique qu'il n'y a pas de contact d'origine machine, dans ce cas, les phases de recherche de ce contact seront occultées. La valeur par défaut est zéro.

5.4.3.2 HOME_LATCH_VEL=0

Spécifie la vitesse et la direction utilisée par le mobile pendant la dernière phase de la prise d'origine, c'est la recherche précise du contact d'origine machine, si il existe et de l'emplacement de l'impulsion d'index, si elle est présente. Cette vitesse est plus lente que celle de la phase initiale, afin d'améliorer la précision. Si "HOME_SEARCH_VEL" et "HOME_LATCH_VEL" sont de mêmes signes, la phase de recherche précise s'effectuera dans le même sens que la phase de recherche initiale. Dans ce cas, le mobile dégagera d'abord le contact en sens inverse avant de revenir vers lui à la vitesse définie ici. L'acquisition de la position d'origine se fera sur la première impulsion de changement d'état du contact. Si "HOME_SEARCH_VEL" et "HOME_LATCH_VEL" sont de signes opposés, la phase de recherche précise s'effectuera dans le sens opposé à celui de la recherche initiale. Dans ce cas, EMC dégagera le contact à la vitesse définie ici. L'acquisition de la position d'origine se fera sur la première impulsion de changement d'état du contact lors de son dégagement. Si "HOME_SEARCH_VEL" est à zéro, signifiant qu'il n'y a pas de contact et que "HOME_LATCH_VEL" et différente de zéro, le mobile continuera jusqu'à la prochaine impulsion d'index. Si "HOME_SEARCH_VEL" est différent de zéro et que "HOME_LATCH_VEL" est égal à zéro, c'est une cause d'erreur, l'opération de prise d'origine échouera. La valeur par défaut est zéro.



Séquences de prise d'origine possibles

5.4.3.3 HOME_IGNORE_LIMITS = YES/NO

Peut contenir les valeurs YES ou NO. Cette variable détermine si EMC doit ignorer les fins de course de limites d'axe. Certaines machines n'utilisent pas un contact d'origine séparé, à la place, elles utilisent un des interrupteurs de fin de course comme contact d'origine. Dans ce cas, EMC doit ignorer l'activation de cette limite de course pendant la séquence de prise d'origine. La valeur par défaut de ce paramètre est NO.

5.4.3.4 HOME_USE_INDEX = YES/NO

Spécifie si une impulsion d'index doit être prise en compte (cas de règles de mesure ou de codeurs de positions). Si cette variable est vraie (HOME_USE_INDEX = YES), EMC fera l'acquisition de l'origine machine sur le premier front de l'impulsion d'index. Si elle est fausse (=NO), EMC fera l'acquisition de l'origine sur le premier front produit par le contact d'origine (dépendra des signes de "HOME_SEARCH_VEL" et "HOME_LATCH_VEL"). La valeur par défaut est NO.

5.4.3.5 HOME_OFFSET

Contient l'emplacement du point d'origine ou de l'impulsion d'index, en coordonnées relatives. Il peut aussi être traité comme le décalage entre le point d'origine machine et le zéro de l'axe. A la détection de l'impulsion d'origine, EMC ajuste les coordonnées de l'axe à la valeur de "HOME_OFFSET". La valeur par défaut est zéro.

5.4.3.6 HOME

C'est la position sur laquelle ira le mobile à la fin de la séquence de prise d'origine. Après avoir détecté le contact d'origine, avoir ajusté les coordonnées de ce point à la valeur de "HOME_OFFSET", le mobile va se déplacer sur la valeur de "HOME", c'est le point final de la séquence de prise d'origine. La valeur par défaut est zéro. Notez que même si ce paramètre est égal à la valeur de "HOME_OFFSET", le mobile dépassera très légérement la position du point d'aquisition de l'origine machine avant de s'arrêter. Donc il y aura toujours un petit mouvement à ce moment là (sauf bien sûr si "HOME_SEARCH_VEL" est à zéro, et que toute la séquence de POM a été sautée). Ce mouvement final s'effectue en vitesse de déplacement rapide. Puisque l'axe est maintenant référencé, il n'y a plus de risque pour la machine, un mouvement rapide est donc la façon la plus rapide de finir la séquence de prise d'origine.[5]

5.4.3.7 HOME_IS_SHARED

Si cet axe n'a pas un contact d'origine séparé des autres, mais plusieurs contacts câblés sur la même broche, mettez cette valeur à 1 pour éviter de commencer la prise d'origine si un de ces contacts partagés est déjà activé. Mettez cette valeur à 0 pour permettre la prise d'origine même si un contact est déjà attaqué.

5.4.3.8 HOME_SEQUENCE

Utilisé pour définir l'ordre des séquences "HOME ALL" de prise d'origine des différents axes (exemple : la POM de l'axe X ne pourra se faire qu'après celle de Z). La POM d'un axe ne pourra se faire qu'après tous les autres en ayant la valeur la plus petite de "HOME_SEQUENCE" et après qu'ils soient déjà tous à "HOME_OFFSET". Si deux axes ont la même valeur de "HOME_SEQUENCE", leur POM s'effectueront simultanément. Si "HOME_SEQUENCE" est égale à -1 ou non spécifiée, l'axe ne sera pas

[5] La distinction entre l'origine machine et le décalage d'origine n'est pas aussi claire que je le voudrais. J'envisage de faire un petit dessin et un exemple pour la clarifier.

compris dans la séquence “HOME ALL”. Les valeurs de “HOME_SEQUENCE” débutent à 0, il ne peut pas y avoir de valeur inutilisée.

Chapitre 6

Introduction

6.1 Qu'est-ce que HAL ?

HAL est le sigle de Hardware Abstraction Layer, le terme Anglais pour Couche d'Abstraction Matériel[1]. Au plus haut niveau, il s'agit simplement d'une méthode pour permettre à un grand nombre de "modules" d'être chargés et interconnectés pour assembler un système complexe. La partie "matériel" devient abstraite parce que HAL a été conçu à l'origine pour faciliter la configuration d'EMC pour une large gamme de matériels. Bon nombre de ces modules sont des pilotes de périphériques. Cependant, HAL peut faire beaucoup plus que configurer les pilotes du matériel.

6.1.1 HAL est basé sur le système traditionnel d'étude des projets techniques

HAL est basé sur le même principe que celui utilisé pour l'étude des circuits et des systèmes techniques, il va donc être utile d'examiner d'abord ces principes.

N'importe quel système, y compris les machines CNC, est fait de composants interconnectés. Pour les machines CNC, ces composants pourraient être le contrôleur principal, les amplis de servomoteurs, les amplis ou les commandes de puissance des moteurs pas à pas, les moteurs, les codeurs, les interrupteurs de fin de course, les panneaux de boutons de commande, les manivelles, peut être aussi un variateur de fréquence pour le moteur de broche, un automate programmable pour gérer le changeur d'outils, etc. Le constructeur de machine doit choisir les éléments, les monter et les câbler entre eux pour obtenir un système complet et fonctionnel.

6.1.1.1 Choix des organes

Il ne sera pas nécessaire au constructeur de machine de se soucier du fonctionnement de chacun des organes, il les traitera comme des boîtes noires. Durant la phase de conception, il décide des éléments qu'il va utiliser, par exemple, moteurs pas à pas ou servomoteurs, quelle marque pour les amplis de puissance, quels types d'interrupteurs de fin de course et combien il en faudra, etc. La décision d'intégrer tel ou tel élément spécifique plutôt qu'un autre, repose sur ce que doit faire cet élément et sur ses caractéristiques fournies par le fabricant. La taille des moteurs et la charge qu'ils doivent supporter affectera le choix des interfaces de puissance nécessaires pour les piloter. Le choix de l'ampli affectera le type des signaux de retour demandés ainsi que le type des signaux de vitesse et de position qui doivent lui être transmis.

Dans le monde de HAL, l'intégrateur doit décider quels composants de HAL sont nécessaires. Habituellement, chaque carte d'interface nécessite un pilote. Des composants supplémentaires peuvent être demandés, par exemple, pour la génération logicielle des impulsions d'avance, les fonctionnalités des automates programmables, ainsi qu'une grande variété d'autres tâches.

[1] Note du traducteur : nous garderons le sigle HAL dans toute la documentation.

6.1.1.2 Étude des interconnections

Le créateur d'un système matériel, ne sélectionnera pas seulement les éléments, il devra aussi étudier comment ils doivent être interconnectés. Chaque boîte noire dispose de bornes, deux seulement pour un simple contact, ou plusieurs douzaines pour un pilote de servomoteur ou un automate. Elles doivent être câblées entre elles. Les moteurs câblés à leurs interfaces de puissance, les fins de course câblés au contrôleur et ainsi de suite. Quand le constructeur de machine commence à travailler sur le câblage, il crée un grand plan de câblage représentant tous les éléments de la machine ainsi que les connections qui les relient entre eux.

En utilisant HAL, les *composants* sont interconnectés par des *signaux*. Le concepteur peut décider quels signaux sont nécessaires et à quoi ils doivent être connectés.

6.1.1.3 Implémentation

Une fois que le plan de câblage est complet, il est possible de construire la machine. Les pièces sont achetées et montées, elles peuvent alors être câblées et interconnectées selon le plan de câblage. Dans un système physique, chaque interconnection est un morceau de fil qui doit être coupé et raccordé aux bornes appropriées.

HAL fournit un bon nombre d'outils d'aide à la "construction" d'un système HAL. Certains de ces outils permettent de "connecter" (ou déconnecter) un simple "fil". D'autres permettent d'enregistrer une liste complète des organes, du câblage et d'autres informations à propos du système, de sorte qu'il puisse être "reconstruit" d'une simple commande.

6.1.1.4 Mise au point

Très peu de machines marchent bien dès la première fois. Lors des tests, le technicien peut utiliser un appareil de mesure pour voir si un fin de course fonctionne correctement ou pour mesurer la tension fournie aux servomoteurs. Il peut aussi brancher un oscilloscope pour examiner le réglage d'une interface ou pour rechercher des interférences électriques et déterminer leurs sources. En cas de problème, il peut s'avérer indispensable de modifier le plan de câblage, peut être que certaines pièces devront être recâblées différemment, voir même remplacées par quelque chose de totalement différent.

HAL fournit les équivalents logiciels du voltmètre, de l'oscilloscope, du générateur de signaux et les autres outils nécessaires à la mise au point et aux réglages d'un système. Les même commandes utilisées pour construire le système, seront utilisées pour faire les changements indispensables.

6.1.2 En résumé

Ce document est destiné aux personnes déjà capables de concevoir ce type de réalisation matérielle, mais qui ne savent pas comment connecter le matériel à EMC.

La conception de matériel, telle que décrite précédemment, s'arrête à l'interface de contrôle. Au delà, il y a un tas de boîtes noires, relativement simples, reliées entre elles pour faire ce qui est demandé. À l'intérieur, un grand mystère, c'est juste une grande boîte noire qui fonctionne, nous osons l'espérer.

HAL étend cette méthode traditionnelle de conception de matériel à l'intérieur de la grande boîte noire. Il transforme les pilotes de matériels et même certaines parties internes du matériel, en petites boîtes noires pouvant être interconnectées, elles peuvent alors remplacer le matériel externe. Il permet au "plan de câblage" de faire voir une partie du contrôleur interne et non plus, juste une grosse boîte noire. Plus important encore, il permet à l'intégrateur de tester et de modifier le contrôleur en utilisant les mêmes méthodes que celles utilisées pour le reste du matériel.

Les termes tels que moteurs, amplis et codeurs sont familiers aux intégrateurs de machines. Quand nous parlons d'utiliser un câble extra souple à huit conducteurs blindés pour raccorder un codeur de position à sa carte d'entrées placée dans l'ordinateur. Le lecteur comprend immédiatement de quoi il s'agit et se pose la question, "quel type de connecteurs vais-je devoir monter de chaque côté de ce câble ?" Le même genre de réflexion est indipensable pour HAL mais le cheminement de la pensée est différent. Au début les mots utilisés par HAL pourront sembler un peu étranges, mais ils sont identiques au concept de travail évoluant d'une connection à la suivante.

HAL repose sur une seule idée, l'idée d'étendre le plan de câblage à l'intérieur du contrôleur. Si vous êtes à l'aise avec l'idée d'interconnecter des boîtes noires matérielles, vous n'aurez sans doute aucune difficulté à utiliser HAL pour interconnecter des boites noires logicielles.

6.2 Concept de HAL

Cette section est un glossaire qui définit les termes clés de HAL mais il est différent d'un glossaire traditionnel en ce sens que les termes ne sont pas classés par ordre alphabétique. Ils sont classés par leur relation ou par le sens du flux à l'intérieur de HAL.

Component : (Composant) Lorsque nous avons parlé de la conception du matériel, nous avons évoqué les différents éléments individuels comme "pièces", "modules", "boîtes noires", etc. L'équivalent HAL est un "component" ou "HAL component". (ce document utilisera : "HAL component" quand la confusion avec un autre type de composant est possible, mais normalement, utilisez juste : "component".) Un HAL component est une pièce logicielle avec, bien définis, des entrées, des sorties, un comportement, qui peuvent éventuellement être interconnectés.

Parameter : (Paramètre) De nombreux composants matériels ont des réglages qui ne sont raccordés à aucun autre composant mais qui sont accessibles. Par exemple, un ampli de servomoteur a souvent des potentiomètres de réglage et des points tests sur lesquels on peut poser une pointe de touche de voltmètre ou une sonde d'oscilloscope pour visualiser le résultat des réglages. Les HAL components aussi peuvent avoir de tels éléments, ils sont appelés "parameters". Il y a deux types de paramètres : "Input parameters" qui sont des équivalents des potentiomètres. Ce sont des valeurs qui peuvent être réglées par l'utilisateur, elles gardent leur valeur jusqu'à un nouveau réglage. "Output parameters" qui ne sont pas ajustables. Ils sont équivalents aux points tests qui permettent de mesurer la valeur d'un signal interne.

Pin : (Broche) Les composants matériels ont des broches qui peuvent être interconnectées entre elles. L'équivalent HAL est une "pin" ou "HAL pin". ("HAL pin" est utilisé quand c'est nécessaire pour éviter la confusion.) Toutes les HAL pins sont nommées et les noms des pins sont utilisés lors des interconnexions entre elles. Les HAL pins sont des entités logicielles qui n'existent qu'à l'intérieur de l'ordinateur.

Physical_Pin : (Broche physique) La plupart des interfaces d'entrées/sorties ont des broches physiques réelles pour leur connection avec l'extérieur, par exemple, les broches du port parallèle. Pour éviter la confusion, elles sont appelées "physical pins". Ce sont des repères pour faire penser au monde physique réel.

Signal : Dans une machine physique réelle, les terminaisons des différents organes sont reliées par des fils. L'équivalent HAL d'un fil est un "signal" ou "HAL signal". Ces signaux connectent les "HAL pins" entre elles comme le requiert le concepteur de la machine. Les "HAL signals" peuvent être connectés et déconnectés à volonté (même avec la machine en marche).

Type : Quand on utilise un matériel réel, il ne viendrait pas à l'idée de connecter la sortie 24V d'un relais à l'entrée analogique +/-10V de l'ampli d'un servomoteur. Les "HAL pins" ont les même restrictions, qui sont fondées sur leur type. Les "pins" et les "signals" ont tous un type, un "signals" ne peux être connecté qu'à une "pins" de même type. Il y a actuellement les 4 types suivants :

- BIT - une simple valeur vraie ou fausse TRUE/FALSE ou ON/OFF
- FLOAT - un flottant de 32 bits, avec approximativement 24 bits de résolution et plus de 200 bits d'échelle dynamique.

– U32 - un entier non signé de 32 bits, les valeurs légales vont de 0 à +4294967295
– S32 - un entier signé de 32 bits, les valeurs légales vont de -2147483648 à +2147483647

Function : (Fonction) Les composants matériels réels ont tendance à réagir immédiatement à leurs signaux d'entrée. Par exemple, si la tension d'entrée d'un ampli de servo varie, la sortie varie aussi automatiquement. Les composants logiciels ne peuvent pas réagir immédiatement. Chaque composant a du code spécifique qui doit être exécuté pour faire ce que le composant est sensé faire. Dans certains cas, ce code tourne simplement comme une partie du composant. Cependant dans la plupart des cas, notamment dans les composants temps réel, le code doit être exécuté selon un ordre bien précis et à des intervalles trés précis. Par exemple, les données en entrée doivent d'abord être lues avant qu'un calcul ne puisse être effectué sur elles et les données en sortie ne peuvent pas être écrites tant que le calcul sur les données d'entrée n'est pas terminé. Dans ces cas, le code est confié au système sous forme de "functions". Chaque “function” est un bloc de code qui effectue une action spécifique. L'intégrateur peut utiliser des "threads" pour combiner des séries de “functions” qui seront exécutées dans un ordre particulier et selon des intervalles de temps spécifiques.

Thread : (Fil) Un "thread" est une liste de “functions” qui sont lancées à intervalles spécifiques par une tâche temps réel. Quand un “thread” est créé pour la première fois, il a son cadencement spécifique (période), mais pas de “functions”. Les “functions” seront ajoutées au “thread” et elle seront exécutées dans le même ordre, chaque fois que le “tread” tournera.

Prenons un exemple, supposons que nous avons un composant de port parallèle nommé “hal_parport”. Ce composant défini une ou plusieurs “HAL pins” pour chaque “physical pin”. Les “pins” sont décrites dans ce composant, comme expliqué dans la section “component“de cette doc, par : leurs noms, comment chaque “pin” est en relation avec la “physical pin”, est-elle inversée, peut-on changer sa polarité, etc. Mais ça ne permet pas d'obtenir les données des “HAL pins” aux “physical pins”. Le code est utilisé pour faire ça, et c'est la où les “functions” entrent en oeuvre. Le composant parport nécessite deux “functions” : une pour lire les broches d'entrée et mettre à jour les “HAL pins”, l'autre pour prendre les données des “HAL pins” et les écrire sur les broches de sortie “physical pins”. Ce deux fonctions font partie du pilote “hal_parport”.

6.3 Composants HAL

Chaque composant HAL est un morceau de logiciel avec, bien définis, des entrées, des sorties et un comportement. Ils peuvent être installés et interconnectés selon les besoins. Cette section liste certains des composants actuellement disponibles et décrit brièvement ce que chacun fait. Les détails complets sur chacun seront donnés plus loin dans ce document.

6.3.1 Programmes externes attachés à HAL

motion Un module temps réel qui accepte les commandes de mouvement en NML et inter-agit avec HAL

iocontrol Un module d'espace utilisateur qui accepte les commandes d'entrée/sortie (I/O) en NML et inter-agit avec HAL

classicladder Un automate programmable en langage à contacts utilisant HAL pour les entrées/sorties (I/O)

halui Un espace de utilisateur de programmation qui inter-agit avec HAL et envoie des commandes NML, Il est destiné à fonctionner comme une interface utilisateur en utilisant les boutons et interrupteurs externes.

6.3.2 Composants internes

stepgen Générateur d'impulsions de pas avec boucle de position.

encoder Codeur/compteur logiciel.

pid Boucle de contrôle Proportionnelle/Intégrale/Dérivée.

siggen Générateur d'ondes : sinusoïdale/cosinusoïdale/triangle/carrée, pour la mise au point.

supply Une simple alimentation, pour la mise au point

blocks Un assortiment de composants (mux, demux, or, and, integ, ddt, limit, wcomp, etc.)

6.3.3 Pilotes de matériels

hal_ax5214h Un pilote pour la carte d'entrées/sorties Axiom Measurement & Control AX5241H

hal_m5i20 Un pilote pour la carte Mesa Electronics 5i20

hal_motenc Un pilote pour la carte Vital Systems MOTENC-100

hal_parport Pilote pour le(ou les) port(s) parallèle(s).

hal_ppmc Un pilote pour la famille de contrôleurs Pico Systems (PPMC, USC et UPC)

hal_stg Un pilote pour la carte Servo To Go (versions 1 & 2)

hal_vti Un pilote pour le contrôleur Vigilant Technologies PCI ENCDAC-4

6.3.4 Outils-Utilitaires

halcmd Ligne de commande pour la configuration et les réglages.

halgui Outil graphique pour la configuration et les réglages. (pas encore implémenté).

halmeter Un multimètre pour les signaux HAL.

halscope Un oscilloscope digital à mémoire, complétement fonctionnel pour les signaux HAL.

Chacun de ces modules est décrit en détail dans les chapitres suivants.

6.4 Tinkertoys, Erector Sets, Legos et le HAL

Cette première introduction au concept de HAL peut être un peu déconcertante pour l'esprit. Construire quelque chose avec des blocs peut être un défi, pourtant certains jeux de construction avec lesquels nous avons joué étant enfants peuvent nous aider à construire un système HAL.

6.4.1 Une tour

> Je regardais mon fils et sa petite fille de six ans construire une tour à partir d'une boîte pleine de blocs de différentes tailles, de barres et de pièces rondes, des sortes de couvercles. L'objectif était de voir jusqu'où la tour pouvait monter. Plus la base était étroite plus il restait de pièces pour monter. Mais plus la base était étroite, moins la tour était stable. Je les voyais étudier combien de blocs ils pouvaient poser et où ils devaient les poser pour conserver l'équilibre avec le reste de la tour.

La notion d'empilage de cartes pour voir jusqu'où on peut monter est une très vieille et honorable manière de passer le temps. En première lecture, l'intégrateur pourra avoir l'impression que construire un HAL est un peu comme ça. C'est possible avec une bonne planification, mais l'intégrateur peut avoir à construire un système stable aussi complexe qu'une machine actuelle l'exige.

6.4.2 Erector Sets[2] (Meccano en France)

C'était une grande série de boites de construction en métal, des tôles perforées, plates ou en cornières, toutes avaient des trous régulierements espacés. Vous pouviez concevoir des tas de choses et les monter avec ces éléments maintenus entre eux par des petits boulons.

> J'ai eu ma première boîte Erector pour mon quatrième anniversaire. Je sais que la boîte était prévue pour des enfants beaucoup plus agés que moi. Peut être que mon père se faisait vraiment un cadeau à lui même. J'ai eu une période difficile avec les petites vis et les petits écrous. J'ai vraiment eu envie d'avoir quatre bras, un pour visser avec le tournevis, un pour tenir la vis, les pièces et l'écrou. En persévérant, de même qu'en agaçant mon père, j'ai fini par avoir fait tous les montages du livret. Bientôt, je lorgnais vers les plus grandes boîtes qui étaient imprimées sur ce livret. Travailler avec ces pièces de taille standard m'a ouvert le monde de la construction et j'ai bientôt été au delà des projets illustrés.

Les composants Hal ne sont pas tous de même taille ni de même forme mais ils permettent d'être regroupés en larges unités qui feront bien du travail. C'est dans ce sens qu'ils sont comme les pièces d'un jeu Erector. Certains composants sont longs et minces. Ils connectent essentiellement les commandes de niveau supérieur aux "physical pins". D'autres composants sont plus comme les plateformes rectangulaires sur lesquelles des machines entières pourraient être construites. Un intégrateur parviendra rapidement au delà des brefs exemples et commencera à assembler des composants entre eux d'une manière qui lui sera propre.

6.4.3 Tinkertoys[3]

> Le jouet en bois Tinkertoys est plus humain que l'acier froid de l'Erector. Le coeur de la construction avec TinkerToys est un connecteur rond avec huit trous équidistants sur la circonférence. Il a aussi un trou au centre, perpendiculaire aux autres trous répartis autour du moyeu.
>
> Les moyeux pouvaient être connectés avec des tiges rondes de différentes longueurs. Le constructeur pouvait faire une grosse roue à l'aide de rayons qui partaient du centre.
>
> Mon projet favori était une station spaciale rotative. De courtes tiges rayonnaient depuis les trous du moyeu central et étaient connectées avec d'autres moyeux aux extrémités des rayons. Ces moyeux extérieurs étaient raccordés entre eux avec d'autres rayons. Je passais des heures à rêver de vivre dans un tel dispositif, marchant de moyeu en moyeu et sur la passerelle extérieure qui tournait lentement à cause de la gravité dans l'espace en état d'apesanteur. Les provisions circulaient par les rayons et les ascenceurs qui les transféraient dans la fusée arrimée sur le rayon central pendant qu'on déchargeait sa précieuse cargaison.

L'idée qu'une "pin" ou qu'un "component" est la plaque centrale pour de nombreuses connections est aussi une notion facile avec le HAL. Les exemples deux à quatre (voir section 7) connectent le multimètre et l'oscilloscope aux signaux qui sont prévus pour aller ailleurs. Moins facile, la notion d'un moyeu pour plusieurs signaux entrants. Mais, c'est également possible avec l'utilisation appropriée des fonctions dans ce composant de moyeu qui manipulent les signaux quand ils arrivent, venant d'autres composants.

Une autre réflexion qui vient à partir de ce jouet mécanique est une représentation de "HAL threads". Un "thread" pourrait ressembler un peu à un chilopode, une chenille, ou un perce-oreille. Une épine dorsale, des "HAL components", raccordés entre eux par des tiges, les "HAL signals". Chaque composant prend dans ses propres paramètres et selon l'état de ses broches d'entrée, les passe sur ses broches de sortie à l'intention du composant suivant. Les signaux voyagent ainsi de bout en bout, le long de l'épine dorsale où ils sont ajoutés ou modifiés par chaque composant son tour venu.

[2] Le jeu Erector Set est une invention de AC Gilbert

[3] Tinkertoy est maintenant registered trademark of the Hasbro company.

Les “Threads” sont tous synchronisés et exécutent une série de tâches de bout en bout. Une représentation mécanique est possible avec Thinkertoys si on pense à la longueur du jouet comme étant la mesure du temps mis pour aller d'un bout à l'autre. Un thread, ou épine dorsale, très différent est créé en connectant le même ensemble de modules avec des tiges de longueur différente. La longueur totale de l'épine dorsale peut aussi être changée en jouant sur la longueur des tiges pour connecter les modules. L'ordre des opérations est le même mais le temps mis pour aller d'un bout à l'autre est très différent.

6.4.4 Un exemple en Lego[4]

Lorsque les blocs de Lego sont arrivés dans nos magasins, ils étaient à peu près tous de la même taille et de la même forme. Bien sûr il y avait les demi taille et quelques uns en quart de taille mais tous rectangulaires. Les blocs de Lego se relient ensembles en enfonçant les broches mâles d'une pièce dans les trous femelles de l'autre. En superposant les couches, les jonctions peuvent être rendues très solides, même aux coins et aux tés.

> J'ai vu mes enfants et mes petits-enfants construire avec des pièces Lego (les mêmes Lego). Il y en a encore quelques milliers dans une vieille et lourde boîte en carton qui dort dans un coin de la salle de jeux. Ils sont stockés dans cette boîte car c'était trop long de les ranger et de les ressortir à chacune de leur visite et ils étaient utilisés à chaque fois. Il doit bien y avoir les pièces de deux douzaines de boîtes différentes de Lego. Les petits livrets qui les accompagnaient ont été perdus depuis longtemps, mais la magie de la construction avec l'imbrication de ces pièces toutes de la même taille est quelque chose à observer.

6.5 Problèmes de timming dans HAL

Contrairement aux modèles physiques du câblage entre les boîtes noires sur lequel, nous l'avons dit, HAL est basé, il suffit de relier deux broches avec un signal hal, on est loin de l'action physique.

La vraie logique à relais consiste en relais connectés ensembles, quand un relais s'ouvre ou se ferme, le courant passe (ou s'arrête) immédiatement. D'autres bobines peuvent changer d'état etc. Dans le style langage à contacts d'automate comme le Ladder ça ne marche pas de cette façon. Habituellement dans un Ladder simple passe, chaque barreau de l'échelle est évalué dans l'ordre où il se présente et seulement une fois par passe. Un exemple parfait est un simple Ladder avec un contact en série avec une bobine. Le contact et la bobine actionnent le même relais.

Si c'était un relais conventionnel, dès que la bobine est sous tension, le contact s'ouvre et coupe la bobine, le relais retombe etc. Le relais devient un buzzer.

Avec un automate programmable, si la bobine est OFF et que le contact est fermé quand l'automate commence à évaluer le programme, alors à la fin de la passe, la bobine sera ON. Le fait que la bobine ouvre le contact qui la prive de courant est ignoré jusqu'à la prochaine passe. À la passe suivante, l'automate voit que le contact est ouvert et désactive la bobine. Donc, le relais va battre rapidement entre on et off à la vitesse à laquelle l'automate évalue le programme.

Dans HAL, c'est le code qui évalue. En fait, la version Ladder HAL temps réel de ClassicLadder exporte une fonction pour faire exactement cela. Pendant ce temps, un thread exécute les fonctions spécifiques à intervalle régulier. Juste comme on peut choisir de régler la durée de la boucle de programme d'un automate programmable à 10ms, ou à 1 seconde, on peut définir des “HAL threads” avec des périodes différentes.

Ce qui distingue un thread d'un autre n'est pas ce qu'il fait mais quelles fonctions lui sont attachées. La vraie distinction est simplement combien de fois un thread tourne.

[4]The Lego name is a trademark of the Lego company.

Dans EMC on peut avoir un thread à 50μs et un thread à 1ms. En se basant sur les valeurs de BASE_PERIOD et de SERVO_PERIOD. Valeurs fixées dans le fichier ini.

La prochaine étape consiste à décider de ce que chaque thread doit faire. Certaines de ces décisions sont les mêmes dans (presque) tous les systèmes emc. Par exemple, le gestionnaire de mouvement est toujours ajouté au servo-thread.

D'autres connections seront faites par l'intégrateur. Il pourrait s'agir de brancher la lecture d'un codeur par une carte STG à un DAC pour écrire les valeurs dans le servo thread, ou de brancher une fonction stepgen au base-thread avec la fonction parport pour écrire les valeurs sur le port.

Chapitre 7

Tutoriel de HAL

7.1 Introduction

La configuration passe de la théorie à la pratique de HAL. Pour ceux qui ont juste un peu de pratique avec la programmation des ordinateurs, cette section est le “Hello World” de HAL. Comme indiqué précédemment halrun peut être utilisé pour créer un système qui fonctionne. Il s’agit d’un outil de configuration et de mise au point en ligne de commande ou en fichier texte. Les exemples suivants illustrent son installation et son fonctionnement.

7.1.1 Notation

Les exemples en ligne de commande sont représentés en police **`bold typewriter`**. Les réponses de l’ordinateur sont en police `typewriter`. Le texte optionnel est entre crochets `[comme ça]`. Le texte `<comme ça>` représente un champ qui peut prendre différentes valeurs, le paragraphe adjacent explique quelles sont les valeurs appropriées. Les élements textuels séparés par des barres verticales indiquent qu’une valeur ou l’autre mais pas les deux, doit être présente. Toutes les lignes de commandes considèrent que vous êtes dans le répertoire `emc2/` , les chemins sont affichés en accord avec ce principe.

7.1.2 L’environnement RTAPI

RTAPI est le sigle de Real Time Application Programming Interface. De nombreux composants HAL travaillent en temps réel et tous les composants de HAL stockent leurs données dans la mémoire partagée, de sorte que les composants temps réel puissent y accéder. Normalement, Linux ne prend pas en charge les programmes temps réel ni le type de mémoire partagée dont HAL a besoin. Heureusement, il existe des systèmes d’exploitation temps réel RTOS qui fournissent les extensions nécessaires à Linux. Malheureusement, chaque RTOS fait les choses différemment des autres.

Pour remédier à ces différences, l’équipe d’EMC a proposé RTAPI, qui fournit une manière cohérente aux programmes de parler au RTOS. Si vous êtes un programmeur qui veut travailler à l’intérieur d’EMC, vous pouvez étudier `emc2/src/rtapi/rtapi.h` pour comprendre l’API. Mais si vous êtes une personne normale, tout ce que vous avez besoin de savoir à propos de RTAPI est qu’il doit être (avec le RTOS) chargé dans la mémoire de votre ordinateur avant de pouvoir faire n’importe quoi avec HAL.

Pour ce tutoriel, nous allons supposer que vous avez compilé avec succès l’arborescence emc2/src et, si nécessaire, invoqué le script `emc-environment` pour préparer votre shell. Dans ce cas, tout ce que vous avez à faire est de charger le RTOS requis et les modules RTAPI dans la mémoire. Tapez juste les commandes suivantes dans une console :

```
emc2$ halrun
halcmd :
```

Une fois l'OS temps réel et RTAPI chargés, vous pouvez aller au premier exemple. Notez que le prompt a changé, il est passé de "$" à "halcmd". La raison en est que les commandes ultérieures seront interprétées comme des commandes HAL et non plus comme des commandes shell. `halrun` est un simple script shell, il est plus ou moins équivalent de lancer :

```
emc2$ realtime start
emc2$ halcmd -kf
```

Pour quitter `halcmd` et que halrun arrête le système temps réel, tapez :

```
emc2$ realtime stop
```

Vous pouvez également passer des arguments à halrun, il seront répercutés sur halcmd, ou passer le nom d'un fichier .hal. Parce que `halrun` arrête le système temps réel quand il se termine, le fichier hal fonctionnera de cette façon et s'arrêtera généralement avec une commande comme : `loadrt -w halscope`.

7.2 Tab-complétion

Votre version de halcmd peut inclure la complétion avec la touche tab. Au lieu de compléter les noms de fichiers comme le fait un shell, il complète les commandes avec les identifiants HAL. Essayez de presser la touche tab après le début d'une commande HAL :

```
halcmd : lo<TAB>
loadrt    loadusr   lock
halcmd : loadrt d<TAB>
ddt       debounce
```

7.3 Un exemple simple

7.3.1 Chargement d'un composant temps réel

Pour le premier exemple, nous allons utiliser un composant HAL appelé `siggen`, qui est un simple générateur de signal. Une description complète du composant `siggen` reste disponible à la section **??** de ce document. Il s'agit d'un composant en temps réel, mis en oeuvre comme un module du noyau Linux. Pour charger `siggen` utiliser la commande `halcmd loadrt` :

```
halcmd : loadrt siggen
```

7.3.2 Examiner HAL

Maintenant que le module est chargé, il faut introduire `halcmd`, l'outil en ligne de commande utilisé pour configurer le HAL. Pour une description plus complète essayez : `man halcmd`, ou consultez la section `halcmd` à la section **??** de ce document. La première commande de halcmd et show, qui affichera les informations concernant l'état actuel du HAL. Pour afficher tout ce qui est installé tapez :

```
halcmd : show comp
Loaded HAL Components :
ID      Type  Name                                      PID   State
32769   RT    siggen                                          ready
9775    User  halcmd9775                                 9775 initializing
```

Puisque `halcmd` lui même est un composant HAL, il sera toujours présent dans la liste[1]. La liste montre aussi le composant `siggen` que nous avions installé à l'étape précédente. Le "RT" sous "Type" indique que `siggen` est un composant temps réel.

Ensuite, voyons quelles pins `siggen` rend disponibles :

```
halcmd : show pin
Component Pins :
Owner   Type  Dir    Value      Name 02    float -W   0.00000e+00  siggen.0.cosine
32769   float OUT   0.00000e+00  siggen.0.sawtooth
32769   float OUT   0.00000e+00  siggen.0.sine
32769   float OUT   0.00000e+00  siggen.0.square
32769   float OUT   0.00000e+00  siggen.0.triangle
```

Cette commande affiche toutes les pins dans le HAL. Un système complexe peut avoir plusieurs dizaines ou centaines de pins. Mais pour le moment il y a seulement cinq pins. Toutes ces cinq pins sont des flottants, elles transportent toutes des données en provenance du composant `siggen`. Puisque nous n'avons pas encore exécuté le code contenu dans le composant, toutes les pins ont une valeur de zéro.

L'étape suivante consiste à examiner les paramètres :

```
halcmd : show param
Parameters :
Owner     Type  Dir    Value      Name
 32769    float  RW   1.00000e+00  siggen.0.amplitude
 32769    float  RW   1.00000e+00  siggen.0.frequency
 32769    float  RW   0.00000e+00  siggen.0.offset
 32769    s32    RO             0   siggen.0.update.time
 32769    s32    RW             0   siggen.0.update.tmax
```

La commande "show param" affiche tous les paramètres de HAL. Pour le moment chaque paramètre à la valeur par défaut attribuée quand le composant a été chargé. Notez dans la colonne `Dir`. Les valeurs marquées `-W` écriture possible, pour ceux qui ne sont jamais modifié par le composant lui-même, mais qui sont modifiable par l'utilisateur pour contrôler le composant. Nous verrons comment plus tard. Les paramètres marqués `R-` sont en lecture seule. Il ne peuvent être modifiés que par le composant. Finalement, les paramètres marqués `RW` sont en lecture/écriture. Ils peuvent être modifiés par le composant et aussi par l'utilisateur. Nota : les paramètres `siggen.0.update.time` et `siggen.0.update.tmax` existent dans un but de déboguage, ils ne sont pas couverts par cette documentation.

La plupart des composants temps réel exportent une ou plusieurs fonctions pour que le code qu'elles contiennent soit exécuté en temps réel. Voyons ce que la fonction `siggen` exporte :

```
halcmd : show funct
Exported Functions :
Owner    CodeAddr   Arg    FP   Users  Name
 32769   b7f74ac5 b7d0c0b4 YES    0   siggen.0.update
```

Le composant siggen exporte une seule fonction. Il nécessite un flottant (Floating Point). Il n'est lié à aucun thread, puisque "users" est à zero[2].

[1] Le nombre après halcmd dans la liste des composants est le "process ID". Il est toujours possible de lancer plus d'une instance de halcmd en même temps (dans différentes fenêtres par exemple), Le numéro PID est ajouté à la fin du nom pour rendre celui-ci unique.

[2] Les champs codeaddr et arg ont été utilisés pendant le développement et devraient probablement disparaître.

7.3.3 Exécuter le code temps réel

Pour faire tourner le code actuellement contenu dans la fonction `siggen.0.update`, nous avons besoin d'un thread temps réel. C'est le composant appelé `threads` qui est utilisé pour créer le nouveau thread. Créons un thread appelé `test-thread` avec une période de 1ms (1000000ns) :

```
halcmd : loadrt threads name1=test-thread period1=1000000
```

Voyons si il fonctionne :

```
halcmd : show thread
Realtime Threads :
   Period   FP   Name    (Time, Max-Time)
     999849 YES  test-thread    ( 0, 0 )
```

Il fonctionne. La période n'est pas exactement de 1000000ns à cause des limitations dues au matériel, mais nous avons bien un thread qui tourne à une période approximativement correcte et qui peut manipuler des fonctions en virgule flottante. La prochaine étape sera de connecter la fonction au thread :

```
halcmd : addf siggen.0.update test-thread
```

Pour le moment nous avions utilisé `halcmd` seulement pour regarder le HAL. Mais cette fois-ci, nous avons utilisé la commande `addf` (add function) pour changer quelque chose dans le HAL. Nous avons dit à `halcmd` d'ajouter la fonction `siggen.0.update` au thread `test-thread` et la commande suivante indique qu'il a réussi :

```
halcmd : show thread
Realtime Threads :
   Period   FP   Name    (Time, Max-Time)
     999849 YES  test-thread    ( 0, 0 )
                 1 siggen.0.update
```

Il y a une étape de plus avant que le composant `siggen` ne commence à générer des signaux. Quand le HAL est démarré pour la première fois, les threads ne sont pas en marche. C'est pour vous permettre de compléter la configuration du système avant que le code temps réel ne démarre. Une fois que vous êtes satisfait de la configuration, vous pouvez lancer le code temps réel comme ceci :

```
halcmd : start
```

Maintenant le générateur de signal est en marche. Regardons ses pins de sortie :

```
halcmd : show pin
Component Pins :
Owner  Type  Dir     Value      Name
 32769    float OUT   2.12177e-01  siggen.0.cosine
 32769    float OUT  -5.64055e-01  siggen.0.sawtooth
 32769    float OUT   9.79820e-01  siggen.0.sine
 32769    float OUT  -1.00000e+00  siggen.0.square
 32769    float OUT   1.28110e-01  siggen.0.triangle
halcmd : show pin
Component Pins :
Owner  Type  Dir     Value      Name
 32769    float OUT   5.19530e-01  siggen.0.cosine
 32769    float OUT   6.73893e-01  siggen.0.sawtooth
 32769    float OUT  -8.54452e-01  siggen.0.sine
 32769    float OUT   1.00000e+00  siggen.0.square
 32769    float OUT   3.47785e-01  siggen.0.triangle
```

Nous avons fait, très rapidement, deux commandes `show pin` et vous pouvez voir que les sorties ne sont plus à zéro. Les sorties sinus, cosinus, dents de scie et triangle changent constamment. La sortie carrée fonctionne également, mais elle passe simplement de +1.0 à -1.0 à chaque cycle.

7.3.4 Modifier des paramètres

La réelle puissance de HAL est de permettre de modifier les choses. Par exemple, on peut utiliser la commande `setp` pour ajuster la valeur d'un paramètre. Modifions l'amplitude du signal de sortie du générateur de 1.0 à 5.0 :

```
halcmd : setp siggen.0.amplitude 5
emc2$
```

Voyons encore une fois les paramètres et les pins :

```
halcmd : setp siggen.0.amplitude 5
halcmd : show param
Parameters :
Owner  Type  Dir    Value      Name
 32769    float  RW   5.00000e+00  siggen.0.amplitude
 32769    float  RW   1.00000e+00  siggen.0.frequency
 32769    float  RW   0.00000e+00  siggen.0.offset
 32769    s32    RO          397   siggen.0.update.time
 32769    s32    RW       109100   siggen.0.update.tmax
halcmd : show pin
Component Pins :
Owner  Type  Dir     Value      Name
 32769    float OUT   4.78453e+00  siggen.0.cosine
 32769    float OUT  -4.53106e+00  siggen.0.sawtooth
 32769    float OUT   1.45198e+00  siggen.0.sine
 32769    float OUT  -5.00000e+00  siggen.0.square
 32769    float OUT   4.02213e+00  siggen.0.triangle
```

Notez que la valeur du paramètre `siggen.0.amplitude` est bien passée à 5.000 et que les pins ont maintenant des valeurs plus grandes.

7.3.5 Enregistrer la configuration de HAL

La plupart de ce que nous avons fait jusqu'ici avec `halcmd` a été de simplement regarder les choses avec la commande `show`. Toutefois, deux commandes ont rééllement modifié des valeurs. Au fur et à mesure que nous concevons des systèmes plus complexes avec HAL, nous allons utiliser de nombreuses commandes pour le configurer comme nous le souhaitons. HAL a une mémoire d'éléphant et peut retenir sa configuration jusqu'à ce qu'il s'arrête. Mais qu'en est-il de la prochaine fois ? Nous ne voulons pas entrer une série de commande à chaque fois que l'on veut utiliser le système. Nous pouvons enregistrer la configuration de l'ensemble de HAL en une seule commande :

```
halcmd : save
# components
loadrt threads name1=test-thread period1=1000000
loadrt siggen
# signals
# links
# parameter values
setp siggen.0.amplitude  5.00000e+00
setp siggen.0.frequency  1.00000e+00
```

```
setp siggen.0.offset  0.00000e+00
# realtime thread/function links
addf siggen.0.update test-thread
```

La sortie de la commande `save` est une séquence de commandes HAL. Si vous commencez par un HAL "vide" et que vous tapez toute la séquence de commandes HAL, vous aurez la configuration qui existait lors de l'exécution de la commande `save`. Pour sauver ces commandes pour une utilisation ultérieure, nous allons simplement rediriger la sortie vers un fichier :

```
halcmd : save all saved.hal
```

7.3.6 Restaurer la configuration de HAL

Pour restaurer la configuration de HAL stockée dans `saved.hal`, nous avons besoin d'exécuter toutes ces commandes HAL. Pour ce faire, nous utiliserons la commande `-f <filename>` qui lit les commandes à partir d'un fichier, le `-I` qui affichera le prompt halcmd après l'exécution des commandes :

```
emc2$ halrun -I -f saved.hal
```

Notez qu'il n'y a pas de commande 'start' dans le fichier saved.hal. Il est nécessaire de la retaper (ou d'éditer saved.hal pour l'ajouter) :

```
halcmd : start
```

7.4 Visualiser le HAL avec halmeter

Vous pouvez construire des systèmes HAL vraiment complexes sans utiliser d'interface graphique. Mais il y a quelque chose de rassurant à visualiser le résultat du travail. Le premier, et le plus simple des outils graphiques pour le HAL, est "halmeter". C'est un programme très simple qui s'utilise comme un multimètre.

Nous allons utiliser de nouveaux éléments du composant siggen pour vérifier halmeter. Si vous avez fini l'exemple précédent, alors siggen est déjà chargé. Sinon, on peut charger tout comme nous l'avons fait précédemment :

```
emc2$ halrun
halcmd : loadrt siggen
halcmd : loadrt threads name1=test-thread period1=1000000
halcmd : addf siggen.0.update test-thread
halcmd : start
halcmd : setp siggen.0.amplitude 5
```

7.4.1 Lancement de halmeter

À ce stade, nous avons chargé le composant siggen qui est en cours d'exécution. Nous pouvons lancer halmeter. Puisque halmeter est une application graphique, X doit être actif.

```
halcmd : loadusr halmeter
```

Dans le même temps, une fenêtre s'ouvre sur votre écran, ressemblant à la figure 7.1.

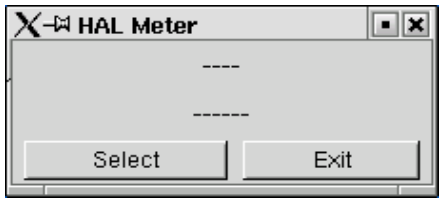


FIG. 7.1 – Lancement de Halmeter, rien n'est sélectionné

FIG. 7.2 – Dialogue de sélection de Halmeter

7.4.2 Utilisation de halmeter

La fenêtre de la figure 7.1 n'est pas d'une grande utilité, parce qu'elle n'affiche rien. Pour afficher des valeurs, cliquez sur le bouton 'Select', le dialogue de sélection des éléments à mesurer (figure 7.2).

Ce dialogue contient trois onglets. Le premier onglet affiche toutes les HAL pins du système. La seconde affiche tous les signaux et le troisième affiche tous les paramètres. Si nous voulons analyser la pin `siggen.0.triangle`, il suffit de cliquer sur elle puis sur le bouton 'OK'. Le dialogue de sélection se ferme et la mesure s'affiche, quelque chose comme sur la figure 7.3.

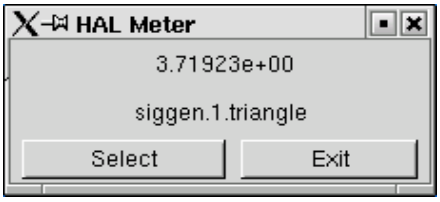


FIG. 7.3 – Affichage dans Halmeter de la valeur d'une pin

Vous devriez voir la valeur évoluer à mesure que siggen génère son signal triangulaire. Halmeter est raffraîchi environ 5 fois par seconde.

Si vous voulez visualiser rapidement des pins successives, vous pouvez cliquer le bouton 'Accept' du dialogue de sélection. Cliquez 'Select' pour ouvrir le dialoque une nouvelle fois. Mais cette fois, cliquez une autre pin, comme siggen.0.cosine, puis cliquez 'Accept'. La nouvelle valeur s'affiche alors immédiatement, mais le dialogue de sélection reste ouvert. Essayez d'afficher un paramètre au lieu d'une pin. Cliquez sur l'onglet 'Parameters', puis sélectionnez un paramètre et cliquez encore sur 'Accept'. Vous pouvez ainsi afficher très rapidement la valeur d'un élément puis d'un autre en quelques clics.

Pour arrêter halmeter, cliquez simplement sur le bouton "quitter".

Si vous voulez regarder plus d'une pin, plus d'un signal, ou plusieurs paramètres à la fois, il vous suffit de lancer plusieurs instances de halmeters. La fenêtre de halmeter a été conçue petite, pour permettre d'en avoir beaucoup à la fois sur l'écran. [3]

[3] Halmeter sera réécrit. Cette réécriture le rendra plus agréable à utiliser. La notation scientifique sera éliminée, elle est trop difficile à lire. Certaines formes d'échelles seront ajoutées (y compris, une échelle automatique) pour afficher des échelles plus grandes sans notation scientifique. Un affichage par "bargraph analogique" sera également ajouté pour donner une idée des évolutions rapides. Quand cette réécriture sera terminée, ces captures d'écran et les textes les accompagnant seront révisés pour refléter la nouvelle version.

7.5 Un exemple un peu plus complexe.

Jusqu'à maintenant, nous avons chargé un composant HAL. Mais l'idée générale de HAL est de vous permettre de charger et de relier un grand nombre de composants pour en faire un système complexe. Le prochain exemple va utiliser deux composants.

Avant de mettre en place ce nouvel exemple, nous allons commencer par un petit nettoyage. Si vous avez fini l'un des exemples précédents, il faut supprimer tous les composants et ensuite recharger la RTAPI et les librairies de HAL en faisant :

```
halcmd : exit
emc2$ halrun
```

7.5.1 Installation des composants

Maintenant, nous allons charger le composant générateur d'impulsions. Pour l'instant, nous pouvons nous passer des détails et exécuter les commandes suivantes :[4]

```
halrun : loadrt freqgen step_type=0,0
halcmd : loadrt siggen
halcmd : loadrt threads name1=fast fp1=0 period1=50000 name2=slow period2=1000000
```

La première commande charge deux générateurs d'impulsions, configurés pour générer des impulsions de type 0. La seconde commande charge notre vieil ami siggen et la troisième crée deux threads, un rapide (fast) avec une période de 50µs et un lent avec une période de 1ms. Le thread rapide ne prend pas en charge les fonctions à virgule flottante (fp1=0).

Comme précédemment, on peut utiliser `halcmd show` pour jeter un coup d'oeil au HAL. Cette fois, nous aurons beaucoup plus de pins et de paramètres que précédemment :

```
halcmd : show pin
Component Pins :
Owner  Type  Dir    Value      Name
 03    float -W   0.00000e+00  siggen.0.cosine
 03    float -W   0.00000e+00  siggen.0.sawtooth
 03    float -W   0.00000e+00  siggen.0.sine
 03    float -W   0.00000e+00  siggen.0.square
 03    float -W   0.00000e+00  siggen.0.triangle
 02    s32   -W            0   freqgen.0.counts
 02    bit   -W     FALSE      freqgen.0.dir
 02    float -W   0.00000e+00  freqgen.0.position
 02    bit   -W     FALSE      freqgen.0.step
 02    float R-   0.00000e+00  freqgen.0.velocity
 02    s32   -W            0   freqgen.1.counts
 02    bit   -W     FALSE      freqgen.1.dir
 02    float -W   0.00000e+00  freqgen.1.position
 02    bit   -W     FALSE      freqgen.1.step
 02    float R-   0.00000e+00  freqgen.1.velocity
halcmd : show param
Parameters :
Owner  Type  Dir    Value      Name
 03    float  -W   1.00000e+00  siggen.0.amplitude
 03    float  -W   1.00000e+00  siggen.0.frequency
```

[4] Le signe "\" à la fin d'une longue ligne indique que la ligne est tronquée (c'est nécessaire pour formater ce document). Quand vous entrez la commande en ligne dans la console, sautez simplement le "\" (ne pressez pas Entrée) et continuez à taper la ligne suivante.

```
03    float  -W   0.00000e+00  siggen.0.offset
02    u32    -W     000000001  freqgen.0.dirhold
02    u32    -W     000000001  freqgen.0.dirsetup
02    float  R-   0.00000e+00  freqgen.0.frequency
02    float  -W   0.00000e+00  freqgen.0.maxaccel
02    float  -W   1.00000e+15  freqgen.0.maxfreq
02    float  -W   1.00000e+00  freqgen.0.position-scale
02    s32    R-             0  freqgen.0.rawcounts
02    u32    -W     000000001  freqgen.0.steplen
02    u32    -W     000000001  freqgen.0.stepspace
02    float  -W   1.00000e+00  freqgen.0.velocity-scale
02    u32    -W     000000001  freqgen.1.dirhold
02    u32    -W     000000001  freqgen.1.dirsetup
02    float  R-   0.00000e+00  freqgen.1.frequency
02    float  -W   0.00000e+00  freqgen.1.maxaccel
02    float  -W   1.00000e+15  freqgen.1.maxfreq
02    float  -W   1.00000e+00  freqgen.1.position-scale
02    s32    R-             0  freqgen.1.rawcounts
02    u32    -W     000000001  freqgen.1.steplen
02    u32    -W     000000001  freqgen.1.stepspace
02    float  -W   1.00000e+00  freqgen.1.velocity-scale
```

7.5.2 Connecter les pins avec des signaux

Nous avons donc deux générateurs d'impulsions carrées et un générateur de signaux. Maintenant, nous allons créer des signaux HAL pour connecter ces trois composants. Nous allons faire comme si nous pilotions les axes X et Y d'une machine avec nos générateurs d'impulsions. Nous voulons déplacer la table en ronds. Pour ce faire, nous allons envoyer un signal cosinusoïdal à l'axe des X et un signal sinusoïdal à l'axe des Y. Le module siggen créera le sinus et le cosinus, mais nous aurons besoin de "fils" pour connecter les modules ensemble. Dans le HAL, les "fils" sont appelés signaux. Nous devons en créer deux. Nous pouvons les appeler comme on veut, pour cet exemple il y aura `X_vel` et `Y_vel`. Le signal `X_vel` partira de la sortie cosinus du générateur de signal et arrivera sur l'entrée "velocity" du premier générateur d'impulsions. La première étape consiste à connecter le signal à la sortie du générateur de signaux. Pour connecter un signal à une pin, nous utilisons la commande `net` :

```
halcmd : net X_vel <= siggen.0.cosine
```

Pour voir l'effet de la commande `net`, regardons les signaux :

```
halcmd : show sig
ignals :
Type      Value      Name
float   0.00000e+00  X_vel
                         <== siggen.0.cosine
```

Quand un signal est connecté à une ou plusieurs pins, la commande `show` liste les pins immédiatement suivies par nom du signal. Les flèches montrent la direction du flux de données, dans ce cas, les flux vont de la pin `siggen.0.cosine` pour le signal `X_vel`. Maintenant, connectons `X_vel` à l'entrée "velocity" du générateur d'impulsions carrées :

```
halcmd : net X_vel => freqgen.0.velocity
```

On peut aussi connecter l'axe Y au signal `Y_vel`. Il doit partir de la sortie sinus du générateur de signal pour arriver sur l'entrée du second générateur d'impulsions carrées. La commande suivante fait en une ligne, la même chose que les deux commandes `net` précédentes ont fait pour `X_vel` :

```
halcmd : net Y_vel siggen.0.sine => freqgen.1.velocity
```

Regardons une dernière fois les signaux et les pins connectés ensembles :

```
halcmd : show sig
Signals :
Type      Value      Name
float   0.00000e+00  X_vel
                         <== siggen.0.cosine
                         ==> freqgen.0.velocity
float   0.00000e+00  Y_vel
                         <== siggen.0.sine
                         ==> freqgen.1.velocity
```

Avec la commande `show sig` on voit clairement comment les flux de données traversent le HAL. Par exemple, le signal `X_vel` part de la pin `siggen.0.cosine` et arrive sur la pin `freqgen.0.velocity`.

7.5.3 Exécuter les réglages du temps réel - threads et functions

Penser à ce qui circule dans les “fils” rend les pins et les signaux assez faciles à comprendre. Les threads et les fonctions sont un peu plus difficiles. Les fonctions contiennent des instructions pour l'ordinateur. Les threads sont les méthodes utilisées pour faire exécuter ces instructions quand c'est nécessaire. Premièrement, regardons les fonctions dont nous disposons :

```
halcmd : show funct
Exported Functions :
Owner CodeAddr   Arg    FP  Users  Name
 03   D89051C4 D88F10FC YES    0   siggen.0.update
 02   D8902868 D88F1054 YES    0   freqgen.capture_position
 02   D8902498 D88F1054 NO     0   freqgen.make_pulses
 02   D89026F0 D88F1054 YES    0   freqgen.update_freq
```

En règle générale, vous devez vous référer à la documentation de chaque composant pour voir ce que font ses fonctions. Dans notre exemple, la fonction `siggen.0.update` est utilisée pour mettre à jour les sorties du générateur de signaux. Chaque fois qu'elle est exécutée, le générateur recalcule les valeurs de ses sorties sinus, cosinus, triangle, carrée. Pour générer un signal régulier, il doit fonctionner à des intervalles bien spécifiques.

Les trois autres fonctions sont liées au générateur d'impulsions :

La première, `freqgen.capture_position`, est utilisée pour un retour de position. Elle capture la valeur d'un compteur interne qui compte les impulsions qui sont générées. S'il n'y a pas de perte de pas, ce compteur indique la position du moteur.

La fonction principale du générateur d'impulsions est `freqgen.make_pulses`. Chaque fois que `make_pulses` démarre elle décide qu'il est temps de faire un pas, si oui il fixe les sorties en conséquence. Pour des pas plus doux, il doit fonctionner le plus souvent possible. Parce qu'il a besoin de fonctionner de manière rapide, `make_pulses` est hautement optimisé et n'effectue que quelques calculs. Contrairement aux autres, il n'a pas besoin de virgule flottante pour ses calculs.

La dernière fonction, `freqgen.update_freq`, est responsable de l'échelle et de quelques autres calculs qui ne doivent être effectués que lors d'une commande de changement de fréquence.

Pour notre exemple nous voulons faire tourner `siggen.0.update` avec une vitesse de calcul des valeurs sinus et cosinus modérée. Immédiatement après, nous lançons `siggen.0.update`. Nous voulons lancer `freqgen.update_freq` pour charger les nouvelles valeurs dans le générateur d'impulsions. Finalement nous lancerons `freqgen.make_pulses` aussi vite que possible pour des pas plus doux. Comme nous n'utilisons pas de retour de position, nous n'avons pas besoin de lancer `freqgen.capture_position`.

Nous lançons les fonctions en les ajoutant aux threads. Chaque thread va à une vitesse spécifique. Regardons de quels threads nous disposons :

```
halcmd : show thread
Realtime Threads :
   Period   FP   Name
    1005720 YES  slow    ( 0, 0 )
      50286 NO   fast    ( 0, 0 )
```

Les deux `threads` ont été créés lorsque nous les avons chargés. Le premier, `slow`, tourne toutes les millisecondes, il est capable d'exécuter des fonctions en virgule flottante (FP). Nous l'utilisons pour `siggen.0.update` et `freqgen.update_freq`. Le deuxième thread est `fast`, il tourne toutes les 50 microsecondes, il ne prend pas en charge les calculs en virgule flottante. Nous l'utilisons pour `freqgen.make_pulses`. Pour connecter des fonctions au bon thread, nous utilisons la commande `addf`. Nous spécifions la fonction en premier suivie par le thread :

```
halcmd : addf siggen.0.update slow
halcmd : addf freqgen.update_freq slow
halcmd : addf freqgen.make_pulses fast
```

Après avoir lancé ces commandes, nous pouvons lancer la commande `show thread` une nouvelle fois pour voir ce qui ce passe :

```
halcmd : show thread
Realtime Threads :
   Period   FP   Name    (Time, Max-Time)
    1005720 YES  slow    ( 0, 0 )
                  1 siggen.0.update
                  2 freqgen.update-freq
      50286 NO   fast    ( 0, 0 )
                  1 freqgen.make-pulses
```

Maintenant, chaque thread est suivi par les noms des fonctions, dans l'ordre dans lequel les fonctions seront exécutées.

7.5.4 Réglage des paramètres

Nous sommes presque prêts à démarrer notre système HAL. Mais il faut auparavant régler quelques paramètres. Par défaut le composant siggen génère des signaux qui varient entre +1 et -1. Pour notre exemple, c'est très bien, nous voulons que la vitesse de la table varie de +1 à -1 pouce par seconde. Toutefois, l'échelle du générateur d'impulsions de pas n'est pas bonne. Par défaut, il génère une fréquence de sortie de 1 pas par seconde avec une capacité de 1000. Il est fort improbable qu'un pas par seconde nous donne une vitesse de déplacement de la table d'un pouce par seconde. Supposons que notre vis fasse 5 tours par pouce, couplée à un moteur pas à pas de 200 pas par tour et une interface qui fournit 10x micropas par pas. Il faut donc 2000 pas pour faire un tour de vis et 5 tours pour faire un pouce. Ce qui signifie que notre montage utilisera 10000 pas par pouce. Nous avons besoin de multiplier la vitesse d'entrée à l'étape générateur d'impulsions par 10000 pour obtenir la bonne valeur. C'est exactement pour cela qu'existe le paramètre `freqgen.n.velocity-scale`. Dans notre cas, les axes X et Y ont la même échelle et nous pouvons passer les deux paramètres à 10000 :

```
halcmd : setp freqgen.0.velocity-scale 10000
halcmd : setp freqgen.1.velocity-scale 10000
```

Cela signifie que, avec la pin `freqgen.0.velocity` à 1.000 et le générateur réglé pour 10000 impulsions par seconde (10kHz), avec le moteur et la vis décrits précédemment, nos axes auront une vitesse de déplacement de exactement 1.000 pouce par seconde. Cela illustre une notion clé du concept de HAL, des éléments comme les échelles étant au plus bas niveau possible, dans notre exemple le générateur d'impulsions de pas, le signal interne `X_vel` est celui de la vitesse de déplacement de la table en pouces par seconde. Les autres composants comme `siggen` ne savent rien du tout à propos de l'échelle des autres. Si on change de vis, ou de moteur, il n'y a qu'un seul paramètre à changer, l'échelle du générateur d'impulsions de pas.

7.5.5 Lançons le !

Nous avons maintenant tout configuré et sommes prêts à démarrer. Tout comme dans le premier exemple, nous utilisons la commande `start` :

```
halcmd : start
```

Bien que rien ne semble se produire, à l'intérieur de l'ordinateur les impulsions de pas sont présentes sur la sortie du générateur, variant entre 10kHz dans un sens et 10kHz dans l'autre à chaque seconde. Dans la suite de ce tutoriel, nous allons voir comment convertir ces signaux internes des moteurs dans le monde réel, mais nous allons d'abord les examiner pour voir ce qui se passe.

7.6 Voyons-y de plus près avec halscope.

L'exemple précédent génère certains signaux très intéressants. Mais beaucoup de ce qui se passe est beaucoup trop rapide pour être vu avec halmeter. Pour examiner de plus près ce qui se passe à l'intérieur de HAL, il faudrait un oscilloscope. Heureusement HAL en a un, appelé halscope.

7.6.1 Démarrer Halscope

Halscope comporte deux parties, une partie en temps réel qui est chargée comme un module de noyau et une partie utilisateur qui fournit l'interface graphique et l'affichage. Cependant, vous n'avez pas à vous inquiéter à ce sujet car l'interface demandera automatiquement que la partie temps réel soit chargée :

```
halcmd : loadusr halscope
```

La fenêtre graphique du scope s'ouvre, immédiatement suivie par un dialogue "Realtime function not linked" visible sur la figure 7.4[5].

Ce dialogue est l'endroit où vous définissez le taux d'échantillonnage de l'oscilloscope. Pour le moment nous voulons un échantillon par milliseconde, alors cliquez sur le thread "slow" 1.03mSec et laissez le multiplicateur à 1. Nous allons aussi passer la longueur d'enregistrement à 4047 samples (échantillons), de sorte que nous puissions utiliser jusqu'à 4 canaux simultanément. Quand vous sélectionnez un thread puis que vous cliquez sur le bouton "OK", le dialogue disparaît et la fenêtre du scope affiche quelque chose comme sur la figure 7.5.

[5]Plusieurs de ces captures d'écran font référence à des threads nommés "siggen.thread" et "stepgen.thread" au lieu de "slow" et "fast". Lorsque les captures d'écran ont été faites, les composants thread n'existaient pas et différentes méthodes étaient utilisées pour créer des threads en leur donnant des noms différents. Aussi, les captures d'écran montrent des , etc, comme "stepgen.xxx" au lieux de "freqgen.xxx". Le nom original du module freqgen était stepgen et je n'ai pas eux le temps depuis pour refaire toutes les captures avec les nouveaux noms. Le nom "stepgen" actuel fait référence à un générateur d'impulsions de pas différent, un qui accepte les commandes de position au lieu de celles de vitesse. Les deux sont décrit en détail plus loin dans ce document.

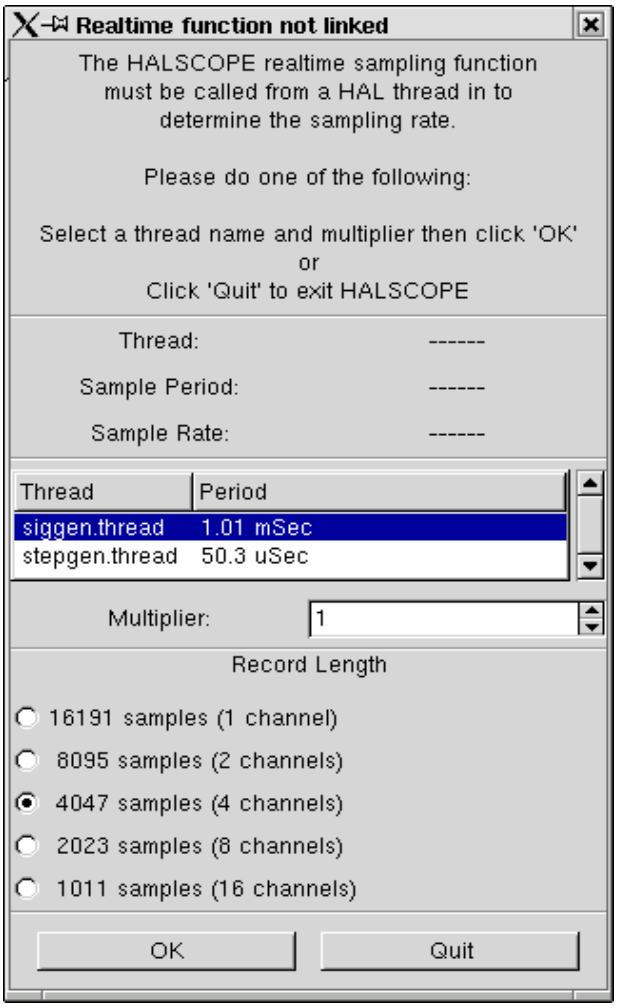


FIG. 7.4 – Dialogue“Realtime function not linked”

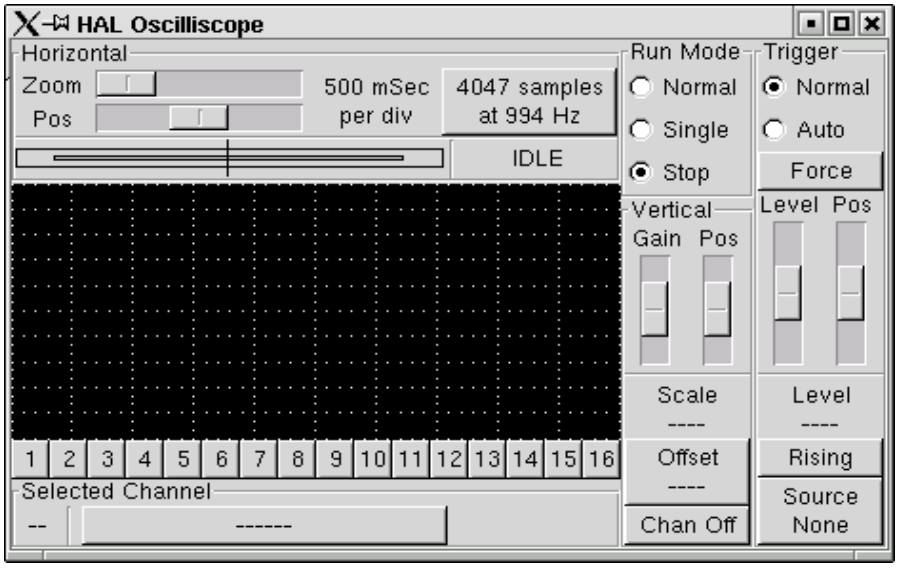


FIG. 7.5 – Fenêtre initiale du scope

7.6.2 Branchement des “sondes du scope”

À ce stade, Halscope est prêt à l’emploi. Nous avons déjà choisi le taux d’échantillonnage et la longueur d’enregistrement, de sorte que la prochaine étape consiste à décider de ce qu’il faut mesurer. C’est équivalent à brancher les “sondes virtuelles du scope” à HAL. Halscope dispose de 16 canaux, mais le nombre que vous pouvez utiliser à un moment donné dépend de la longueur d’enregistrement, plus il y a de canaux, plus les enregistrements doivent être courts, car la mémoire disponible pour l’enregistrement est fixée à environ 16000 échantillons.

Les boutons des canaux se situent en dessous de l’écran du scope. Cliquez le boutton “1” et vous verrez apparaître le dialogue de sélection “Select Channel Source”, comme sur la figure 7.6. Ce dialogue est très similaire à celui utilisé par Halmeter. Nous aimerions bien regarder les signaux que nous avons défini précédemment, pour cela, cliquons sur l’onglet “Signaux” et le dialogue affichera tous les signaux existants dans le HAL (seulement deux dans notre exemple).

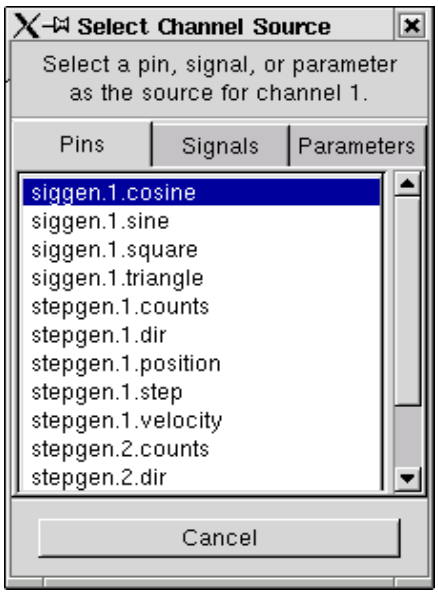


FIG. 7.6 – Dialogue de sélection du scope

Pour choisir un signal, il suffit de cliquer dessus. Dans notre cas, nous voulons utiliser le canal 1 pour afficher le signal “ X_vel”. Lorsque l’on clique sur “ X_vel”, la fenêtre se ferme et le canal a été sélectionné. Le bouton du canal 1 est pressé et le numéro du canal 1 et le nom “ X_vel” apparaissent sous la rangée de boutons. L’affichage indique toujours le canal sélectionné, vous pouvez avoir beaucoup de canaux sur l’écran, mais celui qui est actif sera en surbrillance. Les différents contrôles comme la position verticale et l’amplitude sont toujours relatifs au canal 1. Pour ajouter un signal au canal 2, cliquez sur le bouton "2”. Dans la fenêtre de dialogue, cliquez sur l’onglet "Signals”, puis cliquez sur “ Y_vel”.

Nous voulons aussi voir les signaux carrés et triangles produits. Il n’existe pas de signaux connectés à ces pins, nous utilisons donc l’onglet “Pins”. Pour le canal 3, sélectionnez “ siggen.0.triangle” et pour le canal 4, choisissez “ siggen.0.square”.

7.6.3 Capturer notre première forme d'onde

Maintenant que nous avons plusieurs sondes branchées sur HAL, il est temps de capturer quelques formes d'ondes. Pour démarrer le scope, cochez la case "Normal" du groupe "run mode" (en haut à droite). Puisque nous avons une longueur d'enregistrement de 4000 échantillons et une acquisition de 1000 échantillons par seconde, il faudra à halscope environ 2 secondes pour remplir la moitié de son tampon. Pendant ce temps, une barre de progression juste au-dessus de l'écran principal affichera le remplissage du tampon. Une fois que le tampon est à moitié plein, scope attend un déclencheur. Puisque nous n'en avons pas encore configuré, il attendra toujours. Pour déclencher manuellement, cliquez sur le bouton "Force" du groupe " Trigger" en haut à droite. Vous devriez voir le reste de la zone tampon se remplir, puis l'écran afficher les ondes capturées. Le résultat ressemble à la figure 7.7.

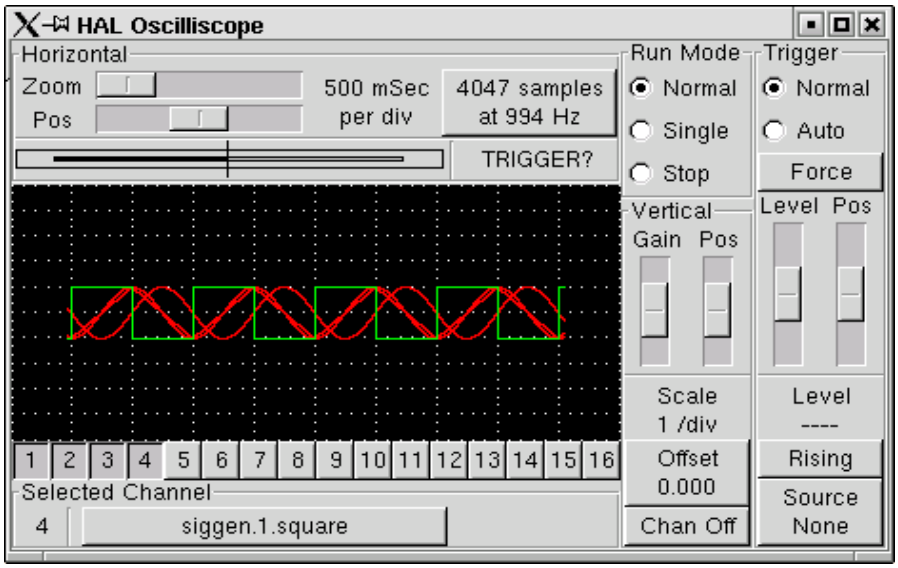


FIG. 7.7 – Capture d'ondes

Le grand bouton "Selected Channel" en bas, indique que le canal 4 est actuellement sélectionné, donc, qu'il correspond à l'onde verte, celle de la mesurée sur la pin "siggen.1.square". Essayez de cliquer sur les autres canaux pour mettre leurs traces en évidence.

7.6.4 Ajustement vertical

Les traces sont assez difficiles à distinguer car toutes les quatre sont les unes sur les autres. Pour résoudre ce problème, nous utilisons le curseur “Vertical” situé à droite de l’écran. Ces contrôles agissent sur le canal actuellement sélectionné. En ajustant le gain, notez qu’il couvre une large échelle (contrairement aux scopes réels), celle-ci permet d’afficher des signaux très petits (pico unités) à très grands (Tera - unités). Le curseur “position” déplace la trace affichée de haut en bas sur toute la hauteur de l’écran. Pour de plus grands ajustements le bouton Offset doit être utilisé (voir les références halscope dans la section **??** pour plus de détails).

7.6.5 Triggering

L’utilisation du bouton “Force” n’est parfois pas satisfaisante pour déclencher le scope. Pour régler un déclenchement (triggering) réel, cliquez sur le bouton “Source” situé en bas à droite. Il ouvre alors le dialogue “Trigger Source”, qui est simplement la liste de toutes les sondes actuellement branchées (Figure 7.8). Sélectionnez la sonde à utiliser pour déclencher en cliquant dessus. Pour notre exemple nous utilisons 3 canaux, essayez l’onde triangle.

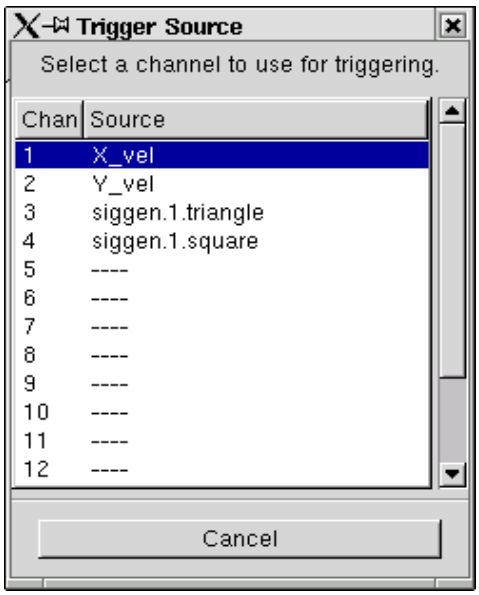


FIG. 7.8 – Dialogue des sources de déclenchement

Après avoir défini les sources de déclenchement, vous pouvez ajuster le niveau de déclenchement avec les curseurs dans la boîte “ Trigger” le long du bord droit. Le niveau peut être modifié à partir du haut vers le bas de l’écran, et est affiché sous les curseurs. La position est l’emplacement du point de déclenchement dans l’enregistrement complet. Avec le curseur tout en bas, le point de déclenchement est à la fin de l’enregistrement, et halscope affiche ce qui s’est passé avant le point de déclenchement. Lorsque le curseur est tout en haut, le point de déclenchement est au début de l’enregistrement, l’affichage représente ce qui s’est passé après le point de déclenchement. Le point de déclenchement est visible comme une ligne verticale dans la barre de progression située juste au dessus de l’écran. La polarité du signal de déclenchement peut être inversée en cliquant sur le bouton situé juste sous l’affichage du niveau de déclenchement. Notez que la modification de la position de déclenchement arrête le scope une fois la position ajustée, vous relancez le scope en cliquant sur le bouton “Normal” du groupe “Run Mode”.

Maintenant que nous avons réglé la position verticale et le déclenchement, l’écran doit ressembler à la figure 7.9.

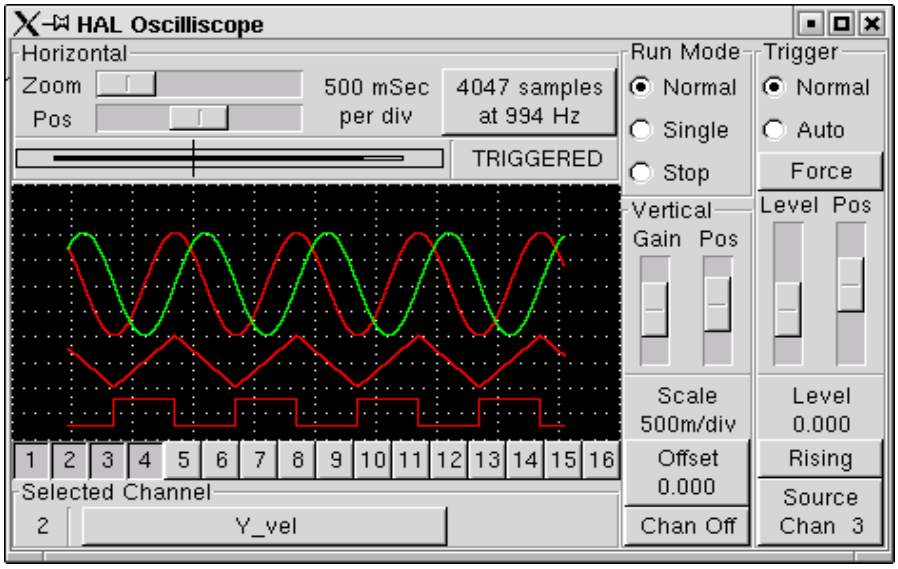


FIG. 7.9 – Formes d'ondes avec déclenchement

7.6.6 Ajustement horizontal

Pour examiner de près une partie d'une forme d'onde, vous pouvez utiliser le zoom au dessus de l'écran pour étendre la trace horizontalement et le curseur de position pour déterminer quelle partie de l'onde zoomée est visible. Parfois simplement élargir l'onde n'est pas suffisant et il faut augmenter la fréquence d'échantillonnage. Par exemple, nous aimerions voir les impulsions de pas qui sont générés dans notre exemple. Mais les impulsions de pas font seulement 50µs de long, l'échantillonnage à 1kHz n'est pas assez rapide. Pour changer le taux d'échantillonnage, cliquez sur le bouton qui affiche la longueur de l'enregistrement et l'échantillonnage pour avoir le dialogue "Select Sample Rate", figure 7.10. Pour notre exemple, nous cliquerons sur le thread à 50uS, "fast", qui fournira un échantillonnage à environ 20KHz. Maintenant au lieu d'afficher environ 4 secondes de données, un enregistrement est de 4000 échantillons à 20KHz, soit environ 0.20 seconde.

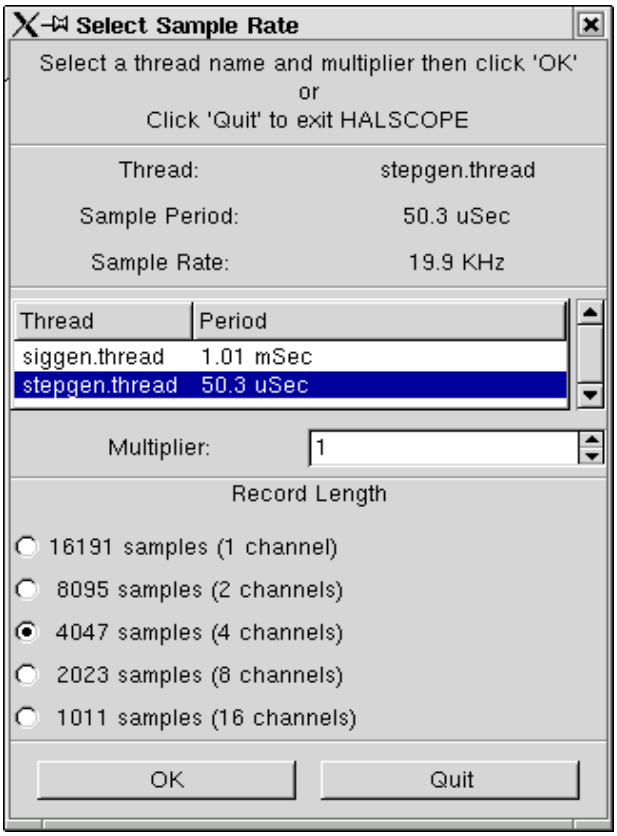


FIG. 7.10 – Dialogue de choix d'échantillonnage

7.6.7 Plus de canaux

Maintenant regardons les impulsions de pas. Halscope dispose de 16 canaux, mais pour cet exemple, nous en utilisons seulement 4 à la fois. Avant de sélectionner tout autre canal, nous avons besoin d'en éteindre certains. Cliquez sur le canal 2, puis sur le bouton "Off" sous le groupe " vertical". Ensuite, cliquez sur le canal 3, mettez le off et faites de même pour le canal 4. Même si les circuits sont éteints, ils ont encore en mémoire ce à quoi ils sont connectés et en fait, nous continuerons d'utiliser le canal 3 comme source de déclenchement. Pour ajouter de nouveaux canaux, sélectionnez le canal 5, choisissez la pin " stepgen.1.dir", puis canal 6 et sélectionnez "stepgen.1.step". Ensuite, cliquez sur "mode Normal" pour lancer le scope, ajustez le zoom horizontal à 5ms par division. Vous devriez voir les impulsions de pas ralentir à la vitesse commandée (channel 1) approcher de zéro, puis la pin de direction changer d'état et les impulsions de pas reprendre de nouveau de la vitesse. Vous aurez peut être besoin d'augmenter le gain sur le canal 1 à environ 20ms par division afin de mieux voir l'évolution de la vitesse de commande. Le résultat devrait être proche de celui de la figure 7.11.

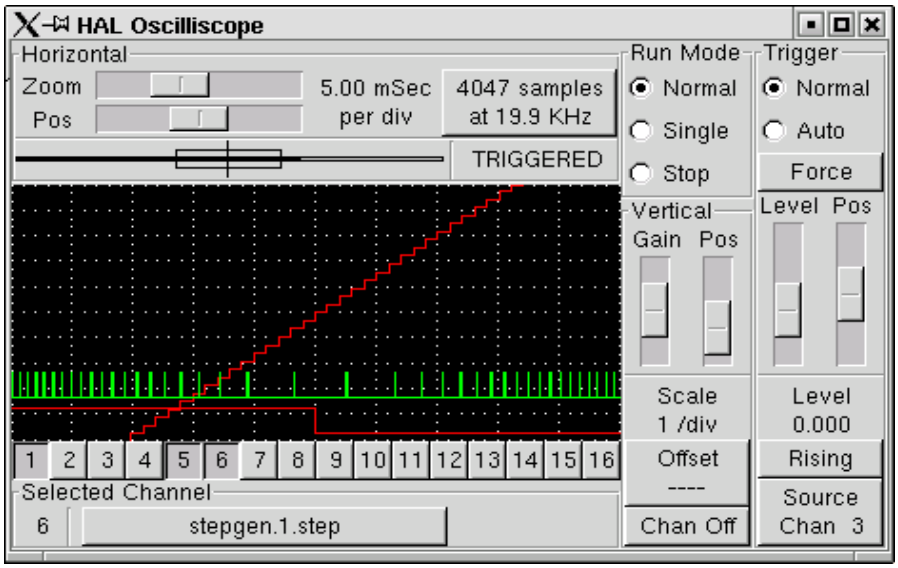


FIG. 7.11 – Observer les impulsions de pas

7.6.8 Plus d'échantillons

Si vous souhaitez enregistrer plus d'échantillons à la fois, redémarrez le temps réel et chargez halscope avec un argument numérique qui indique le nombre d'échantillons que vous voulez capturer, comme :

```
halcmd : loadusr halscope 80000
```

Si le composant `scope_rt` n'est pas déjà chargé, halscope va le charger et lui demander un total de 80000 échantillons, de sorte que lorsque l'échantillonnage se fera sur 4 canaux à la fois, il y aura 20000 échantillons par canal. (Si `scope_rt` est déjà chargé, l'argument numérique passé à halscope sera sans effet)

Chapitre 8

Pilotes matériels

8.1 Parport

Parport est un pilote pour le port parallèle traditionnel des PC. Le port dispose d'un total de 17 broches physiques. Le port parallèle originel a divisé ces broches en trois groupes : données, contrôle et état. Le groupe "données" consiste en 8 broches de sortie, le groupe "contrôle" consiste en 4 broches et le groupe "état" consiste en 5 broches d'entrée.

Au début des années 1990, le port parallèle bidirectionnel est arrivé, ce qui permet à l'utilisateur d'ajuster le groupe des données comme étant des sorties ou comme étant des entrées. Le pilote de HAL supporte le port bidirectionnel et permet à l'utilisateur de configurer le groupe des données en entrée ou en sortie. Si il est configuré en sortie, un port fournit un total de 12 sorties et 5 entrées. Si il est configuré en entrée, il fournit 4 sorties et 13 entrées.

Dans certains ports parallèles, les broches du groupe contrôle sont des collecteurs ouverts, ils peuvent aussi être mis à l'état bas par une porte extérieure. Sur une carte avec les broches de contrôle en collecteur ouvert, le mode "x" de HAL permet un usage plus flexible avec 8 sorties dédiées, 5 entrées dédiées et 4 broches en collecteur ouvert. Dans d'autres ports parallèles, les broches du groupe contrôle sont en push-pull et ne peuvent pas être utilisées comme des entrées.[1]

Aucune autre combinaison n'est supportée. Un port ne peut plus être modifié pour passer d'entrées en sorties dès que le pilote est installé. La figure 8.1 montre deux diagrammes, un montre le pilote quand le groupe de données est configuré en sortie et le second le montre configuré en entrée.

Le pilote parport peut contrôler au maximum 8 ports (défini par MAX_PORTS dans le fichier hal_parport.c). Les ports sont numérotés à partir de zéro.

8.1.1 Installation

```
loadrt hal_parport cfg="<config-string>"
```

[1] HAL ne peut pas déterminer automatiquement si les broches en mode bidirectionnel "x" sont effectivement en collecteur ouvert (OC). Si elles n'y sont pas, elles ne peuvent pas être utilisées comme des entrées. Essayer de les passer à l'état BAS par une source extérieure peut détériorer le matériel.

Pour déterminer si votre port a des broches de contrôle en "collecteur ouvert", charger hal_parport en mode "x", positionner les broches de contrôle à une valeur HAUTE. HAL doit lire des pins à l'état VRAI. Ensuite, insérer une résistance de 470Ω entre une des broches de contrôle et GND du port parallèle. Si la tension de cette broche de contrôle est maintenant proche de 0V et que HAL la lit comme une pin FAUSSE, alors vous avez un port OC. Si la tension résultante est loin de 0V ou que HAL ne la lit pas comme étant FAUSSE, votre port ne ne peut pas être utilisé en mode "x".

Le matériel extérieur qui pilote les broches de contrôle devrait également utiliser des portes en collecteur ouvert (ex : 74LS05...). Généralement, une pin de HAL -out devrait être VRAIE quand la pin physique est utilisée comme une entrée.

Sur certaines machines, les paramètres du BIOS peuvent affecter la possibilité d'utiliser le mode "x". Le mode "SPP" est le mode qui fonctionne le plus fréquemment.

La chaine config-string représente l'adresse hexadécimale du port, suivie, optionnellement, par une direction, le tout répété pour chaque port. Les directions sont "in", "out", ou "x", elles déterminent la direction des broches physiques 2 à 9 et s'il y a lieu de créer des pins d'entrée de HAL pour les broches de contrôle physiques. Si la direction n'est pas précisée, le groupe données sera par défault en sortie. Par exemple :

```
loadrt hal_parport cfg="0x278 0x378 in 0x20A0 out"
```

Cet exemple installe les pilotes pour un port 0x0278, avec les broches 2 à 9 en sorties (par défaut, puisque ni " in ", ni "out" n'est spécifié), un port 0x0378, avec les broches 2 à 9 en entrées et un port 0x20A0, avec les broches 2 à 9 explicitement spécifiées en sorties. Notez que vous devez connaître l'adresse de base des ports parallèles pour configurer correctement les pilotes. Pour les ports sur bus ISA, ce n'est généralement pas un problème, étant donné que les ports sont presque toujours à une adresse "bien connue", comme 0278 ou 0378 qui sont typiquement configurées dans le BIOS. Les adresses des cartes sur bus PCI sont habituellement trouvées avec "lspci -v" dans une ligne "I/O ports", ou dans un message du kernel après l'exécution de "`sudo modprobe -a parport_pc`". Il n'y a pas d'adresse par défaut, si `<config-string>` ne contient pas au moins une adresse, un message d'erreur s'affichera.

8.1.2 Pins

- `(BIT) parport.<portnum>.pin-<pinnum>-out` – Pilote une pin de sortie physique.
- `(BIT) parport.<portnum>.pin-<pinnum>-in` – Suit une pin d'entrée physique.
- `(BIT) parport.<portnum>.pin-<pinnum>-in-not` – Suit une pin d'entrée physique, mais inversée.

Pour chaque pin, `<portnum>` est le numéro du port et `<pinnum>` est le numéro de la broche physique du connecteur DB-25.

Pour chaque broche de sortie physique, le pilote crée une simple pin de HAL, par exemple `parport.0.pin-14-ou`
Les pins 2 jusqu'à 9 font partie du groupe donnée est sont des pins de sortie si le port est défini comme un port de sortie (par défaut en sortie). Les broches 1, 14, 16 et 17 sont des sorties dans tous les modes. Ces pin de HAL contrôlent l'état des pins physiques correspondantes.

Pour chaque pin d'entrée physique, le pilote crée deux pins de HAL, par exemple `parport.0.pin-12-in` et `parport.0.pin-12-in-not`. Les pins 10, 11, 12, 13 et 15 sont toujours des sorties. Les pins 2 jusqu'à 9 sont des pins d'entrée seulement si le port est défini comme un port d'entrée. Une pin de HAL `-in` est VRAIE si la pin physique est haute et FAUSSE si la pin physique est basse. Une pin de HAL `-in-not` est invertsée, elle est FAUSSE si la pin physique est haute. En connectant un signal à l'une ou l'autre, l'utilisateur peut décider de l'état de l'entrée. En mode "x", les pins 1, 14, 16 et 17 sont également des pins d'entrée.

8.1.3 Paramètres

- `(BIT) parport.<portnum>.pin-<pinnum>-out-invert` – Inverse une pin de sortie.
- `(BIT) parport.<portnum>.pin-<pinnum>-out-reset` (seulement les pins 2..9) – VRAIE si cette pin doit être réinitialisée quand la fonction de réinitialisation est exécutée.
- `(U32) parport.<portnum>.reset-time` – Le temps (en nanosecondes) entre le moment ou la broche est écrite et le moment ou elle est réititialisée par les fonctions de réinitialisation de HAL.

Le paramètre `-invert` détermine si une pin de sortie est active haute ou active basse. Si `-invert` est FAUX, mettre la pin HAL `-out` VRAIE placera la pin physique à l'état haut et mettre la pin HAL FAUSSE placera la pin physique à l'état bas. Si `-invert` est VRAI, mettre la pin HAL `-out` VRAIE va mettre la pin physique à l'état bas.

Si `-reset` est VRAI, la fonction de réinitialisation va passer la pin à la valeur de `-out-invert`. Ceci peut être utilisé en conjonction avec "stepgen `doublefreq`" pour produire un pas par période.

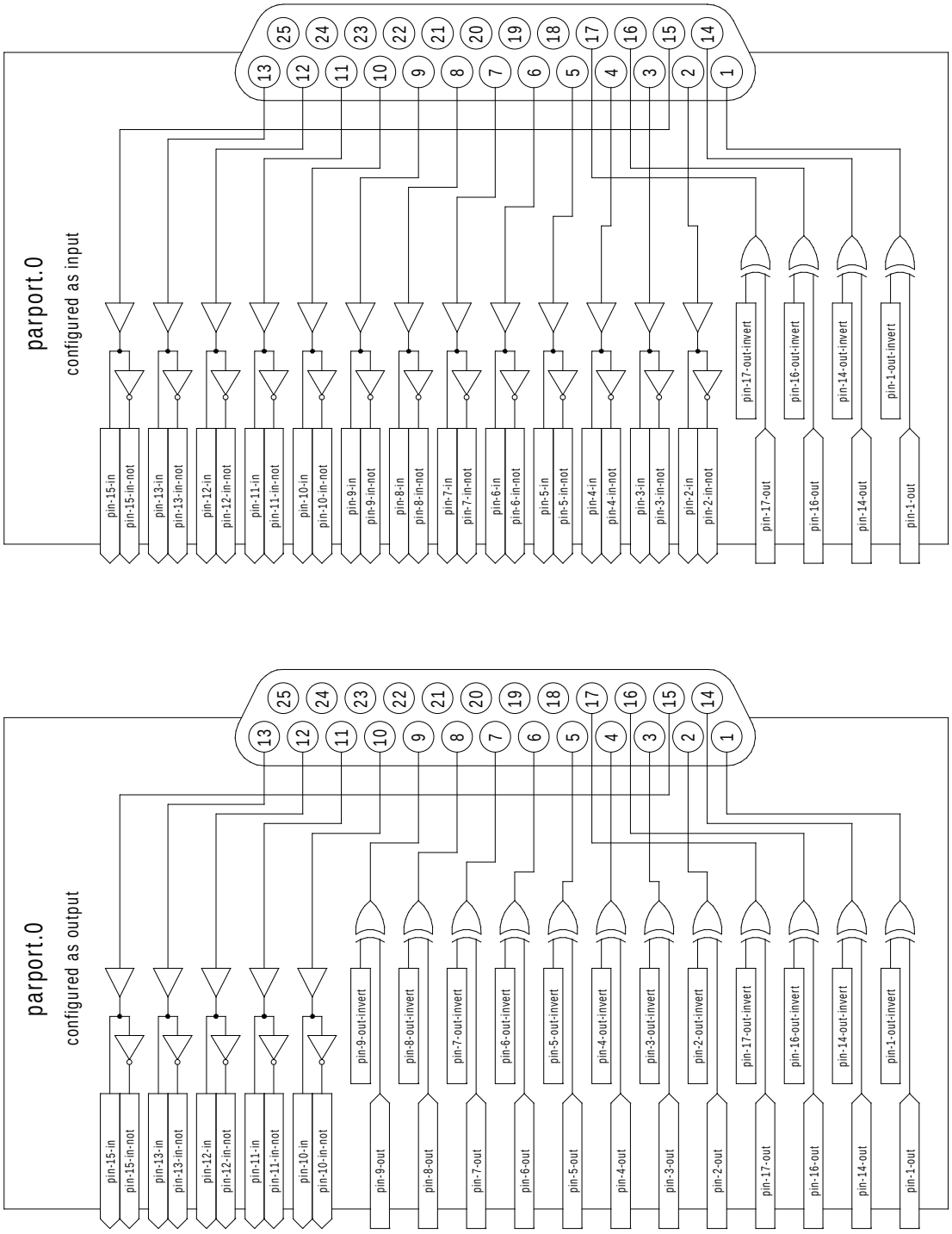


FIG. 8.1 – Diagrammes blocs de Parport

8.1.4 Fonctions

- (FUNCT) parport.<portnum>.read – Lit les pins physiques du port <portnum> et met à jour les pins de HAL -in et -in-not.
- (FUNCT) parport.read-all – Lit les pins physiques de tous les ports et met à jour les pins de HAL -in et -in-not.
- (FUNCT) parport.<portnum>.write – Lit les pins de HAL -out du port <portnum> et met à jour les pins de sortie physiques correspondantes.
- (FUNCT) parport.write-all – Lit les pins de HAL -out de tous les ports et met à jour toutes les pins de sortie physiques.
- (FUNCT) parport.<portnum>.reset – Attends que le délai de mise à jour soit écoulé depuis le dernière écriture, remet à jour les pins aux valeurs indiquées par -out-invert et les paramètres de -out-invert. La réinitialisation doit être plus tard dans le même thread que l'écriture.

Les différentes fonctions sont prévues pour les situations où un port doit être mis à jour dans un thread très rapide, mais d'autres ports peuvent être mis à jour dans un thread plus lent pour gagner du temps CPU. Ce n'est probablement pas une bonne idée d'utiliser en même temps, les fonctions -all et une fonction individuelle.

8.1.5 Problèmes courants

Si le chargement du module le message suivant :

```
insmod : error inserting '/home/jepler/emc2/rtlib/hal_parport.ko' :
-1 Device or resource busy
```

s'assurer que le noyau du kernel standard, parport_pc, n'est pas chargé[2] et qu'aucun périphérique dans le système ne revendique les ports concernés.

SI le module est chargé mais ne semble pas fonctionner, l'adresse du port est incorrecte ou le module probe_parport est requis.

8.2 probe_parport

Dans les PC récents, le port parallèle peut exiger une configuration plug and play (PNP) avant d'être utilisable. Le module probe_parport effectue la configuration de tous les ports PNP présents et devrait être chargé avant hal_parport. Sur les machines sans ports PNP, il peut être chargé mais n'a aucun effet.

8.2.1 Installation

```
loadrt probe_parport
loadrt hal_parport ...
```

Si le kernel Linux affiche un message similaire à :

```
parport : PnPBIOS parport detected.
```

quand le module parport_pc est chargé, avec la commande : sudo modprobe -a parport_pc; sudo rmmod parport_pc, l'utilisation de ce module sera probablement nécessaire.

.

Les modules commerciaux ci-dessous pouront êtres traduits en français sur demande.

The commercial modules below could be translated into French on request.

.

[2] In the emc packages for Ubuntu, the file /etc/modprobe.d/emc2 generally prevents parport_pc from being automatically loaded.

8.3 AX5214H

The Axiom Measurement & Control AX5214H is a 48 channel digital I/O board. It plugs into an ISA bus, and resembles a pair of 8255 chips.[3]

8.3.1 Installing

```
loadrt hal_ax5214h cfg="<config-string>"
```

The config string consists of a hex port address, followed by an 8 character string of "I" and "O" which sets groups of pins as inputs and outputs. The first two character set the direction of the first two 8 bit blocks of pins (0-7 and 8-15). The next two set blocks of 4 pins (16-19 and 20-23). The pattern then repeats, two more blocks of 8 bits (24-31 and 32-39) and two blocks of 4 bits (40-43 and 44-47). If more than one board is installed, the data for the second board follows the first. As an example, the string **"0x220 IIIOIIOO 0x300 OIOOIOIO"** installs drivers for two boards. The first board is at address 0x220, and has 36 inputs (0-19 and 24-39) and 12 outputs (20-23 and 40-47). The second board is at address 0x300, and has 20 inputs (8-15, 24-31, and 40-43) and 28 outputs (0-7. 16-23, 32-39, and 44-47). Up to 8 boards may be used in one system.

8.3.2 Pins

- (BIT) ax5214.<boardnum>.out-<pinnum> – Drives a physical output pin.
- (BIT) ax5214.<boardnum>.in-<pinnum> – Tracks a physical input pin.
- (BIT) ax5214.<boardnum>.in-<pinnum>-not – Tracks a physical input pin, inverted.

For each pin, <boardnum> is the board number (starts at zero), and <pinnum> is the I/O channel number (0 to 47).

Note that the driver assumes active LOW signals. This is so that modules such as OPTO-22 will work correctly (TRUE means output ON, or input energized). If the signals are being used directly without buffering or isolation the inversion needs to be accounted for. The in- HAL pin is TRUE if the physical pin is low (OPTO-22 module energized), and FALSE if the physical pin is high (OPTO-22 module off). The in-<pinnum>-not HAL pin is inverted – it is FALSE if the physical pin is low (OPTO-22 module energized). By connecting a signal to one or the other, the user can determine the state of the input.

8.3.3 Parameters

- (BIT) ax5214.<boardnum>.out-<pinnum>-invert – Inverts an output pin.

The -invert parameter determines whether an output pin is active high or active low. If -invert is FALSE, setting the HAL out- pin TRUE drives the physical pin low, turning ON an attached OPTO-22 module, and FALSE drives it high, turning OFF the OPTO-22 module. If -invert is TRUE, then setting the HAL out- pin TRUE will drive the physical pin high and turn the module OFF.

8.3.4 Functions

- (FUNCT) ax5214.<boardnum>.read – Reads all digital inputs on one board.
- (FUNCT) ax5214.<boardnum>.write – Writes all digital outputs on one board.

[3] In fact it may be a pair of 8255 chips, but I'm not sure. If/when someone starts a driver for an 8255 they should look at the ax5214 code, much of the work is already done.

8.4 Servo-To-Go

The Servo-To-Go is one of the first PC motion control cards[4] supported by EMC. It is an ISA card and it exists in different flavours (all supported by this driver). The board includes up to 8 channels of quadrature encoder input, 8 channels of analog input and output, 32 bits digital I/O, an interval timer with interrupt and a watchdog.

8.4.1 Installing :

```
loadrt hal_stg [base=<address>] [num_chan=<nr>] [dio="<dio-string>"] [model=<model>]
```

The base address field is optional ; if it's not provided the driver attempts to autodetect the board. The num_chan field is used to specify the number of channels available on the card, if not used the 8 axis version is assumed. The digital inputs/outputs configuration is determined by a config string passed to insmod when loading the module. The format consists of a four character string that sets the direction of each group of pins. Each character of the direction string is either "I" or "O". The first character sets the direction of port A (Port A - DIO.0-7), the next sets port B (Port B - DIO.8-15), the next sets port C (Port C - DIO.16-23), and the fourth sets port D (Port D - DIO.24-31). The model field can be used in case the driver doesn't autodetect the right card version[5]. For example :

```
loadrt hal_stg base=0x300 num_chan=4 dio="IOIO"
```

This example installs the stg driver for a card found at the base address of 0x300, 4 channels of encoder feedback, DAC's and ADC's, along with 32 bits of I/O configured like this : the first 8 (Port A) configured as Input, the next 8 (Port B) configured as Output, the next 8 (Port C) configured as Input, and the last 8 (Port D) configured as Output

```
loadrt hal_stg
```

This example installs the driver and attempts to autodetect the board address and board model, it installs 8 axes by default along with a standard I/O setup : Port A & B configured as Input, Port C & D configured as Output.

8.4.2 Pins

- (s32) stg.<channel>.counts – Tracks the counted encoder ticks.
- (FLOAT) stg.<channel>.position – Outputs a converted position.
- (FLOAT) stg.<channel>.dac-value – Drives the voltage for the corresponding DAC.
- (FLOAT) stg.<channel>.adc-value – Tracks the measured voltage from the corresponding ADC.
- (BIT) stg.in-<pinnum> – Tracks a physical input pin.
- (BIT) stg.in-<pinnum>-not – Tracks a physical input pin, but inverted.
- (BIT) stg.out-<pinnum> – Drives a physical output pin

For each pin, <channel> is the axis number, and <pinnum> is the logic pin number of the STG[6].

The in- HAL pin is TRUE if the physical pin is high, and FALSE if the physical pin is low. The in-<pinnum>-not HAL pin is inverted – it is FALSE if the physical pin is high. By connecting a signal to one or the other, the user can determine the state of the input.

[4] a motion control card usually is a board containing devices to control one or more axes (the control devices are usually DAC's to set an analog voltage, encoder counting chips for feedback, etc.)

[5] hint : after starting up the driver, 'dmesg' can be consulted for messages relevant to the driver (e.g. autodetected version number and base address)

[6] if IIOO is defined, there are 16 input pins (in-00 .. in-15) and 16 output pins (out-00 .. out-15), and they correspond to PORTs ABCD (in-00 is PORTA.0, out-15 is PORTD.7)

8.4.3 Parameters

- (FLOAT) stg.<channel>.position-scale – The number of counts / user unit (to convert from counts to units).
- (FLOAT) stg.<channel>.dac-offset – Sets the offset for the corresponding DAC.
- (FLOAT) stg.<channel>.dac-gain – Sets the gain of the corresponding DAC.
- (FLOAT) stg.<channel>.adc-offset – Sets the offset of the corresponding ADC.
- (FLOAT) stg.<channel>.adc-gain – Sets the gain of the corresponding ADC.
- (BIT) stg.out-<pinnum>-invert – Inverts an output pin.

The -invert parameter determines whether an output pin is active high or active low. If -invert is FALSE, setting the HAL out- pin TRUE drives the physical pin high, and FALSE drives it low. If -invert is TRUE, then setting the HAL out- pin TRUE will drive the physical pin low.

8.4.4 Functions

- (FUNCT) stg.capture-position – Reads the encoder counters from the axis <channel>.
- (FUNCT) stg.write-dacs – Writes the voltages to the DACs.
- (FUNCT) stg.read-adcs – Reads the voltages from the ADCs.
- (FUNCT) stg.di-read – Reads physical in- pins of all ports and updates all HAL in- and in-<pinnum>-not pins.
- (FUNCT) stg.do-write – Reads all HAL out- pins and updates all physical output pins.

8.5 Mesa Electronics m5i20 “Anything I/O Card”

The Mesa Electronics m5i20 card consists of an FPGA that can be loaded with a wide variety of configurations, and has 72 pins that leave the PC. The assignment of the pins depends on the FPGA configuration. Currently there is a HAL driver for the “4 axis host based motion control” configuration, and this FPGA configurations is also provided with EMC2. It provides 8 encoder counters, 4 PWM outputs (normally used as DACs) and up to 48 digital I/O channels, 32 inputs and 16 outputs.[7]

Installing :

```
loadrt hal_m5i20 [loadFpga=1|0] [dacRate=<rate>]
```

If **loadFpga** is 1 (the default) the driver will load the FPGA configuration on startup. If it is 0, the driver assumes the configuration is already loaded. **dacRate** sets the carrier frequency for the PWM outputs, in Hz. The default is 32000, for 32KHz PWM. Valid values are from 1 to 32226. The driver prints some useful debugging message to the kernel log, which can be viewed with **dmesg**.

Up to 4 boards may be used in one system.

8.5.1 Pins

In the following pins, parameters, and functions, <board> is the board ID. According to the naming conventions the first board should always have an ID of zero, however this driver uses the PCI board ID, so it may be non-zero even if there is only one board.

- (s32) m5i20.<board>.enc-<channel>-count – Encoder position, in counts.
- (FLOAT) m5i20.<board>.enc-<channel>-position – Encoder position, in user units.
- (BIT) m5i20.<board>.enc-<channel>-index – Current status of index pulse input ?
- (BIT) m5i20.<board>.enc-<channel>-index-enable – when TRUE, and an index pulse appears on the encoder input, reset counter to zero and clear index-enable.

[7] Ideally the encoders, “DACs”, and digital I/O would comply with the canonical interfaces defined earlier, but they don’t. Fixing that is on the things-to-do list.

– (BIT) m5i20.<board>.enc-<channel>-reset – When true, counter is forced to zero.
– (BIT) m5i20.<board>.dac-<channel>-enable – Enables DAC if true. DAC outputs zero volts if false ?
– (FLOAT) m5i20.<board>.dac-<channel>-value – Analog output value for PWM "DAC" (in user units, see -scale and -offset)
– (BIT) m5i20.<board>.in-<channel> – State of digital input pin, see canonical digital input.
– (BIT) m5i20.<board>.in-<channel>-not – Inverted state of digital input pin, see canonical digital input.
– (BIT) m5i20.<board>.out-<channel> – Value to be written to digital output, see canonical digital output.
– (BIT) m5i20.<board>.estop-in – Dedicated estop input, more details needed.
– (BIT) m5i20.<board>.estop-in-not – Inverted state of dedicated estop input.
– (BIT) m5i20.<board>.watchdog-reset – Bidirectional, - Set TRUE to reset watchdog once, is automatically cleared. If bit value 16 is set in watchdog-control then this value is not used, and the hardware watchdog is cleared every time the dac-write function is executed.

8.5.2 Parameters

– (FLOAT) m5i20.<board>.enc-<channel>-scale – The number of counts / user unit (to convert from counts to units).
– (FLOAT) m5i20.<board>.dac-<channel>-offset – Sets the DAC offset.
– (FLOAT) m5i20.<board>.dac-<channel>-gain – Sets the DAC gain (scaling).
– (BIT) m5i20.<board>.dac-<channel>-interlaced – Sets the DAC to interlaced mode. Use this mode if you are filtering the PWM to generate an anaolg voltage.[8]
– (BIT) m5i20.<board>.out-<channel>-invert – Inverts a digital output, see canonical digital output.
– (U32) m5i20.<board>.watchdog-control – Configures the watchdog. The value may be a bitwise OR of the following values :

Bit #	Value	Meaning
0	1	Watchdog is enabled
1	2	Watchdog is automatically reset by DAC writes (the HAL dac-write function)

Typically, the useful values are 0 (watchdog disabled) or 3 (watchdog enabled, cleared by dac-write).
– (U32) m5i20.<board>.led-view – Maps some of the I/O to onboard LEDs. See table below.

8.5.3 Functions

– (FUNCT) m5i20.<board>.encoder-read – Reads all encoder counters.
– (FUNCT) m5i20.<board>.digital-in-read – Reads digital inputs.
– (FUNCT) m5i20.<board>.dac-write – Writes the voltages (PWM duty cycles) to the "DACs".
– (FUNCT) m5i20.<board>.digital-out-write – Writes digital outputs.
– (FUNCT) m5i20.<board>.misc-update – Writes watchdog timer configuration to hardware. Resets watchdog timer. Updates E-stop pin (more info needed). Updates onboard LEDs.

8.5.4 Connector pinout

The Hostmot-4 FPGA configuration has the following pinout. There are three 50-pin ribbon cable connectors on the card : P2, P3, and P4. There are also 8 status LEDs.

[8] With normal 10 bit PWM, 50% duty cycle would be 512 cycles on and 512 cycles off = ca 30 kHz with 33 MHz reference counter. With fully interleaved PWM this would be 1 cycle on, 1 cycle off for 1024 cycles (16.66 MHz if the PWM reference counter runs at 33 MHz) = much easier to filter. The 5I20 configuration interlace is somewhat between non and fully interlaced (to make it easy to filter but not have as many transistions as fully interleaved).

8.5.4.1 Connector P2

m5i20 card connector P2	Function/HAL-pin
1	enc-01 A input
3	enc-01 B input
5	enc-00 A input
7	enc-00 B input
9	enc-01 index input
11	enc-00 index input
13	dac-01 output
15	dac-00 output
17	DIR output for dac-01
19	DIR output for dac-00
21	dac-01-enable output
23	dac-00-enable output
25	enc-03 B input
27	enc-03 A input
29	enc-02 B input
31	enc-02 A input
33	enc-03 index input
35	enc-02 index input
37	dac-03 output
39	dac-02 output
41	DIR output for dac-03
43	DIR output for dac-02
45	dac-03-enable output
47	dac-02-enable output
49	Power +5 V (or +3.3V?)
all even pins	Ground

8.5.4.2 Connector P3

Encoder counters 4 - 7 work simultaneously with in-00 to in-11.

If you are using in-00 to in-11 as general purpose IO then reading enc-<4-7> will produce some random junk number.

m5i20 card connector P3	Function/HAL-pin	Secondary Function/HAL-pin
1	in-00	enc-04 A input
3	in-01	enc-04 B input
5	in-02	enc-04 index input
7	in-03	enc-05 A input
9	in-04	enc-05 B input
11	in-05	enc-05 index input
13	in-06	enc-06 A input
15	in-07	enc-06 B input
17	in-08	enc-06 index input
19	in-09	enc-07 A input
21	in-10	enc-07 B input
23	in-11	enc-07 index input
25	in-12	
27	in-13	
29	in-14	
31	in-15	
33	out-00	
35	out-01	
37	out-02	
39	out-03	
41	out-04	
43	out-05	
45	out-06	
47	out-07	
49	Power +5 V (or +3.3V?)	
all even pins	Ground	

8.5.4.3 Connector P4

The index mask masks the index input of the encoder so that the encoder index can be combined with a mechanical switch or opto detector to clear or latch the encoder counter only when the mask input bit is in proper state (selected by mask polarity bit) and encoder index occurs. This is useful for homing. The behaviour of these pins is controlled by the Counter Control Register (CCR), however there is currently no function in the driver to change the CCR. See REGMAP4[9] for a description of the CCR.

[9] emc2/src/hal/drivers/m5i20/REGMAP4E

m5i20 card connector P4	Function/HAL-pin	Secondary Function/HAL-pin
1	in-16	enc-00 index mask
3	in-17	enc-01 index mask
5	in-18	enc-02 index mask
7	in-19	enc-03 index mask
9	in-20	
11	in-21	
13	in-22	
15	in-23	
17	in-24	enc-04 index mask
19	in-25	enc-05 index mask
21	in-26	enc-06 index mask
23	in-27	enc-07 index mask
25	in-28	
27	in-29	
29	in-30	
31	in-31	
33	out-08	
35	out-09	
37	out-10	
39	out-11	
41	out-12	
43	out-13	
45	out-14	
47	out-15	
49	Power +5 V (or +3.3V?)	
all even pins	Ground	

8.5.4.4 LEDs

The status LEDs will monitor one motion channel set by the `m5i20.<board>.led-view` parameter. A call to `m5i20.<board>.misc-update` is required to update the viewed channel.

LED name	Output
LED0	IRQLatch?
LED1	enc-<channel> A
LED2	enc-<channel> B
LED3	enc-<channel> index
LED4	dac-<channel> DIR
LED5	dac-<channel>
LED6	dac-<channel>-enable
LED7	watchdog timeout?

8.6 Vital Systems Motenc-100 and Motenc-LITE

The Vital Systems Motenc-100 and Motenc-LITE are 8- and 4-channel servo control boards. The Motenc-100 provides 8 quadrature encoder counters, 8 analog inputs, 8 analog outputs, 64 (68?) digital inputs, and 32 digital outputs. The Motenc-LITE has only 4 encoder counters, 32 digital inputs and 16 digital outputs, but it still has 8 analog inputs and 8 analog outputs. The driver automatically identifies the installed board and exports the appropriate HAL objects.[10]

Installing :

[10] Ideally the encoders, DACs, ADCs, and digital I/O would comply with the canonical interfaces defined earlier, but they don't. Fixing that is on the things-to-do list.

```
loadrt hal_motenc
```

During loading (or attempted loading) the driver prints some usefull debugging message to the kernel log, which can be viewed with **dmesg**.

Up to 4 boards may be used in one system.

8.6.1 Pins

In the following pins, parameters, and functions, <board> is the board ID. According to the naming conventions the first board should always have an ID of zero. However this driver sets the ID based on a pair of jumpers on the baord, so it may be non-zero even if there is only one board.

- (S32) motenc.<board>.enc-<channel>-count – Encoder position, in counts.
- (FLOAT) motenc.<board>.enc-<channel>-position – Encoder position, in user units.
- (BIT) motenc.<board>.enc-<channel>-index – Current status of index pulse input.
- (BIT) motenc.<board>.enc-<channel>-idx-latch – Driver sets this pin true when it latches an index pulse (enabled by latch-index). Cleared by clearing latch-index.
- (BIT) motenc.<board>.enc-<channel>-latch-index – If this pin is true, the driver will reset the counter on the next index pulse.
- (BIT) motenc.<board>.enc-<channel>-reset-count – If this pin is true, the counter will immediately be reset to zero, and the pin will be cleared.
- (FLOAT) motenc.<board>.dac-<channel>-value – Analog output value for DAC (in user units, see -gain and -offset)
- (FLOAT) motenc.<board>.adc-<channel>-value – Analog input value read by ADC (in user units, see -gain and -offset)
- (BIT) motenc.<board>.in-<channel> – State of digital input pin, see canonical digital input.
- (BIT) motenc.<board>.in-<channel>-not – Inverted state of digital input pin, see canonical digital input.
- (BIT) motenc.<board>.out-<channel> – Value to be written to digital output, seen canonical digital output.
- (BIT) motenc.<board>.estop-in – Dedicated estop input, more details needed.
- (BIT) motenc.<board>.estop-in-not – Inverted state of dedicated estop input.
- (BIT) motenc.<board>.watchdog-reset – Bidirectional, - Set TRUE to reset watchdog once, is automatically cleared.

8.6.2 Parameters

- (FLOAT) motenc.<board>.enc-<channel>-scale – The number of counts / user unit (to convert from counts to units).
- (FLOAT) motenc.<board>.dac-<channel>-offset – Sets the DAC offset.
- (FLOAT) motenc.<board>.dac-<channel>-gain – Sets the DAC gain (scaling).
- (FLOAT) motenc.<board>.adc-<channel>-offset – Sets the ADC offset.
- (FLOAT) motenc.<board>.adc-<channel>-gain – Sets the ADC gain (scaling).
- (BIT) motenc.<board>.out-<channel>-invert – Inverts a digital output, see canonical digital output.
- (U32) motenc.<board>.watchdog-control – Configures the watchdog. The value may be a bitwise OR of the following values :

Bit #	Value	Meaning
0	1	Timeout is 16ms if set, 8ms if unset
2	4	Watchdog is enabled
4	16	Watchdog is automatically reset by DAC writes (the HAL dac-write function)

Typically, the useful values are 0 (watchdog disabled) or 20 (8ms watchdog enabled, cleared by dac-write).

- (U32) motenc.<board>.led-view – Maps some of the I/O to onboard LEDs ?

8.6.3 Functions

- (FUNCT) motenc.<board>.encoder-read – Reads all encoder counters.
- (FUNCT) motenc.<board>.adc-read – Reads the analog-to-digital converters.
- (FUNCT) motenc.<board>.digital-in-read – Reads digital inputs.
- (FUNCT) motenc.<board>.dac-write – Writes the voltages to the DACs.
- (FUNCT) motenc.<board>.digital-out-write – Writes digital outputs.
- (FUNCT) motenc.<board>.misc-update – Updates misc stuff.

8.7 Pico Systems PPMC (Parallel Port Motion Control)

Pico Systems has a family of boards for doing servo, stepper, and pwm control. The boards connect to the PC through a parallel port working in EPP mode. Although most users connect one board to a parallel port, in theory any mix of up to 8 or 16 boards can be used on a single parport. One driver serves all types of boards. The final mix of I/O depends on the connected board(s). The driver doesn't distinguish between boards, it simply numbers I/O channels (encoders, etc) starting from 0 on the first card.

Installing :

```
loadrt hal_ppmc port_addr=<addr1>[,<addr2>[,<addr3>...]]
```

The **port_addr** parameter tells the driver what parallel port(s) to check. By default, **<addr1>** is 0x0378, and **<addr2>** and following are not used. The driver searches the entire address space of the enhanced parallel port(s) at **port_addr**, looking for any board(s) in the PPMC family. It then exports HAL pins for whatever it finds. During loading (or attempted loading) the driver prints some usefull debugging message to the kernel log, which can be viewed with **dmesg**.

Up to 3 parport busses may be used, and each bus may have up to 8 devices on it.

8.7.1 Pins

In the following pins, parameters, and functions, <board> is the board ID. According to the naming conventions the first board should always have an ID of zero. However this driver sets the ID based on a pair of jumpers on the baord, so it may be non-zero even if there is only one board.

- (S32) ppmc.<port>.encoder.<channel>.count – Encoder position, in counts.
- (S32) ppmc.<port>.encoder.<channel>.delta – Change in counts since last read.
- (FLOAT) ppmc.<port>.encoder.<channel>.position – Encoder position, in user units.
- (BIT) ppmc.<port>.encoder.<channel>.index – Something to do with index pulse.[11]
- (BIT) ppmc.<port>.pwm.<channel>.enable – Enables a PWM generator.
- (FLOAT) ppmc.<port>.pwm.<channel>.value – Value which determines the duty cycle of the PWM waveforms. The value is divided by pwm.<channel>.scale, and if the result is 0.6 the duty cycle will be 60%, and so on. Negative values result in the duty cycle being based on the absolute value, and the direction pin is set to indicate negative.
- (BIT) ppmc.<port>.stepgen.<channel>.enable – Enables a step pulse generator.
- (FLOAT) ppmc.<port>.stepgen.<channel>.velocity – Value which determines the step frequency. The value is multiplied by stepgen.<channel>.scale, and the result is the frequency in steps per second. Negative values result in the frequency being based on the absolute value, and the direction pin is set to indicate negative.
- (BIT) ppmc.<port>.in-<channel> – State of digital input pin, see canonical digital input.
- (BIT) ppmc.<port>.in.<channel>.not – Inverted state of digital input pin, see canonical digital input.
- (BIT) ppmc.<port>.out-<channel> – Value to be written to digital output, seen canonical digital output.

[11] Index handling does _not_ comply with the canonical encoder interface, and should be changed.

8.7.2 Parameters

- (FLOAT) ppmc.<port>.enc.<channel>.scale – The number of counts / user unit (to convert from counts to units).
- (FLOAT) ppmc.<port>.pwm.<channel-range>.freq – The PWM carrier frequency, in Hz. Applies to a group of four consecutive PWM generators, as indicated by <channel-range>. Minimum is 153Hz, maximum is 500KHz.
- (FLOAT) ppmc.<port>.pwm.<channel>.scale – Scaling for PWM generator. If scale is X, then the duty cycle will be 100% when the value pin is X (or -X).
- (FLOAT) ppmc.<port>.pwm.<channel>.max-dc – Maximum duty cycle, from 0.0 to 1.0.
- (FLOAT) ppmc.<port>.pwm.<channel>.min-dc – Minimum duty cycle, from 0.0 to 1.0.
- (FLOAT) ppmc.<port>.pwm.<channel>.duty-cycle – Actual duty cycle (used mostly for troubleshooting.)
- (BIT) ppmc.<port>.pwm.<channel>.bootstrap – If true, the PWM generator will generate a short sequence of pulses of both polarities when it is enabled, to charge the bootstrap capacators used on some MOSFET gate drivers.
- (U32) ppmc.<port>.stepgen.<channel-range>.setup-time – Sets minimum time between direction change and step pulse, in units of 100nS. Applies to a group fof four consecutive PWM generators, as indicated by <channel-range>.
- (U32) ppmc.<port>.stepgen.<channel-range>.pulse-width – Sets width of step pulses, in units of 100nS. Applies to a group fof four consecutive PWM generators, as indicated by <channel-range>.
- (U32) ppmc.<port>.stepgen.<channel-range>.pulse-space-min – Sets minimum time between pulses, in units of 100nS. The maximum step rate is 1/(100nS * (pulse-width + pulse-space-min)). Applies to a group fof four consecutive PWM generators, as indicated by <channel-range>.
- (FLOAT) ppmc.<port>.stepgen.<channel>.scale – Scaling for step pulse generator. The step frequency in Hz is the absolute value of velocity * scale.
- (FLOAT) ppmc.<port>.stepgen.<channel>.max-vel – The maximum value for velocity. Commands greater than max-vel will be clamped. Also applies to negative values. (The absolute value is clamped.)
- (FLOAT) ppmc.<port>.stepgen.<channel>.frequency – Actual step pulse frequency in Hz (used mostly for troubleshooting.)
- (BIT) ppmc.<port>.out.<channel>.invert – Inverts a digital output, see canonical digital output.

8.7.3 Functions

- (FUNCT) ppmc.<port>.read – Reads all inputs (digital inputs and encoder counters) on one port.
- (FUNCT) ppmc.<port>.write – Writes all outputs (digital outputs, stepgens, PWMs) on one port.

8.8 Pluto-P : generalities

The Pluto-P is an inexpensive ($60) FPGA board featuring the ACEX1K chip from Altera.

8.8.1 Requirements

1. A Pluto-P board
2. An EPP-compatible parallel port, configured for EPP mode in the system BIOS

8.8.2 Connectors

- The Pluto-P board is shipped with the left connector presoldered, with the key in the indicated position. The other connectors are unpopulated. There does not seem to be a standard 12-pin IDC connector, but some of the pins of a 16P connector can hang off the board next to QA3/QZ3.

– The bottom and right connectors are on the same .1" grid, but the left connector is not. If OUT2. . . OUT9 are not required, a single IDC connector can span the bottom connector and the bottom two rows of the right connector.

8.8.3 Physical Pins

– Read the ACEX1K datasheet for information about input and output voltage thresholds. The pins are all configured in "LVTTL/LVCMOS" mode and are generally compatible with 5V TTL logic.
– Before configuration and after properly exiting emc2, all Pluto-P pins are tristated with weak pull-ups (20kΩ min, 50kΩ max). If the watchdog timer is enabled (the default), these pins are also tristated after an interruption of communication between emc2 and the board. The watchdog timer takes approximately 6.5ms to activate. However, software bugs in the pluto_servo firmware or emc2 can leave the Pluto-P pins in an undefined state.
– In pwm+dir mode, by default dir is HIGH for negative values and LOW for positive values. To select HIGH for positive values and LOW for negative values, set the corresponding dout-NN-invert parameter TRUE to invert the signal.
– The index input is triggered on the rising edge. Initial testing has shown that the QZx inputs are particularly noise sensitive, due to being polled every 25ns. Digital filtering has been added to filter pulses shorter than 175ns (seven polling times). Additional external filtering on all input pins, such as a Schmitt buffer or inverter, RC filter, or differential receiver (if applicable) is recommended.
– The IN1. . . IN7 pins have 22-ohm series resistors to their associated FPGA pins. No other pins have any sort of protection for out-of-spec voltages or currents. It is up to the integrator to add appropriate isolation and protection. Traditional parallel port optoisolator boards do not work with pluto_servo due to the bidirectional nature of the EPP protocol.

8.8.4 LED

– When the device is unprogrammed, the LED glows faintly. When the device is programmed, the LED glows according to the duty cycle of PWM0 (**LED** = **UP0** *xor* **DOWN0**) or STEPGEN0 (**LED** = **STEP0** *xor* **DIR0**).

8.8.5 Power

– A small amount of current may be drawn from VCC. The available current depends on the unregulated DC input to the board. Alternately, regulated +3.3VDC may be supplied to the FPGA through these VCC pins. The required current is not yet known, but is probably around 50mA plus I/O current.
– The regulator on the Pluto-P board is a low-dropout type. Supplying 5V at the power jack will allow the regulator to work properly.

8.8.6 PC interface

– At present, only a single pluto_servo or pluto_step board is supported. At present there is no provision for multiple boards on one parallel port (because all boards reside at the same EPP address) but supporting one board per parallel port should be possible.

8.8.7 Rebuilding the FPGA firmware

The `src/hal/drivers/pluto_servo_firmware/` and `src/hal/drivers/pluto_step_firmware/` subdirectories contain the Verilog source code plus additional files used by Quartus for the FPGA firmwares. Altera's Quartus II software is required to rebuild the FPGA firmware. To rebuild the

firmware from the .hdl and other source files, open the `.qpf` file and press CTRL-L. Then, recompile emc2.

Like the HAL hardware driver, the FPGA firmware is licensed under the terms of the GNU General Public License.

The gratis version of Quartus II runs only on Microsoft Windows, although there is apparently a paid version that runs on Linux.

8.8.8 For more information

The Pluto-P board may be ordered from http://www.knjn.com/ShopBoards_Parallel.html (US based, international shipping is available). Some additional information about it is available from http://www.fpga4fun.com/board_pluto-P.html and from the developer's blog http://emergent.unpy.net/01165081407.

8.9 pluto-servo : Hardware PWM and quadrature counting

The pluto_servo system is suitable for control of a 4-axis CNC mill with servo motors, a 3-axis mill with PWM spindle control, a lathe with spindle encoder, etc. The large number of inputs allows a full set of limit switches.

This driver features :

- 4 quadrature channels with 40MHz sample rate. The counters operate in "4x" mode. The maximum useful quadrature rate is 8191 counts per emc2 servo cycle, or about 8MHz for EMC2's default 1ms servo rate.
- 4 PWM channels, "up/down" or "pwm+dir" style. 4095 duty cycles from -100% to +100%, including 0%. The PWM period is approximately 19.5kHz (40MHz / 2047). A PDM-like mode is also available.
- 18 digital outputs : 10 dedicated, 8 shared with PWM functions. (Example : A lathe with unidirectional PWM spindle control may use 13 total digital outputs)
- 20 digital inputs : 8 dedicated, 12 shared with Quadrature functions. (Example : A lathe with index pulse only on the spindle may use 13 total digital inputs)
- EPP communication with the PC. The EPP communication typically takes around 100uS on machines tested so far, enabling servo rates above 1kHz.

8.9.1 Pinout

UPx The "up" (up/down mode) or "pwm" (pwm+direction mode) signal from PWM generator X. May be used as a digital output if the corresponding PWM channel is unused, or the output on the channel is always negative. The corresponding digital output invert may be set to TRUE to make UPx active low rather than active high.

DNx The "down" (up/down mode) or "direction" (pwm+direction mode) signal from PWM generator X. May be used as a digital output if the corresponding PWM channel is unused, or the output on the channel is never negative. The corresponding digital ouput invert may be set to TRUE to make DNx active low rather than active high.

QAx, QBx The A and B signals for Quadrature counter X. May be used as a digital input if the corresponding quadrature channel is unused.

QZx The Z (index) signal for quadrature counter X. May be used as a digital input if the index feature of the corresponding quadrature channel is unused.

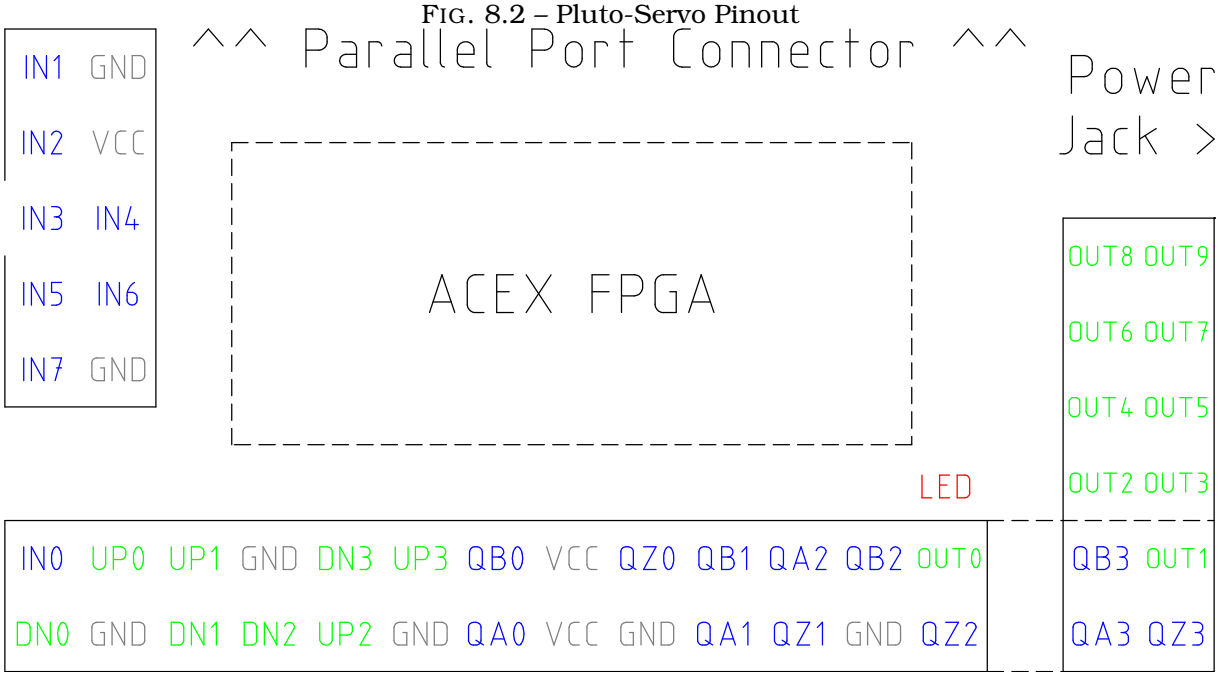
INx Dedicated digital input #x

OUTx Dedicated digital output #x

GND Ground

VCC +3.3V regulated DC

FIG. 8.2 – Pluto-Servo Pinout



TAB. 8.1 – Pluto-Servo Alternate Pin Functions

Primary function	Alternate Function	Behavior if both functions used
UP0	PWM0	When pwm-0-pwmdir is TRUE, this pin is the PWM output
	OUT10	XOR'd with UP0 or PWM0
UP1	PWM1	When pwm-1-pwmdir is TRUE, this pin is the PWM output
	OUT12	XOR'd with UP1 or PWM1
UP2	PWM2	When pwm-2-pwmdir is TRUE, this pin is the PWM output
	OUT14	XOR'd with UP2 or PWM2
UP3	PWM3	When pwm-3-pwmdir is TRUE, this pin is the PWM output
	OUT16	XOR'd with UP3 or PWM3
DN0	DIR0	When pwm-0-pwmdir is TRUE, this pin is the DIR output
	OUT11	XOR'd with DN0 or DIR0
DN1	DIR1	When pwm-1-pwmdir is TRUE, this pin is the DIR output
	OUT13	XOR'd with DN1 or DIR1
DN2	DIR2	When pwm-2-pwmdir is TRUE, this pin is the DIR output
	OUT15	XOR'd with DN2 or DIR2
DN3	DIR3	When pwm-3-pwmdir is TRUE, this pin is the DIR output
	OUT17	XOR'd with DN3 or DIR3
QZ0	IN8	Read same value
QZ1	IN9	Read same value
QZ2	IN10	Read same value
QZ3	IN11	Read same value
QA0	IN12	Read same value
QA1	IN13	Read same value
QA2	IN14	Read same value
QA3	IN15	Read same value
QB0	IN16	Read same value
QB1	IN17	Read same value
QB2	IN18	Read same value
QB3	IN19	Read same value

8.9.2 Input latching and output updating

– PWM duty cycles for each channel are updated at different times.
– Digital outputs OUT0 through OUT9 are all updated at the same time. Digital outputs OUT10 through OUT17 are updated at the same time as the pwm function they are shared with.
– Digital inputs IN0 through IN19 are all latched at the same time.
– Quadrature positions for each channel are latched at different times.

8.9.3 HAL Functions, Pins and Parameters

A list of all 'loadrt' arguments, HAL function names, pin names and parameter names is in the manual page, *pluto_servo.9*.

8.9.4 Compatible driver hardware

A schematic for a 2A, 2-axis PWM servo amplifier board is available (http://emergent.unpy.net/projects/01148303608). The L298 H-Bridge (L298 H-bridge http://www.st.com/stonline/books/pdf/docs/1773.pdf) is inexpensive and can easily be used for motors up to 4A (one motor per L298) or up to 2A (two motors per L298) with the supply voltage up to 46V. However, the L298 does not have built-in current limiting, a problem for motors with high stall currents. For higher currents and voltages, some users have reported success with International Rectifier's integrated high-side/low-side drivers. (http://www.cnczone.com/forums/showthread.php?t=25929)

8.10 Pluto-step : 300kHz Hardware Step Generator

Pluto-step is suitable for control of a 3- or 4-axis CNC mill with stepper motors. The large number of inputs allows for a full set of limit switches.

The board features :

– 4 "step+direction" channels with 312.5kHz maximum step rate, programmable step length, space, and direction change times
– 14 dedicated digital outputs
– 16 dedicated digital inputs
– EPP communuication with the PC

8.10.1 Pinout

STEPx The "step" (clock) output of stepgen channel **x**

DIRx The "direction" output of stepgen channel **x**

INx Dedicated digital input #x

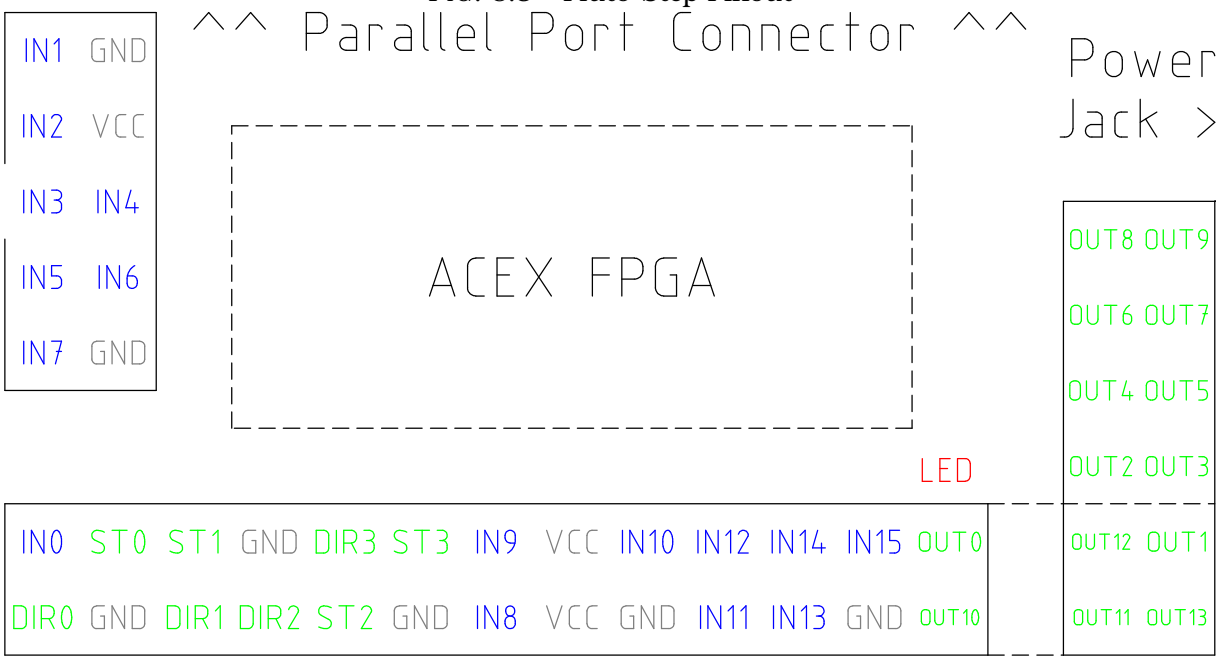
OUTx Dedicated digital output #x

GND Ground

VCC +3.3V regulated DC

While the "extended main connector" has a superset of signals usually found on a Step & Direction DB25 connector–4 step generators, 9 inputs, and 6 general-purpose outputs–the layout on this header is different than the layout of a standard 26-pin ribbon cable to DB25 connector.

FIG. 8.3 – Pluto-Step Pinout



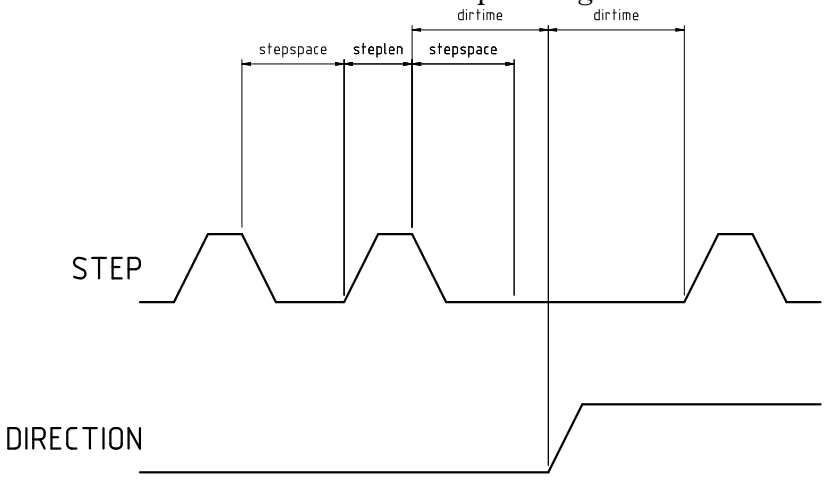
8.10.2 Input latching and output updating

– Step frequencies for each channel are updated at different times.
– Digital outputs are all updated at the same time.
– Digital inputs are all latched at the same time.
– Feedback positions for each channel are latched at different times.

8.10.3 Step Waveform Timings

The firmware and driver enforce step length, space, and direction change times. Timings are rounded up to the next multiple of $1.6\mu s$, with a maximum of $49.6\mu s$. The timings are the same as for the software stepgen component, except that “dirhold” and “dirsetup” have been merged into a single parameter “dirtime” which should be the maximum of the two, and that the same step timings are always applied to all channels.

FIG. 8.4 – Pluto-Step Timings



8.10.4 HAL Functions, Pins and Parameters

A list of all 'loadrt' arguments, HAL function names, pin names and parameter names is in the manual page, *pluto_step.9*.

Troisième partie

Utilisation d'EMC2

Chapitre 9

Utiliser l'interface graphique AXIS

9.1 Introduction

AXIS est une interface utilisateur graphique pour emc2, il offre un aperçu permanent du tracé de la pièce et du cheminement de l'outil. Il est écrit en Python, utilise Tk et OpenGL pour l'affichage de l'interface graphique.

9.2 Pour commencer

Pour choisir AXIS comme interface graphique d'emc2, éditez le fichier .ini et dans la section `[DISPLAY]` changez la ligne `DISPLAY` comme ceci :

```
DISPLAY = axis
```

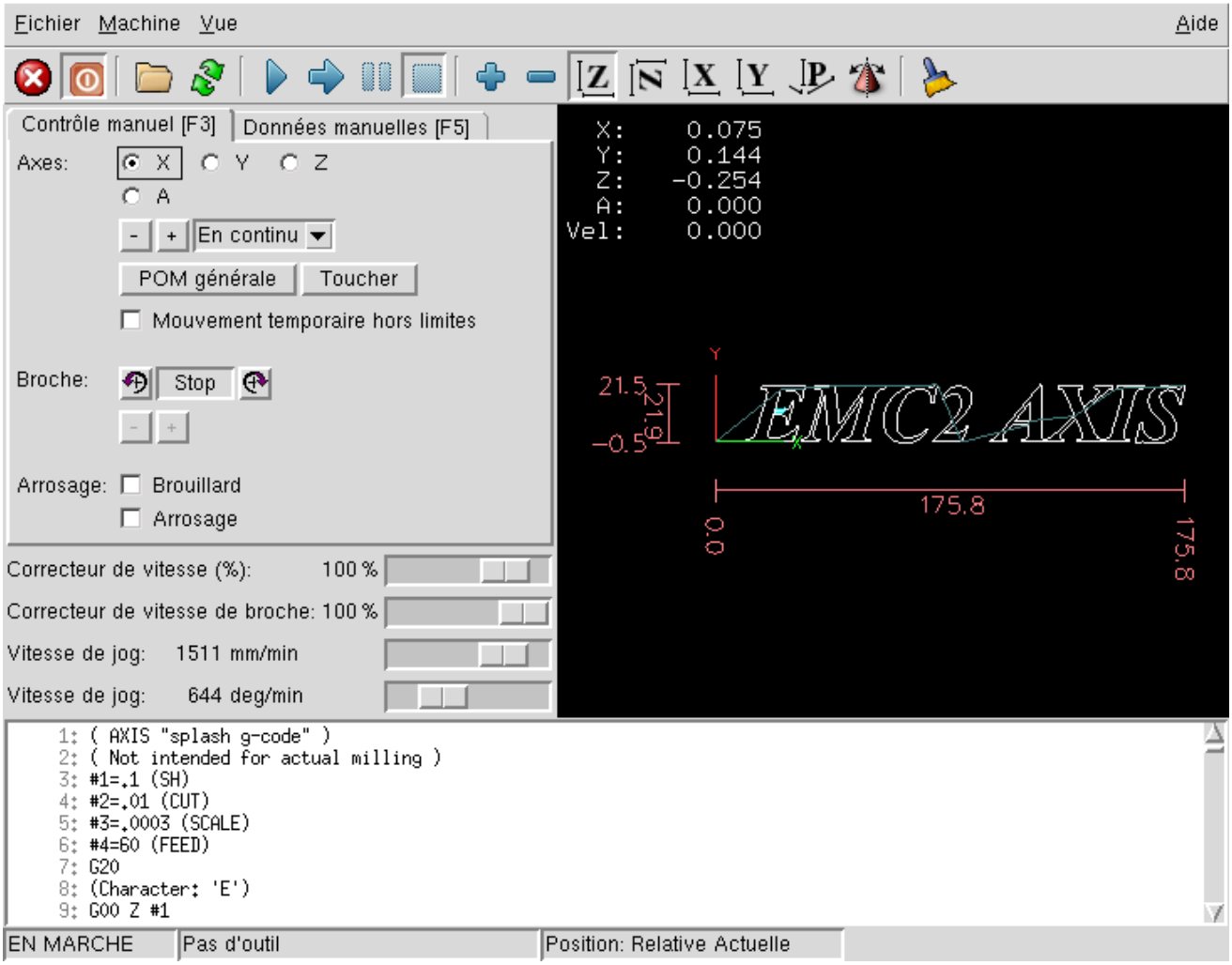
Puis, lancez emc2 et choisissez le fichier ini. La configuration simplifiée `sim/axis.ini` est déjà configurée pour utiliser AXIS comme interface.

Quand vous démarrez AXIS, une fenêtre telle que celle de la figure 9.1 s'ouvre.

9.2.1 Une session typique avec AXIS

1. Lancez emc et sélectionnez un fichier de configuration.
2. Relâchez le bouton d'arrêt d'urgence "A/U" et mettez la machine en marche.
3. Faites les prises d'origine machine "POM" pour chacun des axes.
4. Chargez un fichier d'usinage.
5. Utilisez l'écran de tracé du chemin d'outil pour vérifier que le programme est correct.
6. Bridez le brut à usiner sur la table.
7. Faites les prises d'origine pièce "POP" de chacun des axes avec le jog et en utilisant le bouton "Toucher".
8. Lancez le programme.
9. Pour usiner le même fichier une nouvelle fois, retournez à l'étape 6. Pour usiner un fichier différent, retournez à l'étape 4. Quand vous avez fini, quittez AXIS.

FIG. 9.1 – Fenêtre d'AXIS



9.3 Eléments de la fenêtre d'AXIS

La fenêtre d'AXIS contient les éléments suivants :

- Un espace d'affichage qui montre une pré-visualisation du fichier chargé (dans ce cas, "axis.ngc"), ainsi que la position courante du point programmé par la machine CNC. Plus tard, cette zone affichera le cheminement de l'outil déplacé par la machine CNC, cette zone est appelée le chemin d'outil ("backplot")
- Une barre de menus, une barre d'outils, des curseurs et des onglets permettent d'effectuer différentes actions.
- L'onglet "Contrôle manuel", qui permet de faire des mouvements d'axe, de mettre la broche en rotation ou de l'arrêter, de mettre l'arrosage en marche ou de l'arrêter.
- L'onglet "Données manuelles" (aussi appelée MDI), où les blocs de programme G-code peuvent être entrés à la main, une ligne à la fois.
- Les curseurs "Correcteurs de vitesse", qui permettent d'augmenter ou de diminuer la vitesse de la fonction concernée.
- Une zone de textes qui affiche le G-code du fichier chargé.
- Une barre d'état qui affiche l'état de la machine. Dans cette capture d'écran, la machine est en marche, aucun outil n'est monté, la position affichée est "relative" à l'origine machine (par opposition à une position "absolue") et "actuelle" (par opposition à une position "commandée")

9.3.1 Boutons de la barre d'outils

Signification des boutons, de gauche à droite :

1. "Arrêt d'urgence" (A/U)
2. Marche/Arrêt machine
3. Ouvrir fichier G-code
4. Recharger le fichier courant
5. Exécution du programme
6. Exécuter un seul pas de programme
7. Pause/Reprise
8. Stopper l'exécution du programme
9. Zoom plus
10. Zoom moins
11. Vue prédéfinie "Z" (vue de dessus)
12. Vue prédéfinie"Z basculée"
13. Vue prédéfinie"X" (vue de côté)
14. Vue prédéfinie"Y" (vue de face)
15. Vue prédéfinie"P" (vue en perspective)
16. Rafraîchi le tracé du chemin d'outil

9.3.2 Zones d'affichage graphique du programme

9.3.2.1 Affichage des coordonnées

L'affichage des coordonnées est situé en haut à gauche de l'écran graphique. Il montre les positions de la machine. A gauche du nom de l'axe, un symbole d'origine () est visible si la prise d'origine de l'axe a été faite. A droite du nom de l'axe, un symbole de limite () est visible si l'axe est sur un de ses capteurs de limite.

Pour interpréter correctement ces valeurs, référez vous à l'indicateur "Position :" de la barre d'état. Si la position est "Absolue", alors les valeurs affichées sont exprimées en coordonnées machine. Si la position est "Relative", alors les valeurs affichées sont exprimées en coordonnées relatives à la pièce. Quand les coordonnées affichées sont relatives, une marque d'origine de couleur cyan est visible pour représenter "l'origine machine" (). Si la position est "Commandée", alors il s'agit de la position à atteindre – par exemple, les coordonnées passées dans une commande `G0`. Si la position est "Actuelle", alors il s'agit de la position à laquelle la machine vient de se déplacer. Ces valeurs peuvent varier pour certaines raisons : erreur de suivi, bande morte, résolution d'encodeur ou taille de pas. Par exemple, si vous demandez un mouvement à X 0.08 à votre fraiseuse, mais un pas du moteur fait 0.03, alors la position "Commandée" sera de 0.08 mais la position "Actuelle" sera de 0.06 (2 pas) ou 0.09 (3 pas).

9.3.2.2 Vue du chemin d'outil

Quand un fichier est chargé, une vue du chemin d'outil qu'il produira est visible dans la zone graphique. Les mouvements en vitesse rapide (tels ceux produits par une commande `G0`) sont affichés en lignes pointillées vertes. Les déplacements en vitesse programmée (tels ceux produits par une commande `G1`) sont affichés en lignes continues blanches. Les arrêts (tels ceux produits par la commande `G4`) sont représentés par une petite marque "X".

9.3.2.3 Etendues du programme

Les "étendues" du programme sont visibles pour chacun des axes. Aux extrémités de chacune, la plus petite et la plus grande valeur des coordonnées sont indiquées. Au milieu, la différence entre ces valeurs de coordonnées est visible. Sur la figure 9.1, l'étendue des X du fichier va de -1.95 à 1.88 pouces, un total de 3.83 pouces.

Quand une coordonnée excède une "soft limits" établie dans le fichier .ini, la dimension correspondante est affichée dans une couleur différente. Ici, la limite maximale est dépassée sur l'axe X :

9.3.2.4 Le cône d'outil

L'emplacement de la pointe de l'outil est indiqué par le "cône d'outil". Le cône d'outil ne donne pas d'indication sur la forme, la longueur ou le rayon de l'outil.

Quand un outil est chargé, (par exemple dans le MDI, avec la commande `T1M6`), le cône passe de cônique à cylindrique, il indique alors le diamètre de l'outil lu dans le fichier de la table d'outil.

9.3.2.5 Tracé d'outil

Quand la machine se déplace, elle laisse une trace appelée le tracé d'outil. La couleur des lignes indique le type de mouvement : Jaune pour les jogs, vert clair pour les mouvements en vitesse rapide, rouge pour les mouvements en vitesse d'avance programmée et magenta pour les mouvements circulaires en vitesse d'avance programmée.

9.3.2.6 Interaction avec l'affichage

Par un clic gauche sur une portion du chemin d'outil, la ligne sous la souris passe en surbrillance à la fois dans le graphique et dans le texte. Un clic droit dans une zone vide enlève la surbrillance.

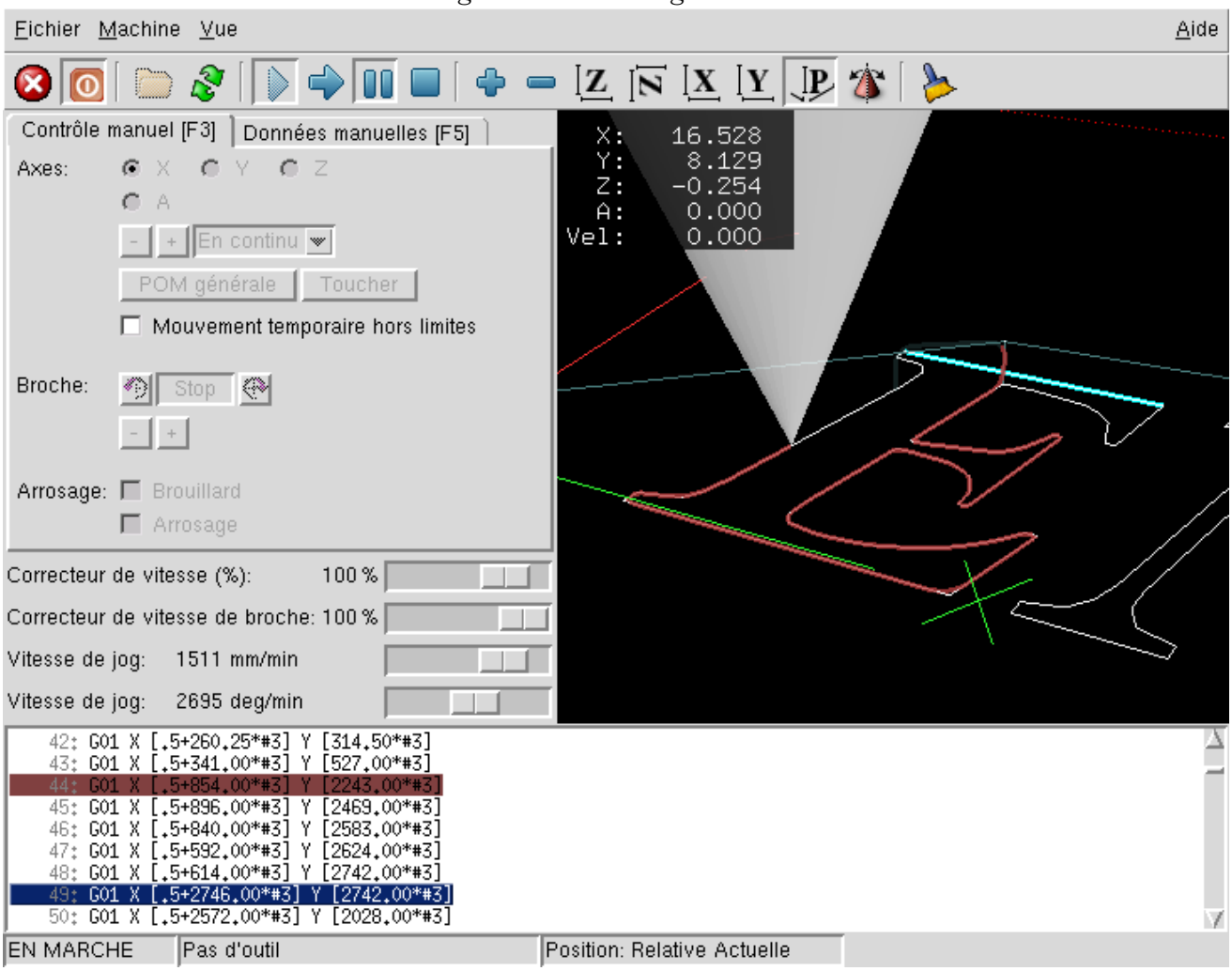
En déplaçant la souris avec son bouton gauche appuyé, la vue est glissée sur l'écran.

En déplaçant la souris avec le bouton "Maj" enfoncé, ou en glissant avec la molette de la souris appuyée, la vue est tournée. Si une ligne du tracé est en surbrillance, elle devient le centre de rotation de la vue. Autrement, le centre de rotation est le milieu du fichier dans son ensemble.

En tournant la molette de la souris ou en glissant la souris avec son bouton droit enfoncé ou encore en glissant la souris avec son bouton gauche enfoncé et la touche Ctrl appuyée, le tracé sera zoomé en plus ou en moins.

En cliquant sur une des icônes de vue pré-définie de la barre d'outils, ou en pressant la touche "V", cette vue est sélectionnée.

FIG. 9.2 – Ligne courante et ligne en surbrillance



9.3.3 Zone texte d'affichage du programme

Un clic gauche sur une ligne du programme passe la ligne en surbrillance à la fois dans la zone texte et dans la zone graphique.

Quand le programme est lancé, la ligne en cours d'exécution est en surbrillance rouge. Si aucune ligne n'est sélectionnée par l'utilisateur, le texte défile automatiquement pour toujours laisser la ligne courante visible.

9.3.4 Contrôle manuel

Quand la machine est en marche mais qu'aucun programme n'est exécuté, les éléments graphiques de l'onglet "Contrôle manuel" peuvent être utilisés pour actionner le centre d'usinage ou mettre en marche et arrêter les différents organes de la machine.

Quand la machine n'est pas en marche ou quand un programme est en cours d'exécution, le contrôle manuel est indisponible.

Certains des éléments décrits plus bas ne sont pas disponibles sur toutes les machines. Quand AXIS détecte qu'une "pin" particulière n'est pas connectée dans le fichier HAL, l'élément correspondant de l'onglet Contrôle manuel est supprimé. Par exemple, si la "pin HAL" `motion.spindle-brake` n'est pas connectée, alors le bouton "Frein de broche" n'apparaîtra pas sur l'écran. Si la variable

d'environnement AXIS_NO_AUTOCONFIGURE est mise à 1, ce comportement est désactivé et tous les boutons sont visibles.

9.3.4.1 Le groupe "Axes"

Les cases à cocher du groupe"Axes" permettent de choisir l'axe de la machine à actionner manuellement. Cette action s'appelle le "jog". Premièrement sélectionnez l'axe à actionner en cochant sa case. Puis cliquez le bouton "+" ou "-" selon le sens de déplacement souhaité. Les quatre premiers axes peuvent aussi être déplacés avec les touches fléchées pour (X et Y), Page précédente et Page suivante pour (Z) et les touches [et] pour (A).

Si "En continu" est sélectionné, le mouvement continuera tant que la touche ou le bouton resteront appuyés. Si une autre valeur est sélectionnée, la machine se déplacera juste de la distance affichée à chaque fois que la touche ou le bouton seront appuyés. Par défaut, les valeurs disponibles sont :

0.1000 0.0100 0.0010 0.0001

Le réglage du fichier .ini [DISPLAY]INCREMENTS peut être utilisé pour remplacer ces valeurs par défaut. Ces valeurs doivent contenir des nombres décimaux (ex. 0.1000) ou des nombres fractionnaires (ex. 1/16), éventuellement suivis par une unité (une parmi 'cm', 'mm', 'um', 'inch', 'in', ou 'mil'). Si aucune unité n'est spécifiée, les unités natives de la machine seront utilisées. Pour les utilisateurs préférant les unités métriques, un bon réglage pourrait être :

```
INCREMENTS = 10 um, 50 um, 0.1mm, 0.5mm, 1mm, 5mm, 10 mm
```

Pour les utilisateurs préférant les unités "impériales", un bon réglage pourrait être :

```
INCREMENTS = 1/4 in, 1/16 in, 1/32 in, 1/64 in, 1 mil, .1 mil
```

ou

```
INCREMENTS = .5 in, .1 in, 50 mil, 10 mil, 5 mil, 1 mil, .1 mil
```

Le mélange entre métrique et impérial est possible :

```
INCREMENTS = 1 inch, 1 mil, 1 cm, 1 mm, 1 um
```

Si votre machine a une séquence de prise d'origines machine définie, le bouton "POM générale" ou les touches Ctrl+origine lanceront cette séquence. Autrement, le bouton sera lu "Origine machine", et enverra l'axe courant à sa position d'origine. L'appui de la touche origine seule enverra l'axe courant à sa position d'origine même si une séquence de prise d'origines machine est définie. Selon votre configuration, la POM peut ajuster la valeur de la position de l'axe comme étant la position absolue 0.0, ou elle peut faire déplacer la machine vers une position d'origine spécifiée et avec l'usage de "capteurs d'origine". Voir la section 5.4 pour plus d'informations sur les prises d'origine.

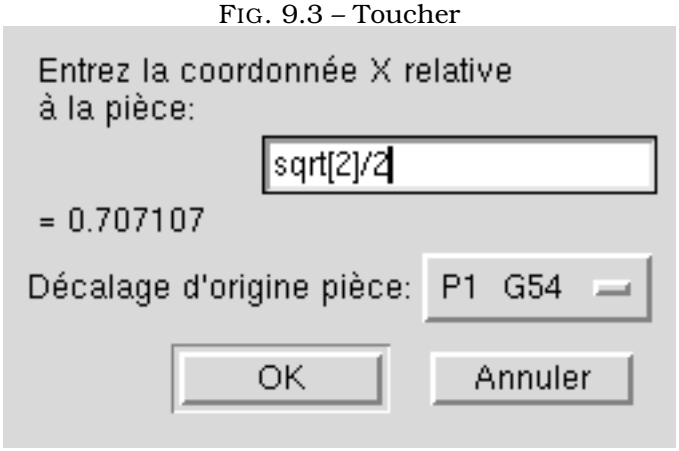
Si le bouton "Toucher" ou la touche Fin sont appuyés, le décalage d'origine pièce "G54" de l'axe actif prendra la valeur spécifiée dans le champ de la boite de dialogue. Les expressions peuvent être entrées en suivant les règles de programmation rs274ngc, sauf les variables qui ne peuvent pas être utilisées. La valeur résultante sera affichée sous le champ.

La case à cocher "Mouvement temporaire hors limites" permet un déplacement temporaire de la machine hors des limites définies dans le fichier ini. Par exemple, pour ramener dans les limites, un axe qui les a dépassé par erreur.

9.3.4.2 Le groupe "Broche"

Les boutons de la première rangée permettent de sélectionner la direction de rotation de la broche : Sens anti-horaire, Arrêt, Sens horaire. Les boutons de la rangée suivante augmentent ou diminuent la fréquence de rotation. La case à cocher de la troisième rangée permet d'engager ou de relâcher le frein de broche. Selon la configuration de votre machine, ces éléments n'apparaîtront peut être pas tous.

FIG. 9.3 – Toucher



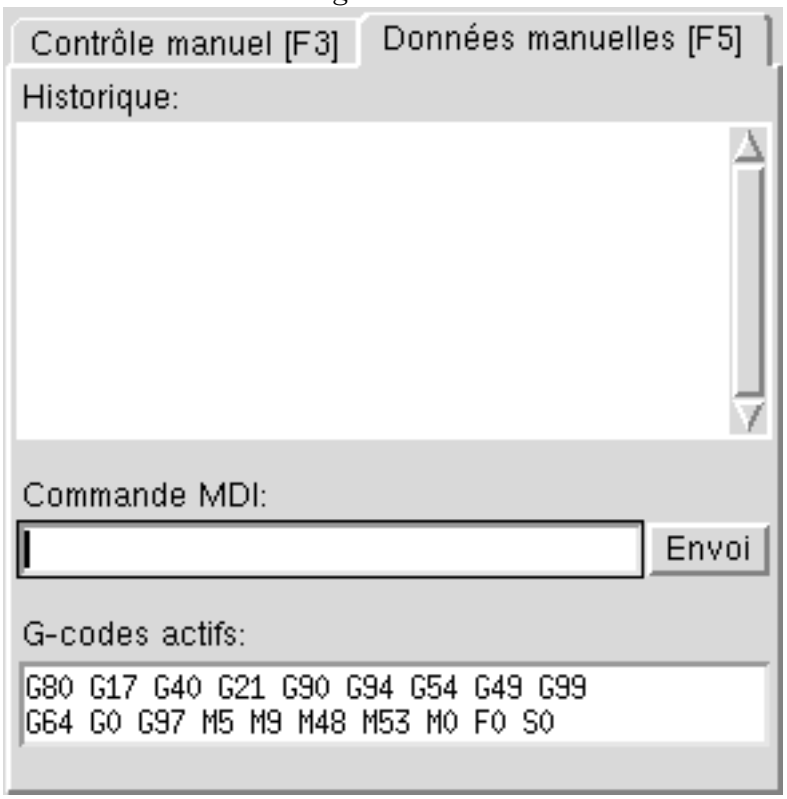
9.3.4.3 Le groupe "Arrosage"

Ces deux boutons permettent d'activer le "Brouillard" et l'"Arrosage" ou de les désactiver. Selon la configuration de votre machine, ces boutons n'apparaîtront peut être pas tous.

9.3.5 Données manuelles

L'onglet d'entrée de données manuelles (encore appelées MDI), permet d'entrer manuellement et une par une, des lignes de programme en G-code. Quand la machine n'est pas en marche, ou quand un programme est en cours d'exécution, cet onglet n'est pas opérationnel.

FIG. 9.4 – L'onglet Données manuelles



9.3.5.1 Historique

Affiche les commandes précédemment tapées au cours de cette session.

9.3.5.2 Commandes MDI

Ce champ permet la saisie d'une ligne de commande à exécuter. La commande sera exécutée par l'appui de la touche "Entrée" ou un clic du bouton "Envoi".

9.3.5.3 G-Codes actifs

Ce champ affiche les "codes modaux" actuellement actifs dans l'interpréteur. Par exemple, "G54" indique que le décalage d'origine "G54" sera appliqué à toutes les coordonnées qui seront entrées.

9.3.6 Correcteurs de vitesse

En déplaçant le curseur, la vitesse de déplacement programmée peut être modifiée. Par exemple, si un programme requiert une vitesse à `F60` et que le curseur est placé sur 120%, alors la vitesse résultante sera de 72.

9.3.7 Correcteur de vitesse de broche

En déplaçant ce curseur, la vitesse programmée de la broche peut être modifiée. Par exemple, si un programme requiert une vitesse à F8000 et que le curseur est placé sur 80%, alors la fréquence de rotation résultante sera de 6400. Cet élément n'apparaît que si la "HAL pin" `motion.spindle-speed-out` est connectée dans .ini.

9.3.8 Vitesse de Jog

En déplaçant ce curseur, la vitesse de jog peut être modifiée. Par exemple, si ce curseur est placé sur 100 mm/mn, alors un jog de 1 mm durera .6 secondes, ou 1/100 de minute. Du côté gauche du curseur (jog lent) l'espacement des valeurs est petit alors que du côté droit (jog rapide) l'espacement des valeurs est plus grand, cela permet une large étendue de vitesses de jog avec un contrôle plus fin du curseur dans les zones les plus importantes.

Sur les machines avec axes rotatifs, un second curseur de vitesse est présent. Il permet d'ajuster la vitesse de rotation des axes rotatifs (A, B et C).

9.4 Raccourcis clavier

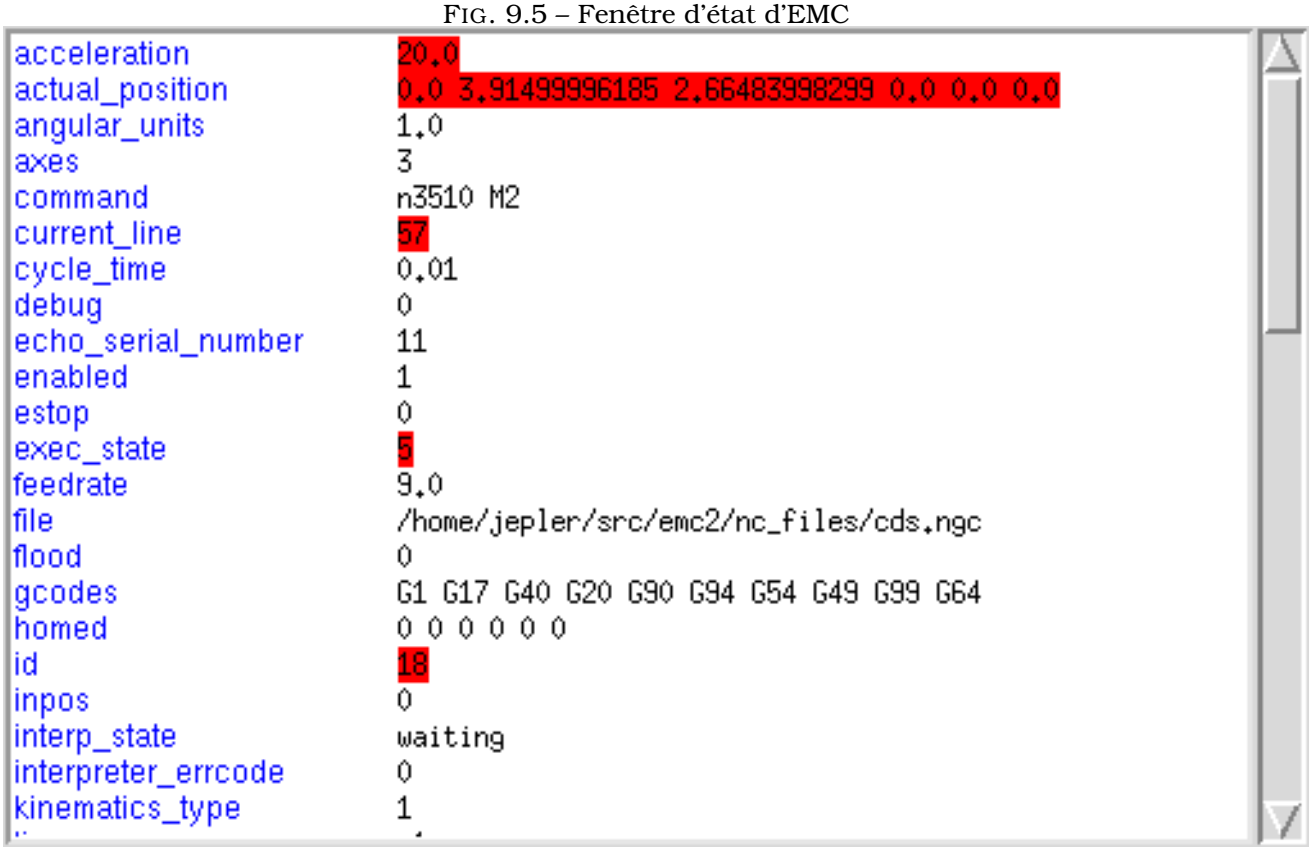
La plupart des actions d'AXIS sont accessibles depuis le clavier. La liste complète des raccourcis clavier est disponible dans l'aide rapide d'AXIS qui s'affiche en cliquant sur AIDE > AIDE RAPIDE. Beaucoup de ces raccourcis sont inaccessible en mode Entrées manuelles.

Les raccourcis clavier les plus fréquents sont visibles dans la table 9.1.

TAB. 9.1 – Raccourcis clavier usuels

Touches	Actions produites
F1	Bascule l'arrêt d'urgence
F2	Bascule le marche/arrêt machine
‘, 1 .. 9, 0	Correcteurs de vitesse de 0% à 100%
X, ‘	Active le premier axe
Y, 1	Active le deuxième axe
Z, 2	Active le troisième axe
A, 3	Active le quatrième axe
I	Sélection d'incrément du jog
C	jog en mode continu
Ctrl+origine	Lance une séquence de POM
Fin	Toucher : valide l'offset G54 de l'axe actif
Gauche, Droite	Jog du premier axe
Up, Down	Jog du deuxième axe
Pg Up, Pg Dn	Jog du troisième axe
[,]	Jog du quatrième axe
O	Ouvrir un fichier
Ctrl+R	Recharger le fichier courant
R	Exécuter le programme
P	Pause dans l'exécution du programme
S	Reprise de l'exécution du programme
ESC	Stopper l'exécution
Ctrl+K	Raffraîchi le tracé d'outil
V	Défilement cyclique des vues prédéfinies

FIG. 9.5 – Fenêtre d'état d'EMC



9.5 emctop : Affichage de l'état d'EMC

AXIS inclut un programme appelé "emctop" qui affiche en détail l'état d'emc. Ce programme est accessible dans le menu MACHINE > FENÊTRE D'ÉTAT D'EMC2

Le nom de chaque entrée est affiché dans la colonne de gauche. La valeur courante de chaque entrée s'affiche dans la colonne de droite. Si la valeur a changé récemment, elle s'affiche en surbrillance rouge.

9.6 mdi : Interface d'entrée de texte (MDI)

AXIS inclut un programme appelé "mdi", il permet d'envoyer des commandes à la session d'EMC2 en cours, sous forme de lignes de texte. Vous pouvez lancer ce programme en ouvrant une console et en tapant :

```
mdi /path/to/emc.nml
```

En cours d'exécution il affiche le prompt : `MDI>`. Quand une ligne vide est entrée, la position courante de la machine est affichée. Quand une commande est entrée, elle est passée à emc qui l'exécute. Une courte session MDI est visible sur la figure 9.6.

FIG. 9.6 – Session MDI

```
$ mdi ~/emc2/configs/sim/emc.nml
MDI>
(0.0, 0.0, 0.0, 0.0, 0.0, 0.0)
MDI> G1 F5 X1
MDI>
(0.5928500000000374, 0.0, 0.0, 0.0, 0.0, 0.0)
MDI>
(1.0000000000000639, 0.0, 0.0, 0.0, 0.0, 0.0)
```

9.7 axis-remote : Commande à distance de l'interface graphique d'AXIS

AXIS inclut un programme appelé "axis-remote" qui permet d'envoyer certaines commandes vers l'application AXIS fonctionnant à distance. Les commandes disponibles sont visibles en faisant : `axis-remote --help` pour vérifier qu'AXIS est en marche, inclure : (`--ping`), charger un fichier, recharger le fichier courant avec : (`--reload`) et quitter le programme AXIS avec : (`--quit`).

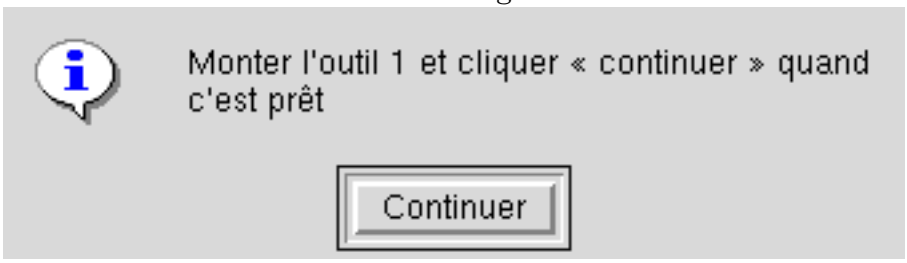
9.8 hal_manualtoolchange : Dialogue de changement manuel d'outil

AXIS inclut un composant Hall appelé "hal_manualtoolchange", il ouvre une fenêtre d'appel d'outil (Figure 9.7) quand la commande `M6` est invoquée. Dés que le bouton Continuer a été pressé, l'exécution du programme reprend.

Le fichier de configuration .hal : `configs/sim/axis_manualtoolchange.hal` montre les commandes HAL nécessaires pour l'utilisation de ce composant.

hal_manualtoolchange peut être utilisé même si l'interface graphique AXIS n'est pas en service.

FIG. 9.7 – La fenêtre de changement manuel d'outil



9.9 Modules en Python

AXIS inclut plusieurs modules en Python qui peuvent être très utiles. Pour des informations complètes sur ces modules, faites : "`pydoc <nom du module>`" ou lisez son code source. Modules inclus :

– `emc` fournit l'accès aux commandes d'emc, à son état et aux chaînes d'erreur
– `gcode` fournit l'accès à l'interpréteur rs274ngc
– `rs274` fournit des outils supplémentaires pour travailler sur les fichiers rs274ngc
– `hal` permet la création par l'utilisateur de composants de HAL écrits en Python
– `_togl` fournit des éléments OpenGL utilisables dans les applications Tkinter
– `minigl` fournit l'accès aux sous-ensembles d'OpenGL utilisés par AXIS

Pour utiliser ces modules dans vos propres scripts, assurez-vous que le répertoire où ils se trouvent est dans le chemin d'accès des modules Python. Avec une version installée d'EMC2, ça se fera automatiquement. Avec une version installée en "in-place", ça peut être fait avec l'aide de : `scripts/emc-environmer`

9.10 Utiliser AXIS pour contrôler un tour CNC

En incluant cette ligne dans le fichier ini le mode tour (lathe) sera sélectionné :

```
[DISPLAY]
LATHE = 1
```

L'axe "Y" ne sera pas visible dans l'affichage des coordonnées, la vue sera modifiée pour placer l'axe Z dans le sens gauche/droite et l'axe X dans le sens avant/arrière et différents éléments (tels que les icônes des vues prédéfinies) seront supprimés.

La touche "V" agit alors sur le zoom pour afficher le tracé complet du fichier chargé.

En mode tour (lathe), la forme de l'outil chargé est visible.

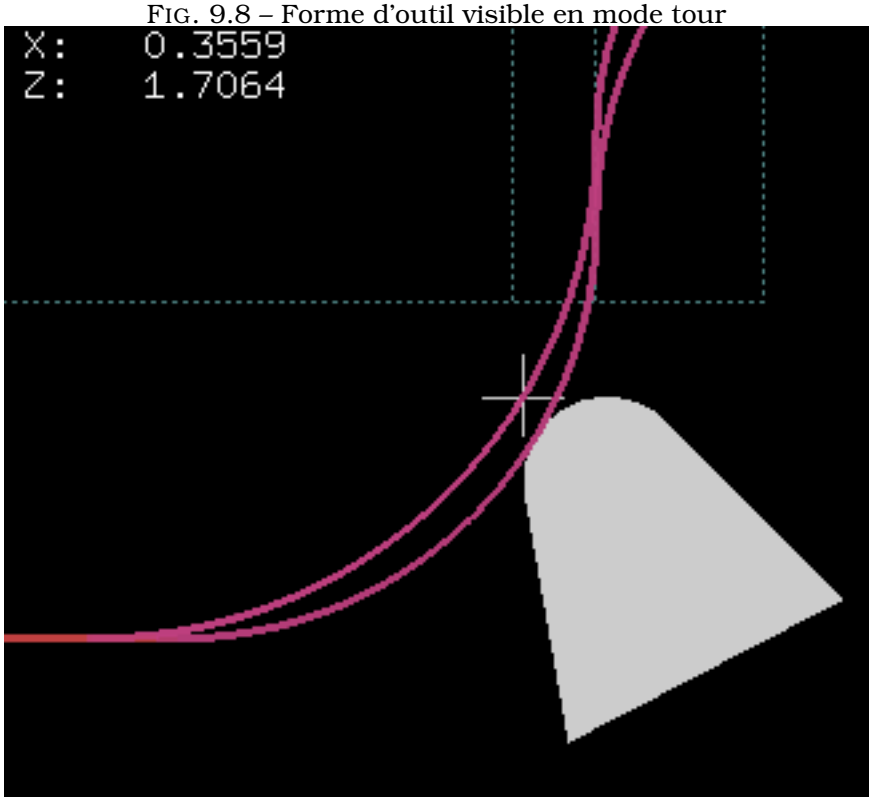
9.11 Configuration avancée d'AXIS

9.11.1 Filtres de programme

AXIS a la capacité d'envoyer des fichiers chargés à travers un "filtre de programme". Ce filtre peut faire diverses tâches : Simple, comme s'assurer que le programme se termine bien par un : '`M2`' ou complexe, comme détecter que l'entrée est une image et générer le g-code qui permettra d'usiner sa forme.

La section `[FILTER]` du fichier ini défini comment les filtres doivent agir. Premièrement, pour chaque type de fichier, écrire une ligne : `PROGRAM_EXTENSION` puis, spécifier le programme à exécuter pour chaque type de fichier. Ce programme reçoit comme argument le nom du fichier d'entrée,

FIG. 9.8 – Forme d'outil visible en mode tour



il doit produire le code selon le standard rs274ngc, en sortie. Les lignes de cette sortie s'affichent alors dans la zone texte, le chemin d'outil résultant est visible dans la zone graphique, enfin il sera exécuté quand emc recevra la commande "Exécuter le programme". Les lignes suivantes fournissent la possibilité d'utiliser "image-to-gcode", le convertisseur d'images fourni avec EMC2 :

```
[FILTER]
PROGRAM_EXTENSION = .png,.gif Greyscale Depth Image
png = image-to-gcode
gif = image-to-gcode
```

Il est également possible de spécifier un interpréteur :

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

De cette manière, n'importe quel script Python pourra être ouvert et sa sortie traitée comme du g-code. Un autre exemple est disponible dans : `nc_files/holecircle.py`. Ce script crée le g-code pour percer une série de trous selon la circonférence d'un cercle.

Si la variable d'environnement : AXIS_PROGRESS_BAR est active, alors les lignes seront écrites sur stderr de la forme :

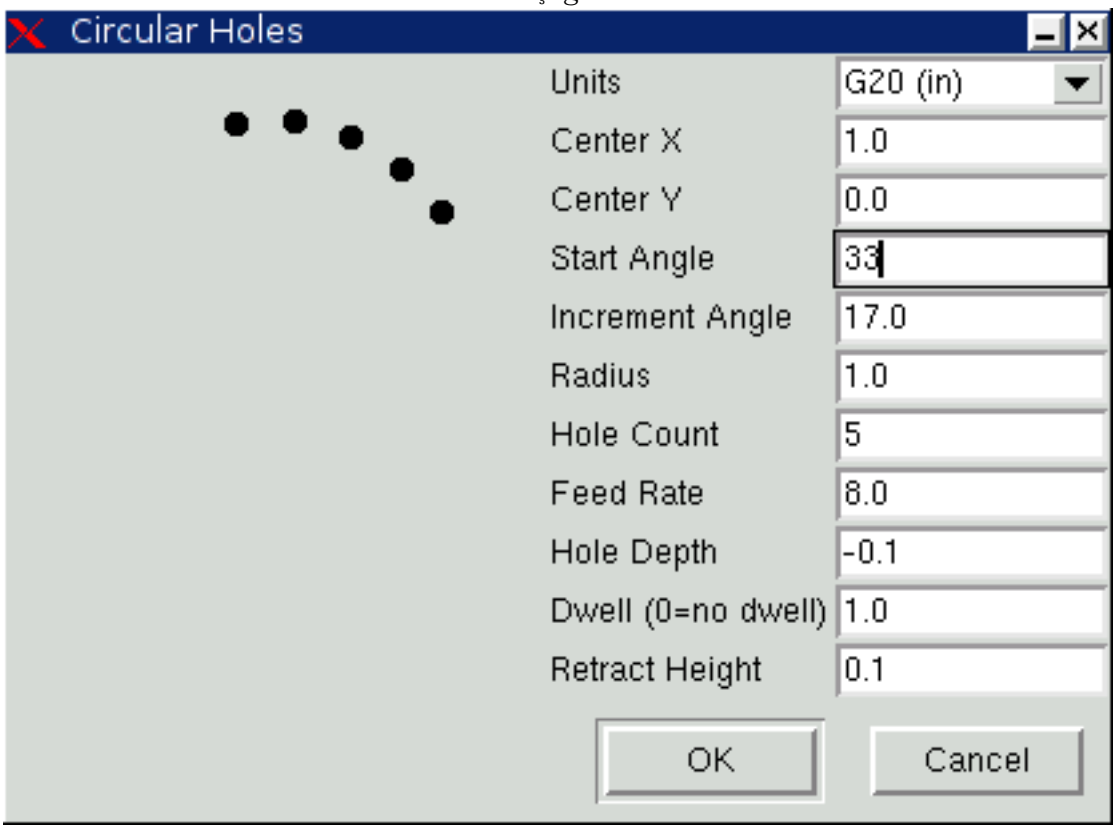
```
FILTER_PROGRESS=%d
```

AXIS fixera la barre de progression selon le pourcentage donné. Cette fonction devrait être utilisée pour un filtre qui fonctionne suffisamment longtemps.

9.11.2 La base de données des ressources X

Les couleurs de la plupart des éléments de l'interface utilisateur d'AXIS peuvent être personnalisées grâce à la base de données X. Le fichier : `axis_light_background` modifie les couleurs de la fenêtre

FIG. 9.9 – Perçages circulaires



du chemin d'outil sur le modèle "lignes noires et fond blanc", il sert aussi de référence des éléments configurables dans l'écran graphique.

Pour plus d'informations au sujet des éléments configurables dans les applications Tk, référez vous aux manuels de Tk.

Les bureaux graphiques modernes effectuent certains réglages dans la base de données des ressources X ces réglages peuvent affecter ceux d'AXIS, par défaut ces réglages sont ignorés. Pour que les éléments des ressources X écrasent ceux par défaut dans AXIS, il faut inclure cette ligne dans vos ressources X :

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

ce qui entraînera la construction des options au niveau "widgetDefault", de sorte que les ressources X (qui sont elles, au niveau "userDefault") puissent l'emporter.

9.11.3 Manivelle de jog

Pour accroître l'interaction d'AXIS avec une manivelle de jog physique, l'axe actif courant sélectionné dans l'interface graphique est aussi reporté sur une "pin Hal" avec un nom comme `axisui.jog.x`. Excepté pendant un court instant après que l'axe courant ait changé, une seule de ces pins à la fois est `TRUE`, les autres restent `FALSE`.

Après qu'AXIS ait créé ces "HAL pins", il exécute le fichier hal déclaré avec : `[HAL]POSTGUI_HALFILE`. Ce qui diffère de `[HAL]HALFILE`, qui lui ne s'utilise qu'une seule fois.

9.11.4 ~/.axisrc

Si il existe, le contenu de : `~/.axisrc` est exécuté comme un code source Python juste avant l'ouverture de l'interface graphique d'AXIS. Les détails de ce qui peut être écrit dans .axisrc sont sujets à changement durant le cycle de développement.

Les lignes visibles sur la figure 9.10 ajoutent un Ctrl+Q comme raccourci clavier pour Quitter et activer l'option "Distance restante" par défaut.

FIG. 9.10 – Exemple de fichier .axisrc

```
root_window.bind("<Ctrl+q>", "destroy .")
help2.append(("Ctrl+Q", "Quitter"))
vars.show_distance_to_go.set(1)
```

9.11.5 Editeur externe

En définissant : `[DISPLAY]EDITOR`, les options de menu : Fichier > Editer ainsi que Fichier > Editer la table d'outils, deviennent accessibles. Deux valeurs qui marchent bien : `EDITOR=gedit` et `EDITOR=gnome-terminal -e vim`.

9.11.6 Panneau de contrôle virtuel

AXIS peut afficher le fichier xml pyVCP dans le panneau droit de l'interface.

Chapitre 10

Vue d'ensemble d'un centre d'usinage

Cette section donne une brève description des interactions entre les organes d'un centre d'usinage et son interpréteur de commandes. Le lecteur est sensé être déjà familier avec les centres d'usinage.

Le langage RS274/NGC et les fonctions d'usinage standards qui en découlent permettent d'envisager un centre d'usinage sous deux points de vue :

(1) les composants mécaniques de la machine.

(2) les composants de contrôle et les données utilisées pour le contrôle de la machine.

10.1 Composants mécaniques

Un centre d'usinage dispose de beaucoup de composants mécaniques pouvant être contrôlés ou qui peuvent avoir une incidence sur la façon dont le contrôle de la machine s'effectue. Cette section décrit les composants qui interagissent avec l'interpréteur. Les composants mécaniques, qui n'interagissent pas avec l'interpréteur, comme les boutons du jog, ne seront pas décrits ici, même si ils affectent le contrôle.

10.1.1 Axes

Tout centre d'usinage dispose d'un ou de plusieurs axes. Les différents types de machines ont différentes combinaisons d'axes. Par exemple, une fraiseuse 4 axes peut avoir la combinaison d'axes XYZA ou XYZB. Un tour classique aura les axes XZ. Une machine de découpe à fil chaud aura les axes XYUZ.[1][2]

10.1.1.1 Axes linéaires primaires

Les axes X, Y et Z produisent des mouvements linéaires dans trois directions, mutuellement orthogonales.

[1] Si le mouvement des composants mécaniques n'est pas indépendant, comme sur une machine hexapode, le langage RS274/NGC et les fonctions standards seront quand même utilisables, tant que le contrôle de bas niveau sait comment contrôler les mécanismes actuels pour produire le mouvement relatif de l'outil et de la pièce qui auraient été produits par des axes indépendants. C'est appelé, la cinématique.

[2] Avec EMC, le cas de la machine à portique XYYZ avec deux moteurs pour un axe est mieux traité par la cinématique que par un axe linéaire supplémentaire.

10.1.1.2 Axes linéaires secondaires

Les axes U, V et W produisent des mouvements linéaires dans trois directions mutuellement orthogonales. Habituellement, X et U sont parallèles, Y et V sont parallèles et Z et W sont parallèles.

10.1.1.3 Axes rotatifs

Les axes A, B et C produisent des mouvements angulaires (rotations). Habituellement, l'axe de rotation de A est parallèle à X, l'axe de rotation de B est parallèle à Y et l'axe de rotation de C est parallèle à Z.

10.1.2 Broche

Un centre d'usinage est équipé d'une broche qui maintient l'outil coupant, la sonde de mesure et d'autres outils. La broche peut tourner dans les deux sens. Elle peut être conçue pour tourner à vitesse constante mais réglable. Excepté sur les machines dont la broche est montée sur un axe rotatif, l'axe de la broche est maintenu parallèle à l'axe Z et il est coïncident avec l'axe Z quand X et Y sont à zéro. La broche peut être stoppée sur une position fixée ou non.

10.1.3 Arrosages

Un centre d'usinage peut être équipé d'un composant fournissant l'arrosage fuide ou en brouillard.

10.1.4 Chargeur de pièces

Un centre d'usinage peut être équipé d'un système de chargement des pièces. Le système se compose de deux porte-pièces sur lesquels sont fixés les bruts des pièces à usiner. Un seul porte-pièce à la fois est en position d'usinage.

10.1.5 Carrousel d'outils

Un centre d'usinage peut être équipé d'un carrousel d'outils dans lequel sont rangés les outils déjà montés dans leurs porte-outils.

10.1.6 Changeur d'outil

Un centre d'usinage peut être équipé d'un mécanisme de changement d'outils (fixés dans les porte-outils) entre le carrousel d'outils et la broche.

10.1.7 Visu des messages

Un centre d'usinage est équipé d'une visualisation capable d'afficher les messages pour l'opérateur.

10.1.8 Correcteurs de vitesse d'avance et de broche

Un centre d'usinage est équipé de boutons de réglage de la vitesse d'avance et de la vitesse de rotation de la broche, ils laissent l'opérateur corriger les vitesses nécessaires pour la broche et l'avance travail, il peut ainsi augmenter ou réduire les vitesses programmées. Voir la section 10.3.1.

10.1.9 Bouton d'effacement de block

Un centre d'usinage peut être équipé d'un bouton d'effacement de block. Voir la section 10.3.2.

10.1.10 Bouton d'arrêt optionnel du programme

Un centre d'usinage peut être équipé d'un bouton d'arrêt du programme. Voir la section 10.3.3.

10.2 Composants de contrôle et de données

10.2.1 Axes linéaires

Les axes X, Y et Z forment un système de coordonnées orthogonales standard. La position d'un axe s'exprime en utilisant ses coordonnées.

Les axes U, V et W forment également un système de coordonnées standard. X et U sont parallèles, Y et V sont parallèles enfin Z et W sont parallèles.

10.2.2 Axes rotatifs

Les axes rotatifs se mesurent en degrés. Leur sens de rotation positif est le sens anti-horaire quand l'observateur est placé face à l'axe. [3]

10.2.3 Point contrôlé

Le point contrôlé est le point dont la position et la vitesse de déplacement sont contrôlés. Quand la compensation de longueur d'outil est zéro (valeur par défaut), c'est un point situé sur l'axe de la broche et proche de la fin de celle-ci. Cette position peut être déplacée le long de l'axe de la broche en spécifiant une compensation de longueur d'outil. Cette compensation correspond généralement à la longueur de l'outil coupant courant. Ainsi, le point contrôlé est à la pointe de l'outil. Sur un tour, les correcteurs d'outil peuvent être spécifiés pour les axes X et Z, le point contrôlé est à la pointe de l'outil ou (correction du rayon de bec) légèrement en retrait du point d'intersection des droites perpendiculaires formées par l'axe des points de tangence à la pièce, de face et sur le côté de l'outil.

10.2.4 Mouvement linéaire coordonné

Pour mener un outil sur une trajectoire spécifiée, un centre d'usinage doit coordonner les mouvements de plusieurs axes. Nous utilisons le terme "mouvement linéaire coordonné" pour décrire une situation dans laquelle, nominalement, chacun des axes se déplace à vitesse constante et tous les axes se déplacent de leur point de départ à leur point d'arrivée en même temps. Si deux des axes X, Y, Z (ou les trois) se déplacent, ceci produit un mouvement en ligne droite, d'où le mot "linéaire" dans le terme. Dans les véritables mouvements, ce n'est souvent pas possible de maintenir la vitesse constante à cause des accélérations et décélérations nécessaires en début et fin de mouvement. C'est faisable, cependant, de contrôler les axes ainsi, chaque axe doit en permanence faire la même fraction du mouvement requis que les autres axes. Ceci déplace l'outil le long du même parcours et nous appelons aussi ce genre de mouvement, mouvement linéaire coordonné.

Un mouvement linéaire coordonné peut être exécuté soit en vitesse travail, soit en vitesse rapide, ou il peut être synchronisé à la rotation de la broche. Si les limites physique de l'axe rendent le déplacement impossible, tous les axes seront ralentis pour maintenir le chemin prévu.

[3] Si les parallèlismes sont particuliers, le constructeur du système devra indiquer à quels sens de rotation correspondent horaire et anti-horaire.

10.2.5 Vitesse

La vitesse à laquelle le point contrôlé se déplace est ajustable par l'opérateur. Sauf cas particulier, (vitesse inverse du temps, vitesse par tour, voir la section 12.20) , dans l'interpréteur, l'interprétation des vitesses est la suivante :

1. Si le déplacement concerne un des axes XYZ, F est en unités machine par minute dans le système Cartésien XYZ et les mouvements des autres axes (UVWABC) sont également dans un même mode de coordonnées.
2. Autrement, si le déplacement concerne un des axes UVW, F est en unités machine par minute dans le système Cartésien UVW, tous les autres axes (ABC) se déplacent dans un même mode de coordonnées.
3. Autrement, le mouvement est purement rotatif et le mot F est en unités de rotation dans le système pseudo-Cartésien ABC.

10.2.6 Arrosage

Arrosage fluide ou brouillard (gouttelettes) peuvent être activés séparément. Le langage RS274/NGC les arrête ensemble (voir la section 13.4).

10.2.7 Temporisation

Une temporisation peut être commandée (ex : pour immobiliser tous les axes) pendant une durée spécifique. La broche n'est pas arrêtée pendant une temporisation ! Sans s'occuper du mode de contrôle de trajectoire (voir la section 10.2.15) la machine s'arrêtera exactement à la fin du dernier mouvement avant la temporisation.

10.2.8 Unités

Les unités utilisées pour les distances le long des axes X, Y et Z peuvent être les pouces ou les millimètres. La vitesse de rotation de la broche est en tours par minute. Les positions des axes rotatifs sont exprimées en degrés. Les vitesses d'avance sont exprimées en unités machine par minute ou en degrés par minute ou en unités de longueur par tour de broche, comme décrit dans la section 10.2.5.

10.2.9 Position courante

Le point contrôlé est toujours à un emplacement appelé la "position courante," et le contrôleur sait toujours où est cette position. Les valeurs représentant la position courante doivent être ajustées en l'absence de tout mouvement des axes si un de ces événements a lieu :

1. Les unités de longueur ont changé.
2. La compensation de longueur d'outil a changé.
3. Le décalage d'origine a changé.

10.2.10 Choix du plan de travail

Il y a toujours un plan sélectionné, qui doit être le plan XY, le plan YZ, ou le plan XZ de la machine. L'axe Z est, bien sûr, perpendiculaire au plan XY, l'axe X perpendiculaire au plan YZ et l'axe Y perpendiculaire au plan XZ.

10.2.11 carrousel d'outils

Aucun ou un outil est assigné à chaque emplacement dans le carrousel.

10.2.12 Changeur d'outil

Un centre d'usinage peut commander un changeur d'outils.

10.2.13 Chargeur de pièce

Les deux porte-pièces peuvent être intervertis par commande.

10.2.14 Boutons des correcteurs de vitesses

Les boutons des correcteurs de vitesses peuvent être activés (ils fonctionnent normalement) ou rendus inopérents (Ils n'ont plus aucun effet). Le langage RS274/NGC dispose d'une commande qui active tous les boutons et une autre qui les désactive (voir la section 13.5). Voir la section 10.3.1 pour d'autres détails.

10.2.15 Modes de contrôle de trajectoire

La machine peut être placée dans un de ces trois modes de contrôle de trajectoire : (1) mode arrêt exact, (2) mode trajectoire exacte ou (3) mode trajectoire continue avec tolérance optionnelle. En mode arrêt exact, le mobile s'arrête brièvement à la fin de chaque mouvement programmé. En mode trajectoire exacte, le mobile suit la trajectoire programmée aussi précisément que possible, ralentissant ou s'arrêtant si nécessaire aux angles vifs du parcours. En mode trajectoire continue, les angles vifs du parcours peuvent être légèrement arrondis pour que la vitesse soit maintenue (sans dépasser la tolérance, si elle est spécifiée). Voir la section 12.15.

10.3 Interaction de l'interpréteur avec les boutons

L'interpréteur interagit avec plusieurs boutons de commande. Cette section décrit ces interactions plus en détail. En aucun cas l'interpréteur ne connait ce que sont les réglages de ces boutons.

10.3.1 Boutons de correction de vitesses

L'interpréteur de commande RS274/NGC autorise (M48) ou interdit (M49) l'action des boutons d'ajustement des vitesses. Pour certains mouvements, tels que la sortie de filet à la fin d'un cycle de filetage, les boutons sont neutralisés automatiquement.

EMC2 réagit aux réglages de ces boutons seulement quand ils sont autorisés.

10.3.2 Bouton d'effacement de block

Si le bouton "Effacement de block" est actif, les lignes de code RS274/NGC commençant par le caractère barre de fraction (caractère d'effacement de block) ne sont pas interprétées. Si le bouton est désactivé, ces mêmes lignes sont interprétées. Normalement le bouton d'effacement de block doit être positionné avant de lancer le programme NGC.

10.3.3 Bouton d'arrêt optionnel du programme

Si ce bouton est actif et qu'un code M1 est rencontré, le programme est mis en pause.

10.4 Fichier d'outils

Un fichier d'outils est requis par l'interpréteur. Le fichier indique dans quels emplacements du carrousel sont placés les outils, la longueur et le diamètre de chacun des outils.

Le fichier est composé d'un certain nombre de lignes d'en-tête, suivies par une ligne vide, suivie d'un nombre quelconque de lignes de données. Les lignes d'en-tête sont ignorées par l'interpréteur. Il est important qu'il y ait une ligne vide (sans espace ni tabulation), avant les données. La ligne d'en-tête montrée dans ce tableau10.1, décrit les colonnes de données, il est donc proposé (mais pas obligatoire) que cette ligne soit toujours présente.

Chaque ligne de données du fichier contient les données d'un outil. La ligne peut contenir 4 ou 5 éléments ("format fraiseuse") ou 8 ou 9 éléments ("format tour").

Les unités utilisées pour la longueur et le diamètre sont en unités machine.

Les lignes n'ont pas à être dans un ordre particulier. Permuter l'ordre des lignes est sans effet, sauf si le même numéro d'emplacement est utilisé sur deux ou plusieurs lignes, ce qui ne devrait normalement pas être fait, dans ce cas, seules les données de la dernière de ces lignes seront utilisées.

Dans emc2, l'emplacement du fichier d'outil est spécifié dans le fichier ini. Voir la section 5.3.8 pour d'autres détails.

Un fichier d'outils peut être un mélange de lignes au "format fraiseuse" et au "format tour", bien que généralement les lignes du style "format tour" soient seulement requises pour les outils de tour.

10.4.1 Fichier d'outils au format fraiseuse

Le "format fraiseuse" d'un fichier d'outils est visible dans le tableau 10.1.

TAB. 10.1 – Exemple de fichier d'outils (format fraiseuse)

Slot	FMS	TLO	Diamètre	Commentaire
1	1	2.0	1.0	
2	2	1.0	0.2	
5	5	1.5	0.25	coupe en bout
10	10	2.4	-0.3	à tester

Chaque ligne comporte 5 éléments. Les quatre premiers éléments sont obligatoires. Le commentaire, placé en cinquième est facultatif. La lecture est rendue plus facile si les éléments sont disposés en colonnes, comme dans le tableau ci-dessus, mais la seule exigence de forme, c'est qu'il y ait au moins un espace ou une tabulation après chacun des trois premiers éléments d'une ligne, et une tabulation ou un saut de ligne à la fin des cinq éléments. La signification des colonnes et le type de données à mettre dans chacune sont les suivants :

La colonne "Slot" contient un entier non signé représentant le numéro d'emplacement dans le carrousel d'outils (numéro de slot) dans lequel cet outil est placé. Les éléments de cette colonne doivent être tous différents.

La colonne "FMS" contient un entier non signé représentant le numéro de code de l'outil. L'utilisateur peut y placer n'importe quel code pour tous les outils, tant que les codes sont des entiers non signés. C'est en général le même que celui de l'emplacement.

La colonne "TLO" contient un nombre réel représentant l'offset de longueur d'outil. Cette valeur sera utilisée si la compensation de longueur d'outil est activée et que ce numéro d'emplacement est choisi. C'est normalement un nombre réel positif, mais il peut être mis à zéro ou tout autre valeur si il n'a pas à être utilisé.

La colonne "Diamètre" contient un nombre réel. Cette valeur est utilisée seulement si la compensation de rayon d'outil est activée et que ce numéro d'emplacement est choisi. Si la trajectoire programmée avec la compensation est le bord du matériau à usiner, ce devrait être un nombre réel positif représentant le diamètre mesuré de l'outil. Si la trajectoire programmée pendant la compensation est la trajectoire d'un outil dont le diamètre est nominal, ce nombre devrait être petit (positif, négatif ou nul) et représenter la différence entre le diamètre mesuré de l'outil et le diamètre nominal. Si la compensation de rayon d'outil n'est pas utilisée avec un outil, le contenu de cette colonne est sans effet.

La colonne "Commentaire" est optionnelle, elle peut être utilisée pour décrire l'outil. Tout type de description convient. Cette colonne améliore la lisibilité du fichier.

10.4.2 Fichier d'outils au format tour

Le "format tour" d'un fichier d'outil est visible dans le tableau10.2.

TAB. 10.2 – Exemple de fichier d'outils (format tour)

Slot	FMS	ZOFFSET	XOFFSET	DIA	FRONTANGLE	BACKANGLE	ORIENTATION	Commentaire
1	1	0.0	0.0	0.1	95.0	155.0	1	
2	2	0.5	0.5	0.1	120	60	6	

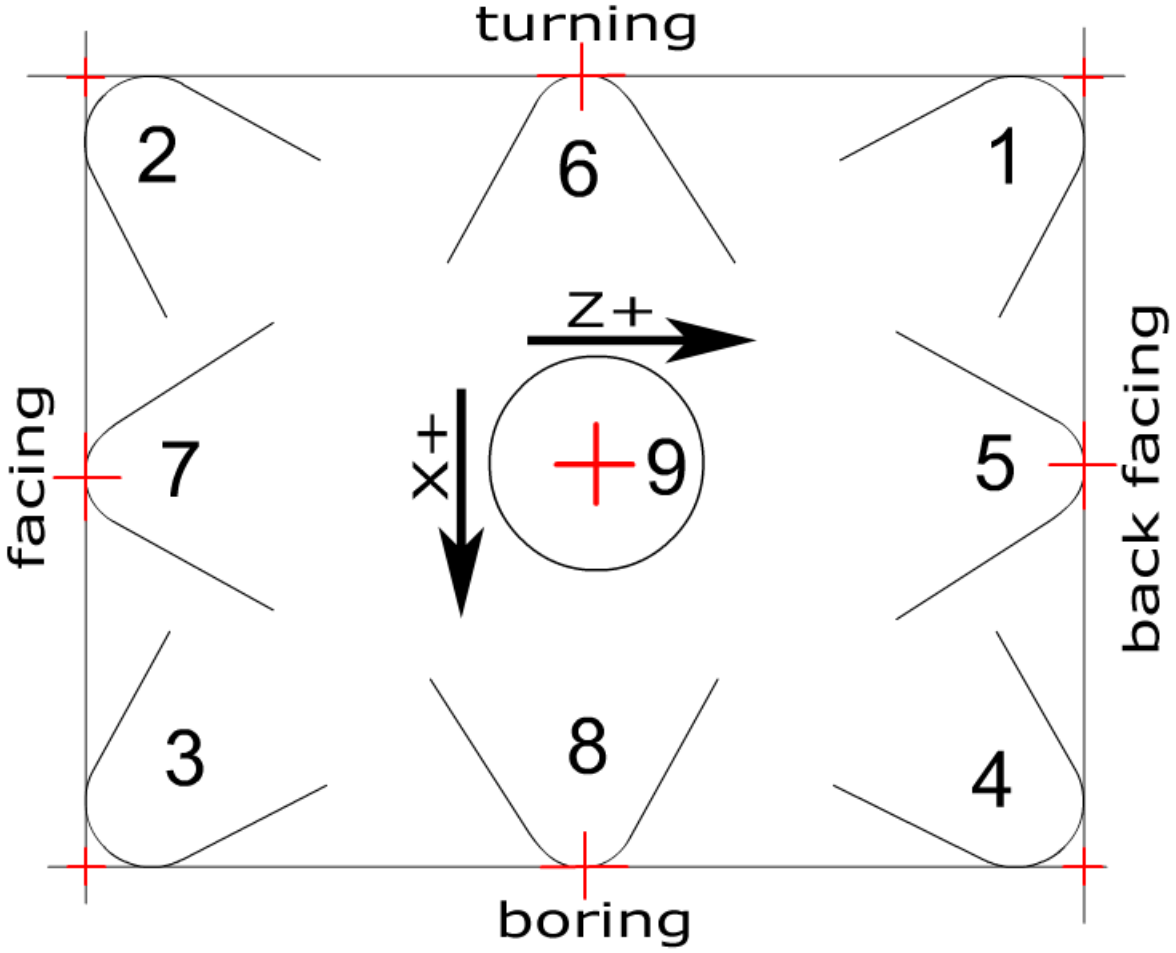
Les champs Slot, FMS, DIA et Commentaire sont les mêmes que dans un fichier d'outils de fraiseuse. La colonne ZOFFSET est la même que la colonne TLO d'un fichier d'outil de fraiseuse ; offset de longueur d'outil.

La colonne XOFFSET fournit l'offset pour la coordonnée X quand la compensation de longueur d'outil est active.

La colonne ORIENTATION donne l'orientation de l'outil de tour, comme illustré sur 10.1. La croix rouge correspond au point contrôlé. Voir 10.2.3.

Les colonnes FRONTANGLE et BACKANGLE sont utilisées par certaines interfaces utilisateur pour afficher une représentation de l'outil de tour.

FIG. 10.1 – Orientations des outils de tour



10.5 Paramètres

Dans le language RS274/NGC, le centre d'usinage maintient un tableau de 5400 paramètres numériques. La plupart d'entre eux ont un usage spécifique. Le tableau de paramètres est persistant, même quand la machine est mise hors tension. EMC2 utilise un fichier de paramètres et assure sa persistance, il donne à l'interpréteur la responsabilité d'actualiser le fichier. L'interpréteur lit le fichier quand il démarre et l'écrit juste avant de s'arrêter.

Le format d'un fichier de paramètres est visible dans la table 10.4. Le fichier est composé d'un certain nombre de lignes d'en-tête, suivie par une ligne vide, suivie d'un nombre quelconque de lignes de données. Les lignes d'en-tête sont ignorées par l'interpréteur. Il est important qu'il y ait une ligne vide (sans espace ni tabulation), avant les données. La ligne d'en-tête montrée dans ce tableau 10.4, décrit les colonnes de données, il est donc proposé (mais pas obligatoire) que cette ligne soit toujours présente.

L'interpréteur lit seulement les deux premières colonnes du tableau. Il ignore la troisième colonne, Commentaire.

Chaque ligne du fichier contient le numéro d'index d'un paramètre dans la première colonne et la valeur attribuée à ce paramètre, dans la deuxième colonne. La valeur est représentée par une nombre flottant en double précision à l'intérieur de l'interpréteur, mais le point décimal n'est pas exigé dans le fichier. Le format des paramètres visible dans le tableau 10.4 est obligatoire et doit être utilisé pour tous les fichiers de paramètres, à l'exception des paramètres représentant une valeur sur un axe rotatif inutilisé, qui peuvent être omis. Une erreur sera signalée si un paramètre requis est absent. Un fichier de paramètres peut inclure tout autre paramètre, tant que son numéro est

TAB. 10.3 – Paramètres utilisés par l'interpréteur RS274NGC

Numéros de paramètre	Signification
5061-5070	Résultats des mesures "G38.2"
5161-5169	"G28" Origines machine
5181-5189	"G30" Origines pièce
5211-5219	"G92" Décalages d'origines
5220	Numéro du système de coordonnées
5221-5229	Système de coordonnées 1
5241-5249	Système de coordonnées 2
5261-5269	Système de coordonnées 3
5281-5289	Système de coordonnées 4
5301-5309	Système de coordonnées 5
5321-5329	Système de coordonnées 6
5341-5349	Système de coordonnées 7
5361-5369	Système de coordonnées 8
5381-5389	Système de coordonnées 9
5399	Résultat de M66 - Surveillance d'entrée

compris dans une fourchette de 1 à 5400. Les numéros de paramètre doivent être disposés dans l'ordre croîssant. Sinon, une erreur sera signalée. Le fichier original est copié comme fichier de sauvegarde lorsque le nouveau fichier est écrit. Les commentaires ne sont pas conservés lorsque le fichier est écrit.

TAB. 10.4 – Format d'un fichier de paramètres

Numéro de paramètre	Valeur du paramètre	Commentaire
5161	0.0	G28 pom X
5162	0.0	G28 pom Y

10.6 Systèmes de coordonnées

Dans le langage RS274/NGC, un centre d'usinage dispose d'un système de coordonnées machine (absolues) et de 9 systèmes de coordonnées programme (relatives).

Vous pouvez définir le décalage d'origine de neuf systèmes de coordonnées programme en utilisant G10 L2 Pn (n étant le numéro du système de coordonnées), avec des valeurs pour les axes en coordonnées absolues. Voir la section 12.6.

Vous pouvez choisir un des neuf systèmes en utilisant `G54`, `G55`, `G56`, `G57`, `G58`, `G59`, `G59.1`, `G59.2`, ou `G59.3` (Voir la section 12.14). Il n'est pas possible de choisir directement le système de coordonnées machine.

Vous pouvez décaler l'origine du système de coordonnées actuel en utilisant `G92` ou `G92.3`. Ce décalage s'appliquera alors à l'ensemble des neuf systèmes de coordonnées programme. Ce décalage peut être annulé avec `G92.1` ou `G92.2`. Voir la section 12.19.

Vous pouvez faire des mouvements dans le système de coordonnées machine absolues en utilisant `G53` avec `G0` ou `G1`. Voir la section 12.13.

Les données des systèmes de coordonnées sont enregistrées dans les paramètres.

Lors de l'initialisation, le système de coordonnées choisi est celui spécifié par le paramètre 5220. Une valeur de 1 signifie le premier système de coordonnées (celui qu'active G54), une valeur de 2 signifie le deuxième système de coordonnées (celui qu'active G55), et ainsi de suite. Il y a erreur si la valeur du paramètre 5220 n'est pas un nombre entier compris entre 1 et 9.

Chapitre 11

Vue d'ensemble du langage

Le langage RS274/NGC est basé sur des lignes de code. Chaque ligne (également appelée un "block") peut inclure des commandes pour faire produire diverses actions au centre d'usinage. Plusieurs lignes de code peuvent être regroupées dans un fichier pour créer un programme.

Une ligne de code typique commence par un numéro de ligne optionnel suivi par un ou plusieurs "mots". Un mot commence par une lettre suivie d'un nombre (ou quelque chose qui permet d'évaluer un nombre). Un mot peut, soit donner une commande, soit fournir un argument à une commande. Par exemple, "`G1 X3`" est une ligne de code valide avec deux mots. "`G1`" est une commande qui signifie "déplaces toi en ligne droite à la vitesse programmée" et "`X3`" fournit la valeur d'argument (la valeur de X doit être 3 à la fin du mouvement). La plupart des commandes RS274/NGC commencent avec G ou M (G pour Général et M pour Miscellaneous (auxiliaire)). Les termes pour ces commandes sont "G codes" et "M codes."

Le langage RS274/NGC n'a pas d'indicateur de début et de fin de programme. L'interpréteur cependant traite les fichiers. Un programme simple peut être en un seul fichier, mais il peut aussi être partagé sur plusieurs fichiers. Un fichier peut être délimité par le signe pourcent de la manière suivante. La première ligne non vide d'un fichier peut contenir un signe "%" seul, éventuellement encadré d'espaces blancs, ensuite, à la fin du fichier on doit trouver une ligne similaire. Délimiter un fichier avec des % est facultatif si le fichier comporte un `M2` ou un `M30`, mais est requis sinon. Une erreur sera signalée si un fichier a une ligne pourcent au début, mais pas à la fin. Le contenu utile d'un fichier délimité par pourcent s'arrête après la seconde ligne pourcent. Tout le reste est ignoré.

Le langage RS274/NGC prévoit les deux commandes (`M2` ou `M30`) pour finir un programme. Le programme peut se terminer avant la fin du fichier. Les lignes placées après la fin d'un programme ne seront pas exécutées. L'interpréteur ne les lit même pas.

11.1 Format d'une ligne

Une ligne de code permise par la norme RS274/NGC est construite de la façon suivante, dans l'ordre avec la restriction à un maximum de 256 caractères sur la même ligne.

1. Un block optionnel contenant le caractère d'effacement de ligne, barre de fraction "/".
2. Un numéro de ligne optionnel.
3. N'importe quel nombre de mots, valeurs de paramètres et commentaires.
4. Un caractère de fin de ligne (retour chariot ou saut de ligne ou les deux).

Toute entrée non explicitement permise est illégale, elle provoquera un message d'erreur de l'interpréteur.

Les espaces sont permis ainsi que les tabulations dans une ligne de code dont ils ne changent pas la signification, excepté dans les commentaires. Ceci peut donner d'étranges lignes, mais elles sont autorisées. La ligne "`g0x +0. 12 34y 7`" est équivalente à"`g0 x+0.1234 y7`", par exemple.

Les lignes vides sont permises, elles seront ignorées.

La casse des caractères est ignorée, excepté dans les commentaires. Toutes les lettres en dehors des commentaires peuvent être, indifféremment des majuscules ou des minuscules sans changer la signification de la ligne.

11.2 Numéro de ligne

Un numéro de ligne commence par la lettre N suivie d'un nombre entier non signé compris entre 0 et 99999 écrit avec moins de six caractères (par exemple : 000009 est interdit). Les numéros de ligne peuvent se suivre ou être dans le désordre, bien qu'une pratique normale évite ce genre d'usage. Les numéros de ligne peuvent être sautés, c'est une pratique normale. L'utilisation d'un numéro de ligne n'est pas obligatoire, mais si il est utilisé, il doit être à sa place.

11.3 Les mots

Un mot est une lettre, autre que N, suivie d'un nombre réel.

Les mots peuvent commencer avec l'une ou l'autre des lettres indiquées dans le tableau 11.1. Le tableau inclus N pour être complet, même si, comme défini précédemment, les numéros de lignes ne sont pas des mots. Plusieurs lettres (I, J, K, L, P, R) peuvent avoir différentes significations dans des contextes différents. Les lettres qui se réfèrent aux noms d'axes ne sont pas valides sur une machine n'ayant pas les axes correspondants.

11.3.1 Nombres

Les règles suivantes sont employées pour des nombres (explicites). Dans ces règles un chiffre est un caractère simple entre 0 et 9.

- Un nombre commence par : (1) un signe plus ou un signe moins optionnel, suivi par (2) de zéro à plusieurs chiffres, peut être suivis par, (3) un point décimal, suivi par (4) de zéro à plusieurs chiffres, il doit au moins y avoir un chiffre.
- Il existe deux types de nombres : les entiers et les décimaux. Un entier n'a pas de point décimal ; un décimal en a un.
- Les nombres peuvent avoir n'importe quel nombre de chiffres, sous réserve de la limitation de longueur d'une ligne. Seulement environ dix-sept chiffres significatifs seront retenus, c'est toutefois suffisant pour toutes les applications connues.
- Un nombre non nul sans autre signe que le premier caractère est considéré positif.

Les zéros non significatifs, ne sont pas nécessaires.

Si un nombre utilisé dans le langage RS274/NGC est proche d'une valeur entière à moins de quatre décimales, il est considéré comme entier, par exemple 0.9999.

11.3.2 Paramètres numérotés

Un paramètre numéroté commence par le caractère `#` suivi par un entier compris entre 1 et 5399. Le paramètre est référencé par cet entier, sa valeur est la valeur stockée dans le paramètre.

Une valeur est stockée dans un paramètre avec l'opérateur = par exemple "`#3 = 15`" signifie que la valeur 15 est stockée dans le paramètre numéro 3.

TAB. 11.1 – Les mots et leur signification

Lettre	Signification
A	Axe A de la machine
B	Axe B de la machine
C	Axe C de la machine
D	Valeur de la compensation de rayon d'outil
F	Vitesse d'avance travail
G	Fonction Générale (voir la table 5)
H	Index d'offset de longueur d'outil
I	Décalage en X pour les arcs et dans les cycles préprogrammés G87
J	Décalage en Y pour les arcs et dans les cycles préprogrammés G87
K	Décalage en Z pour les arcs et dans les cycles préprogrammés G87 Distance de déplacement par tour de broche avec G33
M	Fonction auxiliaire (voir la table 7)
N	Numéro de ligne
P	Temporisation utilisée dans les cycles préprogrammés et avec G4. Mot clé utilisé avec G10.
Q	Incrément Delta en Z dans un cycle préprogrammé G83
R	Rayon d'arc ou plan de retrait dans un cycle préprogrammé
S	Vitesse de rotation de la broche
T	Numéro d'outil
U	Axe U de la machine
V	Axe V de la machine
W	Axe W de la machine
X	Axe X de la machine
Y	Axe Y de la machine
Z	Axe Z de la machine

Le caractère # a une précédence supérieure à celle des autres opérations, ainsi par exemple, "#1+2" signifie la valeur trouvée en ajoutant 2 à la valeur contenue dans le paramètre 1 et non la valeur trouvée dans le paramètre 3. Bien sûr, #[1+2] signifie la valeur trouvée dans le paramètre 3. Le caractère # peut être répété, par exemple ##2 signifie le paramètre dont le numéro est égal à la valeur entière trouvée dans le paramètre 2.

11.3.3 Paramètres nommés

Les paramètres nommés fonctionnent comme les paramètres numérotés mais sont plus faciles à lire. Les paramètres nommés sont convertis en minuscules, les espaces et tabulations sont supprimés. Les paramètres nommés doivent être encadrés des signes < et >.

#<Un paramètre nommé> est un paramètre nommé local. Par défaut, un paramètre nommé est local à l'étendue dans laquelle il est assigné. L'accès à un paramètre local, en dehors de son sous-programme est impossible, de sorte que deux sous-programmes puissent utiliser le même nom de paramètre sans craindre qu'un des deux n'écrase la valeur de l'autre.

#<_un paramètre global> est un paramètre nommé global. Ils sont accessibles depuis des sous-programmes appelés et peuvent placer des valeurs dans tous les sous-programmes accessibles à l'appelant. En ce qui concerne la portée, ils agissent comme des paramètres numérotés. Ils ne sont pas enregistrés dans des fichiers.

Exemples :

– Déclaration d'une variable nommée globale

```
#<_troisdents_dia> = 10.00
```

– Référence à la variable globale précédemment déclarée

```
#<_troisdents_rayon> = [#<_troisdents_dia>/2.0]
```

– Mélange de paramètres nommés et de valeurs littérales

```
o100 call [0.0] [0.0] [#<_interieur_decoupe>-#<_troisdents_dia>] [#<_Zprofondeur>]
[#<_vitesse>]
```

Notes :

Les paramètres globaux _a, _b, _c, ... _z sont réservés pour une utilisation spéciale. Dans le futur, ils pourront fournir l'accès aux derniers Aword, Bword, Cword, ... Zword etc.

11.3.4 Expressions

Une expression est un groupe de caractères commençant avec le crochet gauche `[` et se terminant avec le crocher droit `]`. Entre les crochets, on trouve des nombres, des valeurs de paramètre, des opérations mathématiques et d'autres expressions. Une expression est évaluée pour produire un nombre. Les expressions sur une ligne sont évaluées quand la ligne est lue et avant que quoi que ce soit ne soit éxécuté sur cette ligne. Un exemple d'expression : `[1 + acos[0] - [#3 ** [4.0/2]]]`.

11.3.5 Opérateurs binaires

Les opérateurs binaires ne se rencontrent que dans les expressions. Il y a quatre opérateurs mathématiques de base : addition (`+`), soustraction (`-`), multiplication (`*`) et division (`/`). Il y a trois opérateurs logiques : ou (`OR`), ou exclusif (`XOR`) et logique (`AND`). Le huitième opérateur est le modulo (`MOD`). Le neuvième opérateur est l'élévation à la puissance (`**`) qui éléve le nombre situé à sa gauche à la puissance du nombre situé à sa droite. Les opérateurs de relation sont : égalité (`EQ`), non égalité (`NE`), strictement supérieur (`GT`), supérieur ou égal (`GE`), strictement inférieur (`LT`) et inférieur ou égal (`LE`).

Les opérations binaires sont divisées en plusieurs groupes selon leur précédence. (voir la table 11.2) Si dans une opération se trouvent différents groupes de précédence (par exemple dans l'expression `[2.0 / 3 * 1.5 - 5.5 / 11.0]`), les opérations du groupe supérieur seront effectuées avant celles des groupes inférieurs. Si une expression contient plusieurs opérations du même groupe (comme les premiers `/` et `*` dans l'exemple), l'opération de gauche est effectuée en premier. Notre exemple est équivalent à : `[[[2.0 / 3] * 1.5] - [5.5 / 11.0]]` , qui est équivalent à `[1.0 - 0.5]` , le résultat est : `0.5` .

Les opérations logiques et le modulo sont exécutés sur des nombres réels et non pas seulement sur des entiers. Le zéro est équivalent à un état logique faux (FALSE), tout nombre différent de zéro est équivalent à un état logique vrai (TRUE).

TAB. 11.2 – Précédence des opérateurs

Opérateurs	Précédence
**	*haute*
* / MOD	
+ -	
EQ NE GT GE LT LE	
AND OR XOR	*basse*

11.3.6 Fonctions

Une fonction commence par son nom, ex : "`ATAN`" suivi par une expression divisée par une autre expression (par exemple "`ATAN[2]/[1+3]`") ou tout autre nom de fonction suivi par une expression

(par exemple "SIN[90]"). Les fonctions disponibles sont visibles dans le tableau 11.3. Les arguments pour les opérations unaires sur des angles (COS, SIN et TAN) sont en degrés. Les valeurs retournées par les opérations sur les angles (ACOS, ASIN et ATAN) sont également en degrés.

L'opérateur FIX arrondi vers la gauche (moins positif ou plus négatif), par exemple : FIX[2.8] =2 et FIX[-2.8] = -3 . L'opérateur FUP arrondi vers la droite (plus positif ou moins négatif), par exemple : FUP[2.8] = 3 et FUP[-2.8] = -2 .

TAB. 11.3 – Fonctions

Nom de fonction	Fonction
ATAN[Y]/[X]	Tangente quatre quadrants
ATAN[arg]	Arc tangente
ABS[arg]	Valeur absolue
ACOS[arg]	Arc cosinus
ASIN[arg]	Arc sinus
ATAN[arg]	Arc tangente
COS[arg]	Cosinus
EXP[arg]	Exposant
FIX[arg]	Arrondi à l'entier immédiatement inférieur
FUP[arg]	Arrondi à l'entier immédiatement supérieur
ROUND[arg]	Arrondi à l'entier le plus proche
LN[arg]	Logarithme Néperien
SIN[arg]	Sinus
SQRT[arg]	Racine carrée
TAN[arg]	Tangente

11.4 Commentaires

Un ensemble de caractères et espaces blancs entre parenthèses est un commentaire. Une parenthèse ouvrante débute toujours un commentaire. Le commentaire se termine à la première parenthèse fermante trouvée. Si une parenthèse ouvrante est trouvée sur une ligne, une parenthèse fermante doit être également rencontrée avant la fin de la ligne. Les commentaires ne peuvent pas être imbriqués, une erreur sera signalée si une parenthèse ouvrante est rencontrée après le début d'un commentaire et avant la fin d'un commentaire. Voici un exemple de ligne de commentaire : "G80 M5 (arret du mouvement)". Les commentaires sont seulement informatifs, ils n'ont aucune influence sur la machine.

11.4.1 Messages

Un commentaire contient un message si " MSG" apparaît après la parenthèse ouvrante, et avant tout autre caractère. Les variantes de "MSG" qui incluent un espace blanc et des minuscules sont autorisées. Le reste du texte avant la parenthèse fermante est considéré comme le message. Les messages sont affichés sur la visu de l'interface utilisateur. Les commentaires ne contenant pas de message ne sont pas affichés.

11.4.2 Log des mesures

Un commentaire peut aussi être utilisé pour spécifier le fichier de log des résultats des mesures faites avec G38.x . Voir la section 12.10.

11.4.3 Log général

11.4.3.1 (LOGOPEN,filename)

Ouvre le fichier de log "filename". Si le fichier existe déjà, il sera tronqué.

11.4.3.2 (LOGCLOSE)

Si le fichier est ouvert, il sera fermé.

11.4.3.3 (LOG,...)

Le message "..." est étendu comme décrit plus loin, il est écrit dans le fichier de log si il est ouvert.

11.4.4 Messsages de déboguage

Les commentaires comme : `(debug, reste du commentaire)` sont traités de la même façon que ceux avec `(msg, reste du commentaire)` avec l'ajout de possibilités spéciales pour les paramètres.

Les commentaires comme : `(print, reste du commentaire)` vont directement sur la sortie stderr avec des possibilités spéciales pour les paramètres.

11.4.5 Paramètres dans les commentaires

Dans les commentaires avec DEBUG, PRINT et LOG, les valeurs des paramètres dans le message sont étendues.

Par exemple : pour afficher une variable nommée globale sur la sortie stderr (la fenêtre de la console par défaut) ajouter une ligne au g-code comme :

```
(print,diamètre fraise 3 dents = #<_troisdents_dia>)
```

À l'intérieur de ces types de commentaires, les séquences comme `#123` sont remplacées par la valeur du paramètre 123. Les séquences comme `#<paramètre nommé>` sont remplacées par la valeur du paramètre nommé. Rappelez vous que les espaces dans les noms des paramètres nommés sont supprimés, `#<parametre nomme>` est équivalent à `#<parametrenomme>`.

11.5 Répétitions d'items

Une ligne peut contenir autant de mots G que voulu, mais seulement deux mots G du même groupe modal peuvent apparaître sur la même ligne. (voir la section 11.8)

Une ligne peut avoir de zéro à quatre mots M. Mais pas deux mots M du même groupe modal.

Pour toutes les autres lettres légales, un seul mot commençant par cette lettre peut se trouver sur la même ligne.

Si plusieurs valeurs de paramètre se répètent sur la même ligne, par exemple : "`#3=15 #3=6`", seule la dernière valeur prendra effet. Il est absurde, mais pas illégal, de fixer le même paramètre deux fois sur la même ligne.

Si plus d'un commentaire apparaît sur la même ligne, seul le dernier sera utilisé, chacun des autres sera lu et son format vérifié, mais il sera ignoré. Placer plusieurs commentaires sur la même ligne est très rare.

11.6 Ordre des items

Les trois types d'item dont la commande peut varier sur une ligne (comme indiqué au début de cette section) sont les mots, les paramètres et les commentaires. Imaginez que ces trois types d'éléments sont divisés en trois groupes selon leur type.

Dans le premier groupe les mots, peuvent être arrangés dans n'importe quel ordre sans changer la signification de la ligne.

Dans le second groupe les valeurs de paramètre, quelque soit leur arrangement, il n'y aura pas de changement dans la signification de la ligne sauf si le même paramètre est présent plusieurs fois. Dans ce cas, seule la valeur du dernier paramètre prendra effet. Par exemple, quand la ligne "`#3=15 #3=6`" aura été interprétée, la valeur du paramètre 3 vaudra 6. Si l'ordre est inversé, "`#3=6 #3=15`" après interprétation, la valeur du paramètre 3 vaudra 15.

Enfin dans le troisième groupe les commentaires, si plusieurs commentaires sont présents sur une ligne, seul le dernier commentaire sera utilisé.

Si chaque groupe est laissé, ou réordonné, dans l'ordre recommandé, la signification de la ligne ne changera pas, alors les trois groupes peuvent être entrecroisés n'importe comment sans changer la signification de la ligne. Par exemple, la ligne "`g40 g1 #3=15 (foo) #4=-7.0`" à cinq items est signifiera exactement la même chose dans les 120 ordres d'arrangement possibles des cinq items comme "`#4=-7.0 g1 #3=15 g40 (foo)`".

11.7 Commandes et modes machine

En RS274/NGC, de nombreuses commandes produisent, d'un mode à un autre, quelque chose de différent au niveau de la machine, le mode reste actif jusqu'à ce qu'une autre commande ne le révoque, implicitement ou explicitement. Ces commandes sont appelées "modales". Par exemple, si l'arrosage est mis en marche, il y reste jusqu'à ce qu'il soit explicitement arrêté. Les G-codes pour les mouvements sont également modaux. Si, par exemple, une commande G1 (déplacement linéaire) se trouve sur une ligne, elle peut être utilisée sur la ligne suivante avec seulement un mot d'axe, tant qu'une commande explicite est donnée sur la ligne suivante en utilisant des axes ou un arrêt de mouvement.

Les codes "non modaux" n'ont d'effet que sur la ligne ou ils se présentent. Par exemple, G4 (tempo) est non modale.

11.8 Groupes modaux

Les commandes modales sont arrangées par lots appelés "groupes modaux", à tout moment, un seul membre d'un groupe modal peut être actif. En général, un groupe modal contient des commandes pour lesquelles il est logiquement impossible que deux membres soient actifs simultanément, comme les unités en pouces et les unités en millimètres. Un centre d'usinage peut être dans plusieurs modes simultanément, si seulement un mode pour chaque groupe est actif. Les groupes modaux sont visibles dans le tableau 11.4.

Pour plusieurs groupes modaux, quand la machine est prête à accepter des commandes, un membre du groupe doit être en vigueur. Il y a des paramètres par défaut pour ces groupes modaux. Lorsque la machine est mise en marche ou ré-initialisées, les valeurs par défaut sont automatiquement actives.

Groupe 1, le premier groupe du tableau, est un groupe de G-codes pour les mouvements. À tout moment, un seul d'entre eux est actif. Il est appelé le mode de mouvement courant.

C'est une erreur que de mettre un G-code du groupe 1 et un G-code du groupe 0 sur la même ligne si les deux utilisent les mêmes axes. Si un mot d'axe utilisant un G-code du groupe 1 est

TAB. 11.4 – Groupes modaux

Signification du groupe modal	Mots des membres
Mouvements (“Groupe 1”)	G0 G1 G2 G3 G33 G38.x G80 G81 G82 G83 G84 G85 G86 G87 G88 G89
Choix du plan de travail	G17 G18 G19
Mode de déplacements	G90 G91
Mode de vitesses	G93, G94
Unités machine	G20, G21
Correcteurs de rayon d'outil	G40, G41, G42, G41.1, G42.1
Correcteurs de longueur d'outil	G43, G43.1, G49
Options de retrait des cycles préprogrammés	G98, G99
Systèmes de coordonnées	G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3
Types d'arrêt de programme	M0, M1, M2, M30, M60
Appel d'outil	M6
Types de rotation de la broche	M3, M4, M5
Arrosages	M7, M8, M9. Cas spéciaux : M7 et M8 peuvent être actifs en même temps
Boutons de correction de vitesse	M48, M49
Contrôle de flux	O-
Codes non modaux (“Groupe 0”)	G4, G10, G28, G30, G53 G92, G92.1, G92.2, G92.3 M100 à M199

implicitement actif sur la ligne (en ayant été activé sur une ancienne ligne) et qu'un G-code du groupe 0 utilisant des mots d'axes apparaît sur la même ligne, l'activité du G-code du groupe 1 est révoquée pour le reste de la ligne. Les mots d'axes utilisant des G-codes du groupe 0 sont G10, G28, G30 et G92.

C'est une erreur d'inclure des mots sans rapport sur une ligne avec le contrôle de flux O.

Chapitre 12

G Codes

Les G-codes du langage RS274/NGC sont décrits ci-dessous.

Dans une commande type, le tiret (–) signifie une valeur réelle. Comme décrite précédemment, une valeur réelle peut être (1) un nombre explicite, 4 par exemple, (2) une expression, [2+2] par exemple, (3) une valeur de paramètre, #88 par exemple, ou (4) une fonction unaire de la valeur acos [0], par exemple.

Dans la plupart des cas, si des mots d'axes sont donnés, parmi `X`, `Y`, `Z`, `A`, `B`, `C`, `U`, `V`, `W`, ils spécifient le point de destination. Les axes sont donnés dans le système de coordonnées courant, à moins qu'explicitement décrit comme étant dans le système de coordonnées absolues (machine). Où les axes sont optionnels, tout axe omis gardera sa valeur courante. Tout item dans une commande non explicitement décrit comme optionnel sera requis. Une erreur sera signalée si un item requis est omis.

Dans les commandes, les valeurs suivant les lettres sont souvent données comme des nombres explicites. Sauf indication contraire, les nombres explicites peuvent être des valeurs réelles. Par exemple, G10 L2 pourrait aussi bien être écrite G [2 * 5] L [1 +1]. Si la valeur du paramètre 100 étaient 2, G10 L#100 signifierait également la même chose. L'utilisation de valeurs réelles qui ne sont pas des nombres explicites, comme indiqué dans les exemples sont rarement utiles.

Si `L–` est écrit dans une commande le "–" fera référence à "L nombre". De la même manière, le "–" dans `H–` peut être appelé le "H nombre" et ainsi de suite pour les autres lettres.

12.1 G0 : Interpolation linéaire en vitesse rapide

Pour un mouvement linéaire en vitesse rapide, programmer `G0` *axes*, au moins un mot d'axe doit être présent, les autres sont optionnels. Le `G0` est optionnel si le mode mouvement courant est déjà `G0`. Cela produira un mouvement linéaire vers le point de destination à la vitesse rapide courante (ou moins vite si la machine n'atteint pas cette vitesse). Il n'est pas prévu d'usiner la matière quand une commande G0 est exécutée.

C'est une erreur si :

– tous les axes sont omis.

Si la compensation de rayon d'outil est active, le mouvement sera différent de celui décrit ci-dessus, voir la section 18.4. Si `G53` est programmé sur la même ligne, le mouvement sera également différent, voir la section 12.13.

Si un mouvement `G0` déplace seulement des axes rotatifs et que la position de la cible pour ces axes est dans une échelle de -360 à 360 degrés, le mouvement sera organisé pour que chaque axe rotatif fasse moins d'un tour complet. Exemple :

```
N1 G0 X1 A[20*360]
N2 G0 A0
```

Après la ligne N1, la position de l'axe A sera 7200 degrés. 7200 degrés est égal à zéro degré sur un axe rotatif, de sorte que le mouvement linéaire rapide spécifié en ligne N2 ne produira aucun mouvement.

12.2 G1 : Interpolation linéaire en vitesse travail

Pour un mouvement linéaire en vitesse travail, programmer G1 *axes*, au moins un mot d'axe doit être présent, les autres sont optionnels. Le G1 est optionnel si le mode mouvement courant est déjà G1. Cela produira un mouvement linéaire vers le point de destination à la vitesse de travail courante (ou moins vite si la machine n'atteint pas cette vitesse).

C'est une erreur si :

– tous les axes sont omis.

Si la compensation de rayon d'outil est active, le mouvement sera différent de celui décrit ci-dessus, voir la section 18.4. Si G53 est programmé sur la même ligne, le mouvement sera également différent, voir la section 12.13.

12.3 G2, G3 : Interpolation circulaire en vitesse travail

Un mouvement circulaire ou hélicoïdal est spécifié en sens horaire avec G2 ou en sens anti-horaire avec G3. Les axes du cercle ou de l'hélicoïde, doivent être parallèles aux axes X, Y, ou Z du système de coordonnées machine. Les axes (ou, leurs équivalents, les plans perpendiculaires aux axes) sont sélectionnés avec G17 (axe Z, plan XY), G18 (axe Y, plan XZ), ou G19 (axe X, plan YZ). Si l'arc est circulaire, il se trouve dans un plan parallèle au plan sélectionné.

Si une ligne de code RS274/NGC forme un arc et inclu le mouvement d'un axe rotatif, l'axe rotatif tournera à vitesse constante, de sorte que le mouvement de l'axe rotatif commence et se termine en même temps que les autres axes XYZ. De telles lignes ne sont pratiquement jamais programmées.

Si la compensation de rayon d'outil est active, le mouvement sera différent de celui décrit ci-dessus, voir la section 12.11.

Deux formats sont autorisés pour spécifier un arc : le format centre et le format rayon.

12.3.1 Arc au format centre (format recommandé)

Dans le format centre, les coordonnées du point final de l'arc, dans le plan choisi, sont spécifiées par décalage du centre de l'arc depuis le point courant. Dans ce format, il est permis d'avoir le point final de l'arc identique au point courant. C'est une erreur si :

– Lorsque l'arc est projeté sur le plan choisi, la distance entre le point courant et le centre diffèrent de la distance entre le point d'arrivée et le centre de plus de 0,0002 pouce (si les pouces sont utilisés), soit 0,002 millimètre (si les millimètres sont utilisés).

Lorsque le plan XY est sélectionné, programmer G2 Axe I- J- (ou utiliser G3 au lieu de G2). Les mots d'axes sont tous facultatifs sauf au moins un entre X et Y qui doit être utilisé. I et J sont les décalages du centre de l'arc, par rapport au point actuel (dans le sens X et Y, respectivement). I et J sont facultatifs, sauf qu'au moins un des deux doit être utilisé. C'est une erreur si :

– X et Y sont tous les deux omis.
– I et J sont tous les deux omis .

Lorsque le plan XZ est sélectionné, programmer `G2 axes I- K-` (ou utiliser `G3` au lieu de `G2`). Les mots d'axes sont tous facultatifs sauf au moins un entre X et Z qui doit être utilisé. I et K sont les décalages du centre de l'arc, par rapport au point actuel (dans le sens X et Z , respectivement). I et K sont facultatifs, sauf qu'au moins un des deux doit être utilisé. C'est une erreur si :

– X et Z sont tous les deux omis.
– I et K sont tous les deux omis .

Lorsque le plan YZ est sélectionné, programmer `G2 axes J- K-` (ou utiliser `G3` au lieu de `G2`). Les mots d'axes sont tous facultatifs sauf au moins un entre Y et Z qui doit être utilisé. J et K sont les décalages du centre de l'arc, par rapport au point actuel (dans le sens Y et Z , respectivement). J et K sont facultatifs, sauf qu'au moins un des deux doit être utilisé. C'est une erreur si :

– Y et Z sont tous les deux omis.
– J et K sont tous les deux omis .

Voici un exemple de commande pour usiner un arc au format centre : `G17 G2 x10 y16 i3 j4 z9`.

Cela signifie faire un mouvement en arc ou hélicoïdal en sens horaire (vu du côté positif de l'axe Z), se terminant en X=10, Y=16 et Z=9, avec son centre décalé du point actuel de 3 unités en X et de 4 unités en Y. Si la position courante est X=7, Y=7 au final, le centre sera en X=10, Y=11. Si la position de départ de l'axe Z est 9, ce sera un arc circulaire, autrement ce sera un arc hélicoïdal. Le rayon de l'arc sera de 5.

Dans le format centre, le rayon de l'arc n'est pas spécifié, mais il peut facilement être trouvé comme étant la distance entre le centre de l'arc et soit, son point d'arrivée soit, le point courant.

12.3.2 Arcs au format rayon (format non recommandé)

Dans le format rayon, les coordonnées du point final de l'arc, dans le plan choisi, sont spécifiées en même temps que le rayon de l'arc. Programmer G2 axes R- (ou utiliser `G3` au lieu de `G2`). R est le rayon. Les mots d'axes sont facultatifs sauf au moins un des deux du plan choisi, qui doit être utilisé. Un rayon positif indique que l'arc fait moins de 180 degrés, alors qu'un rayon négatif indique un arc supérieur à 180 degrés. Si l'arc est hélicoïdal, la valeur du point d'arrivée de l'arc dans les coordonnées de l'axe perpendiculaire au plan choisi sera également spécifiée.

C'est une erreur si :

– Les deux mots d'axes pour le plan choisi sont omis.
– Le point d'arrivée de l'arc est identique au point courant.

Ce n'est pas une bonne pratique de programmer au format rayon, des arcs qui sont presque des cercles entiers ou des demi-cercles, car un changement minime dans l'emplacement du point d'arrivée va produire un changement beaucoup plus grand dans l'emplacement du centre du cercle (et donc, du milieu de l'arc). L'effet de grossissement est tellement important, qu'une erreur d'arrondi peut facilement produire un usinage hors tolérance. Par exemple, 1% de déplacement de l'extrémité d'un arc de 180 degrés produit 7% de déplacement du point situé à 90 degrés le long de l'arc. Les cercles presque complets sont encore pires. Autrement l'usinage d'arcs dans une gamme de petits à 165 degrés ou de 195 à 345 degrés sera bon.

Voici un exemple de commande pour usiner un arc au format `rayon : G17 G2 x 10 y 15 r 20 z 5`.

Cela signifie faire un mouvement en arc ou hélicoïdal en sens horaire (vu du côté positif de l'axe Z), se terminant en X=10, Y=15 et Z=5, avec un rayon de 20. Si la valeur de départ de Z est 5, Ce sera un arc de cercle parallèle au plan XY sinon, ce sera un arc hélicoïdal.

12.4 G33, G33.1 : Mouvement avec broche synchronisée

Pour un mouvement avec broche synchronisée dans une direction, programmer `G33 X- Y- Z- K-` où K donne la longueur du mouvement en XYZ pour chaque tour de broche. Par exemple, il commence à `Z=0`, `G33 Z-1 K.0625` produira un mouvement d'un pouce de long en Z, et en même temps

16 tours de broche. Cette commande pourrait être la base d'un programme pour faire un filetage de 16 filets par pouce. Un autre exemple en métrique, G33 Z-15 K1.5 produira un mouvement de 15mm de long pendant que la broche fera 10 tours soit un pas de 1.5mm.

Pour un taraudage rigide avec broche synchronisée et mouvement de retour, programmer `G33.1 X- Y- Z- K-` où `K-` donne la longueur du mouvement pour chaque tour de broche. Un mouvement de taraudage rigide suit cette séquence :

- Un mouvement aux coordonnées spécifiées, synchronisé avec la rotation de la broche, avec un ratio donné et débutant sur l'impulsion d'index du codeur de la broche.
- Quand le point final est atteint, la commande inverse le sens de rotation de la broche (ex : de 300 tours/mn en sens horaire à 300 tours/mn en sens anti-horaire)
- Le mouvement reste synchronisé en continu avec la broche même **au delà** de la coordonnée du point final spécifiée pendant l'arrêt de la broche et son inversion.
- Le mouvement synchronisé se poursuit pour revenir aux coordonnées initiales.
- Quand les coordonnées d'origines sont atteintes, la commande inverse la broche une seconde fois (ex : de 300tr/mn sens anti-horaire à 300tr/mn sens horaire)
- Le mouvement reste synchronisé même **au delà** des coordonnées d'origine pendant que la broche s'arrête et s'inverse.
- Un mouvement **non synchronisé** ramène le mobile en arrière, aux coordonnées d'origine.

Tous les mouvements avec broche synchronisée ont besoin d'un index de broche, pour conserver la trajectoire prévue. Un mouvement avec `G33` se termine au point final programmé, un mouvement avec `G33.1` se termine aux coordonnées initiales.

Les mots d'axes sont facultatifs, sauf au moins un qui doit être utilisé.

C'est une erreur si :

- Tous les axes sont omis.
- La broche ne tourne pas quand cette commande est exécutée.
- Le mouvement linéaire requis excède les limites de vitesse machine en raison de la vitesse de broche.

12.5 G4 : Tempo

Pour une tempo, programmer G4 P- . Les axes s'immobiliseront pour une durée de P secondes. C'est une erreur si :

- Le nombre P est négatif.

12.6 G10 : Établissement du système de coordonnées pièce

Le langage RS274/NGC utilise les systèmes de coordonnées décrits à la section10.6.

Pour définir les coordonnées de l'origine d'un système de coordonnées pièce, programmer `G10 L2 P-` *`axes`*, où l'entier P, compris entre 1 et 9 (correspondance de `G54` à `G59.3`) et tous les mots d'axes, sont optionnels. Les coordonnées de l'origine du système de coordonnées, spécifié par le nombre P, sont remplacées par les coordonnées des valeurs indiquées (en termes de coordonnées absolues). Seules, les coordonnées pour lesquelles un mot d'axe est inclu sur la ligne, seront remplacées.

C'est une erreur si :

- Le nombre P n'est pas un entier compris entre 1 et 9.

Si des décalages d'origine (créés avec `G92` ou `G92.3`) sont actifs avant l'utilisation de `G10`, ils restent actifs après.

Le système de coordonnées dont l'origine est définie par la commande G10 peut être actif ou non au moment de l'exécution de `G10`.

Exemple : `G10 L2 P1 x 3.5 y 17.2` place l'origine du premier système de coordonnées (celui selectionné par G54) au point où X vaut 3.5 et Y vaut 17.2 (en coordonnées absolues). La coordonnée Z de l'origine, ainsi que les coordonnées de tous les axes rotatifs, restent celles qu'elles étaient avant l'exécution de la ligne.

12.7 G17, G18, G19 : Choix du plan de travail

Programmer G17 pour choisir le plan de travail XY, G18 pour choisir le plan de travail XZ, ou G19 pour choisir le plan de travail YZ. Les effets provoqués par un changement de plan de travail sont expliqués dans les sections 12.3 et 12.18

12.8 G20, G21 : Choix des unités machine

Programmer G20 pour utiliser le pouce comme unité de longueur. Programmer G21 pour utiliser le millimètre.

C'est toujours une bonne pratique de programmer soit G20, soit G21, au début d'un programme, avant tout mouvement et de ne plus en changer ailleurs dans le programme. C'est la responsabilité de l'opérateur d'être sûr que toutes les longueurs sont appropriées pour l'utilisation des unités actuelles.

12.9 G28, G30 : Retour à une position absolue prédéfinie

Deux positions sont définies, une par les paramètres 5161-5166 pour `G28` et une par les paramètres 5181-5186 pour `G30`. Les valeurs de ces paramètres sont en coordonnées absolues et en coordonnées machine.

Les commandes G28 et G30 n'utilisent pas de contact d'origine machine pour trouver la position prédéfinie, elles se bornent à effectuer un mouvement en vitesse rapide à la position définie dans les paramètres, ce qui suppose que la prise d'origine machine a déjà été réalisée.

Pour le retour d'un ou plusieurs axes sur une position prédéfinie par le biais de la position programmée, programmer `G28` *axes* (ou utiliser `G30`). La trajectoire est faite par un mouvement rapide depuis la position courante jusqu'à la position programmée, suivi d'un mouvement rapide des axes nommés à la position prédéfinie.

Pour le retour de tous les axes à la position prédéfinie sans la position intermédiaire, programmer `G28` ou `G30` sans mot d'axe.

C'est une erreur si :

– La compensation de rayon d'outil est active.

12.10 G38.2 : Sonde de mesure

Programmer `G38.2` *axes*, pour effectuer une mesure à la sonde. Au moins un mot d'axe est obligatoire, les autres sont optionnels. L'outil dans la broche doit être une sonde.

C'est une erreur si :

– Le point programmé est le même que le point courant.
– Aucun mot d'axe n'est utilisé.
– La compensation de rayon d'outil est active.

– La vitesse travail est à zéro.
– La sonde est déjà au contact de la cible.

En réponse à cette commande, la machine déplace le point contrôlé (qui est la pointe de la sonde), en ligne droite, à la vitesse actuelle, vers le point programmé, en mode vitesse inverse du temps ; la vitesse est telle, que le mouvement depuis le point courant jusqu'au point programmé, prendra le temps spécifié. Le mouvement s'arrête lorsque le point programmé est atteint, ou quand l'entrée de la sonde bascule dans l'état attendu[1].

12.11 G40, G41, G42, G41.1, G42.1 : Compensation de rayon d'outil

Pour désactiver la compensation de rayon d'outil, programmer `G40`. Ce n'est pas une erreur de désactiver la compensation quand elle est déjà inactive.

La compensation de rayon d'outil ne peut être activée que si le plan de travail courant est le plan XY.

Le comportement du centre d'usinage quand la compensation de rayon d'outil est active est décrit dans la section 18.4

12.11.1 Compensation de rayon d'outil depuis une table d'outils

Pour activer la compensation de rayon d'outil à gauche du profil, c'est à dire, la fraise reste à gauche de la trajectoire programmée quand le rayon d'outil est positif, programmer `G41 D-` . Pour activer la compensation de rayon d'outil à droite du profil, c'est à dire, la fraise reste à droite de la trajectoire programmée quand le rayon d'outil est positif, programmer `G42 D-` . Le mot D est optionnel, si il n'y a pas de mot D, le rayon de l'outil monté actuellement dans la broche est utilisé. Si D est utilisé, sa valeur devrait être le numéro de l'outil monté dans la broche, cependant, ce n'est pas obligatoire. Ce n'est pas une erreur d'avoir D à zéro, le rayon utilisé sera de zéro.

C'est une erreur si :

– La valeur de D n'est pas un entier, il est négatif ou supérieur au nombre d'emplacements dans le carrousel.
– Le plan de travail n'est pas YZ.
– La compensation de rayon d'outil est activée alors qu'elle est déjà active.

12.11.2 Compensation dynamique de rayon d'outil

Pour activer la compensation dynamique de rayon d'outil à gauche du profil, programmer `G41.1 D- L-`. Pour activer la compensation dynamique de rayon d'outil à droite du profil, programmer `G42.1 D- L-`. Le mot D spécifie le diamètre de l'outil. Le mot L spécifie l'orientation des arêtes de l'outil, et par défaut 0 si non spécifié.

C'est une erreur si :

– Le plan yz est le plan de travail actif.
– La valeur de L n'est pas comprise entre 0 et 9 inclus.
– La compensation de rayon d'outil est activée alors qu'elle est déjà active.

[1] En réalité, le mouvement continue légérement après que l'état de la sonde ait basculé, à cause des contraintes d'accélération/décélération de la machine.

12.12 G43, G43.1, G49 : Compensation de longueur d'outil

12.12.1 G43, G43.1 : Activation de la compensation de longueur d'outil

G43 et G43.1 modifient les mouvements ultérieurs en décalant, de la longueur de l'outil, les coordonnées de Z et/ou de X. G43 et G43.1 ne provoquent aucun mouvement. L'effet de la compensation ne se produira qu'au prochain mouvement de l'axe compensé.

12.12.1.1 G43 : Compensation de longueur d'outil depuis une table d'outils

Pour utiliser la compensation de longueur d'outil depuis une table d'outils, programmer `G43 H-` , où la valeur de H est l'index souhaité dans la table d'outils. La valeur de H n'est pas obligatoirement la même que celle du numéro d'outil monté dans la broche. Ce n'est pas une erreur d'avoir la valeur de H à zéro, une compensation de zéro sera utilisée.

C'est une erreur si :

– La valeur de H n'est pas un entier, il est négatif, ou il est supérieur au nombre d'outils dans le carrousel.

12.12.1.2 G43.1 : Compensation dynamique de longueur d'outil

Pour utiliser la compensation dynamique de longueur d'outil depuis un programme, utiliser `G43.1 I- K-`, où `I-` donne la compensation de longueur d'outil en X (pour les tours) et `K-` donne la compensation de longueur en Z (pour les tours et les fraiseuses).

C'est une erreur si :

– Une commande de mouvement est sur la même ligne que `G43.1`

12.12.2 G49 : Annulation de la compensation de longueur d'outil

Pour annuler la compensation de longueur d'outil, programmer G49.

Ce n'est pas une erreur de programmer une compensation qui est déjà utilisée. Ce n'est pas non plus une erreur d'annuler la compensation de longueur d'outil alors qu'aucune n'est couramment utilisée.

12.13 G53 : Mouvement en coordonnées absolues

Pour un déplacement à un point exprimé en coordonnées absolues, programmer `G1 G53 X- Y- Z- A- B- C-` (ou utiliser `G0` à la place de `G1`), au moins un mot d'axe est obligatoire, les autres sont optionnels. Le `G0` ou le `G1` est optionnel si il est déjà le mode de mouvement courant. `G53` n'est pas modal, il doit être programmé sur chaque ligne où il doit être actif. Il produit un mouvement linéaire coordonné au point programmé. Si `G1` est actif, la vitesse travail courante est utilisée si la machine est assez rapide. Si `G0` est actif, la vitesse rapide courante sera utilisée si la machine est assez rapide.

C'est une erreur si :

– G53 est utilisé sans que G0 ou G1 ne soit actif.
– G53 est utilisé alors que la compensation de rayon d'outil est active.

Voir la section 10.6 pour une vue complète des systèmes de coordonnées.

12.14 G54 à G59.3 : Choix du système de coordonnées

Le code G54 est apparié avec le système de coordonnées pièce N°1, pour le choisir programmer G54 et ainsi de suite pour les autres systèmes. Les systèmes de coordonnées appariés à un G-code sont les suivants : (1 avec G54), (2 avec G55), (3 avec G56), (4 avec G57), (5 avec G58), (6 avec G59), (7 avec G59.1), (8 avec G59.2) et (9 avec G59.3).

C'est une erreur si :

– Un de ces G-codes est utilisé alors que la compensation de rayon d'outil est active.

Voir la section 10.6 pour une vue complète des systèmes de coordonnées.

12.15 G61, G61.1, G64 : Types de contrôle de trajectoire

Programmer G61 pour passer la machine en mode de trajectoire exacte, G61.1 pour la passer en mode arrêt exact, ou G64 P- pour le mode trajectoire continue avec tolérance optionnelle. Ce n'est pas une erreur de programmer un mode déjà actif. Voir la section 10.2.15 pour une discussion de ces modes.

12.16 G80 : Révocation des codes modaux

Programmer G80 pour s'assurer qu'aucun mouvement d'axe ne surviendra sans G-code modal. C'est une erreur si :

– Des mots d'axes sont programmés quand G80 est actif, sans qu'un G-code modal du groupe 0 ne soit programmé avec les mots d'axes.

12.17 G76 : Cycle de filetage préprogrammé

Programmer G76 P- Z- I- J- R- K- Q- H- E- L- pour produire un cycle de filetage multi-passes. C'est une erreur si :

– Le plan de travail actif n'est pas ZX.
– D'autres mots d'axes comme X- ou Y-, sont spécifiés.
– La dégressivité R- est inférieure à 1.0.
– Certains mots requis ne sont pas spécifiés.
– P-, J-, K- ou H- est négatif.
– E- est supérieure à la moitié de la longueur de la ligne pilote.

La ligne pilote (drive line) est une ligne imaginaire, parallèle à l'axe de la broche, située en sécurité à l'extérieur du matériau à fileter. La ligne pilote va du point initial en Z jusqu'à la fin du filetage donnée par la valeur de Z- dans la commande.

Le pas du filet, ou la longueur du déplacement par tour est donné par la valeur de P-.

La crête du filet est donnée par la valeur de I- , c'est une cote entre la ligne pilote et la surface de la pièce. Une valeur négative de I-, indique un filetage externe et une valeur positive, indique un filetage interne. C'est généralement à ce diamètre nominal que le matériau est cylindré avant de commencer le cycle G76.

La profondeur de la passe initiale est donnée par la valeur de J-. La première passe sera à J- au delà de la crête du filet. J- est positif, même quand I- est négatif.

La profondeur du filet est donnée par la valeur de K-. La dernière passe du filetage sera à K- au delà de la crête du filet. K- est positif, même quand I- est négatif.

La dégressivité de la profondeur de passe est donnée par la valeur de R-. R1.0 sélectionne une profondeur de passe constante pour les passes successives du filetage. R2.0 sélectionne une surface constante. Les valeurs comprises entre 1.0 et 2.0 sélectionnent une profondeur décroîssante mais une surface croîssante. Enfin, les valeurs supérieures à 2.0 sélectionnent une surface décroîssante. Attention : les valeurs inutilement hautes de dégression, demanderont un nombre inutilement important de passes.

L'angle de pénétration oblique est donné par la valeur de Q- c'est l'angle (en degrés) décrivant de combien les passes successives doivent être décalées le long de l'axe. C'est utilisé pour faire enlever plus de matériau d'un côté de l'outil que de l'autre. Une valeur positive de Q fait couper d'avantage le bord principal de l'outil. Typiquement, les valeurs sont 29, 29.5 ou 30 degrés.

Le nombre de passes de finition est donné par la valeur de H-. Les passes de finition sont des passes additionnelles en fond de filet. Pour ne pas faire de passe de finition, programmer H0.

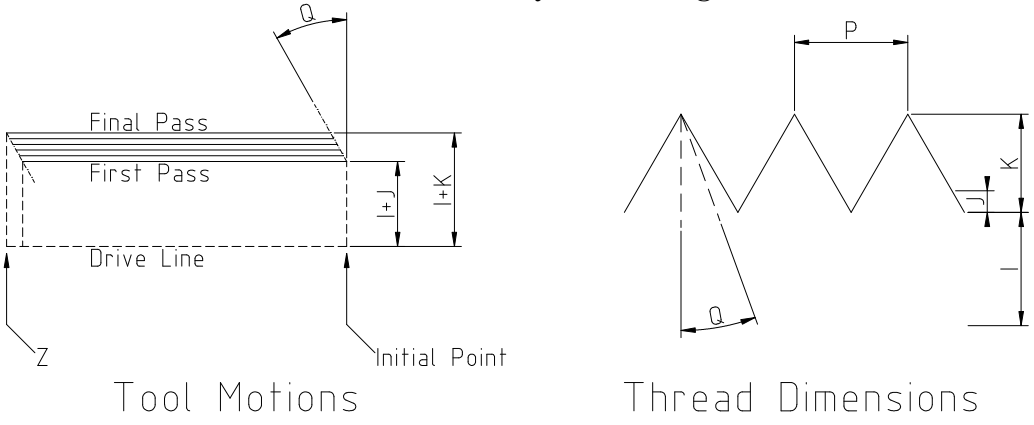
Les entrées et sorties de filetage peuvent être programmées côniques avec les valeurs de E- et L-. E- donne la longueur de la partie cônique le long de l'axe. E0.2 donnera un cône pour le premier et le dernier filet, de 0.2 unités dans le sens du filetage. L- est utilisé pour spécifier quelles extrémités du filetage doivent être côniques. Programmer L0 pour aucune (par défaut), L1 pour une entrée cônique, L2 pour une sortie cônique, ou L3 pour l'entrée et la sortie côniques.

L'outil fera une brève pause pour la synchronisation avant chaque passe de filetage, ainsi une gorge de dégagement sera requise à l'entrée, à moins que le début du filetage ne soit après l'extrémité de la pièce ou qu'un cône d'entrée soit utilisé.

À moins d'utiliser un cône de sortie, le mouvement de sortie (retour rapide sur X d'origine) n'est pas synchronisé sur la vitesse de broche. Avec une broche lente, la sortie pourrait se faire sur une petite fraction de tour. Si la vitesse de broche est augmentée après qu'un certain nombre de passes ne soient déjà faites, la sortie va prendre une plus grande fraction de tour, il en résultera un usinage très brutal pendant ce nouveau mouvement de sortie. Ceci peut être évité en prévoyant une gorge de sortie, ou en ne changeant pas la vitesse de broche pendant le filetage.

Un exemple de programme, g76.ngc montre l'utilisation d'un cycle de filetage G76, il peut être visualisé et exécuté sur n'importe quelle machine utilisant la configuration sim/lathe.ini.

FIG. 12.1 – G76 cycle de filetage



12.18 G81 à G89 : Cycles préprogrammés

Les cycles préprogrammés G81 à G89 sont implémentés comme décrit dans cette section. Deux exemples sont donnés avec la description de G81.

Tous les cycles préprogrammés sont effectués dans le respect du plan de travail courant. N'importe lequel des trois plans de travail XY, YZ, ZX peut être choisi. Dans cette section, la plupart des

descriptions supposeront que le plan de travail XY est le plan courant. Le comportement reste analogue pour les plans de travail YZ ou XZ.

Les mots d'axes rotatifs sont autorisés dans les cycles préprogrammés, mais il est préférable de les omettre. Si les mots d'axes rotatifs sont utilisés, leurs valeurs doivent rester les mêmes que celles de la position courante, de sorte qu'ils ne tournent pas.

Tous les cycles préprogrammés utilisent X-, Y-, R-, et Z- dans le code NC. Ces valeurs sont utilisées pour déterminer les positions de X, Y, R, et Z. La position de R- (signifiant rétraction) est perpendiculaire au plan de travail courant (axe Z pour le plan XY, axe X pour le plan YZ, axe Y pour le plan XZ). Quelques cycles préprogrammés utilisent des arguments supplémentaires.

Dans les cycles préprogrammés, un nombre est appelé "sticky" (collant) si, quand le même cycle est répété sur plusieurs lignes de code en colonne, le nombre doit être indiqué la première fois, mais qu'il devient optionnel pour le reste des lignes suivantes. Les nombres "sticky" conservent leur valeur tant qu'ils ne sont pas explicitement programmés avec une nouvelle valeur. La valeur de R- est toujours "sticky".

En mode de déplacements incrémentaux (G91) : quand le plan courant est XY, les valeurs X-, Y-, est R- sont traitées comme incrémentales à partir de la position courante et Z- comme un incrément depuis la position précédent le mouvement impliquant l'axe Z. Quand le plan YZ ou XZ est le plan courant, le traitement des mots d'axes est analogue. En mode de déplacements absolus, les valeurs de X-, Y-, R-, et Z- sont des positions absolues dans le système de coordonnées courant.

La valeur L- est optionnelle, elle représente le nombre de répétitions. $L = 0$ n'est pas permis. Si les répétitions sont utilisées, elles le sont normalement en mode de déplacements incrémentaux, de sorte que la même séquence de mouvements puisse être répétée à plusieurs endroits, égalements espacés, le long d'un ligne droite. En mode de déplacements absolus, $L > 1$ signifie "faire le même cycle au même endroit, plusieurs fois". L'omission du mot L revient à spécifier $L = 1$. La valeur de L- n'est pas "sticky".

Avec $L > 1$ en mode incrémental et XY comme plan courant, les positions X et Y sont déterminées en ajoutant les valeurs X- et Y- de la commande à celles de la position courante, pour le premier trajet ou, ensuite, à celles de la position finale du précédent trajet, pour les répétitions. Les valeurs de R- et de Z- ne changent pas durant toutes les répétitions.

La hauteur du mouvement de retrait à la fin de chaque répétition (appellée "plan de retrait" dans les descriptions suivantes) est déterminée par le passage en mode : retrait sur la position initiale de Z, si elle est au dessus de la valeur de R- et que le mode de retrait est `G98`, OLD_Z, sinon, à la position de R-. Voir la section 12.22

C'est une erreur si :

- Tous les mots X, Y et Z sont manquants durant un cycle préprogrammé.
- Un nombre P est requis mais un nombre P négatif est utilisé.
- Un nombre L est utilisé mais n'est pas un entier positif.
- Un mouvement d'axe rotatif est utilisé durant un cycle préprogrammé.
- Une vitesse inverse du temps est activée durant un cycle préprogrammé.
- La correction de rayon d'outil est activée durant un cycle préprogrammé.

Quand le plan XY est actif, la valeur de Z est "sticky", et c'est une erreur si :

- La valeur de Z est manquante alors qu'un même cycle préprogrammé n'a pas encore été activé.
- La valeur de R est inférieure à celle de Z.

Quand le plan XZ est actif, la valeur de Y est "sticky", et c'est une erreur si :

- La valeur de Y est manquante alors qu'un même cycle préprogrammé n'a pas encore été activé.
- La valeur de R est inférieure à celle de Y.

Quand le plan YZ est actif, la valeur de X est "sticky", et c'est une erreur si :

- La valeur de X est manquante alors qu'un même cycle préprogrammé n'a pas encore été activé.
- La valeur de R est inférieure à celle de X.

12.18.1 Mouvement préliminaire et Intermédiaire

Tout au début de l'exécution d'un cycle préprogrammé, avec le plan courant XY, si la position actuelle de Z est en dessous de la position de retrait R, l'axe Z va à la position R. Ceci n'arrive qu'une fois, sans tenir compte de la valeur de L.

En plus, au début du premier cycle et à chaque répétition, un ou deux des mouvements suivants sont faits :

1. Un déplacement en ligne droite, parallèle au plan XY, vers le position programmée.
2. Un déplacement en ligne droite, de l'axe Z seul vers la position de retrait R, si il n'est pas déjà à cette position R.

Si un des plans XZ ou YZ est actif, le mouvement préliminaire et intermédiaire est analogue.

12.18.2 G81 : Cycle de perçage

Le cycle `G81` est destiné au perçage. Programmer `G81 X- Y- Z- A- B- C- R- L-` donnera :

1. Un mouvement préliminaire, comme il a été traité ci-dessus.
2. Un déplacement de l'axe Z seul à la vitesse programmée, vers la position Z programmée.
3. Retrait de l'axe Z en vitesse rapide jusqu'au plan de retrait R.

Exemple 1. Supposons que la position courante soit, X1, Y2, Z3 dans le plan XY, la ligne de codes suivante est interprétée :

```
G90 G81 G98 X4 Y5 Z1.5 R2.8
```

Le mode de déplacements absolus est appelé (`G90`), le plan de retrait est positionné sur OLD_Z (`G98`), l'appel du cycle de perçage `G81` va lancer ce cycle une fois. La position X deviendra celle demandée, X4. La position de Y deviendra celle demandée, Y5. La position de Z deviendra celle demandée, Z1.5. La valeur de R fixe le plan de retrait de Z à 2.8. La valeur de OLD_Z est 3. Les mouvements suivants vont se produire.

1. Un mouvement en vitesse rapide, parallèle au plan XY vers X4, Y5, Z3
2. Un mouvement en vitesse rapide, parallèle à l'axe Z vers X4, Y5, Z2.8
3. Un mouvement en vitesse travail, parallèle à l'axe Z vers X4, Y5, Z1.5
4. Un mouvement en vitesse rapide, parallèle à l'axe Z vers X4, Y5, Z3

Exemple 2. Supposons que la position courante soit, X1, Y2, Z3 dans le plan XY, la ligne de codes suivante est interprétée :

```
G91 G81 G98 X4 Y5 Z-0.6 R1.8 L3
```

Le mode de déplacements incrémentaux est appelé (`G91`), le plan de retrait est positionné sur OLD_Z (`G98`), l'appel du cycle de perçage `G81` demande 3 répétitions du cycle. La valeur demandée de X est 4, la valeur demandée de Y est 5, la valeur demandée de Z est -0.6 et le retrait R est à 1.8. La position initiale de X sera 5 (1+4), la position initiale de Y sera 7 (2+5), le plan de retrait sera positionné sur 4.8 (1.8+3) et Z positionné sur 4.2 (4.8-0.6). OLD_Z est à 3.

Le premier mouvement en vitesse rapide le long de l'axe Z vers X1, Y2, Z4.8), puisque OLD_Z est inférieur au plan de retrait.

La première répétition produira 3 mouvements.

1. Un déplacement en vitesse rapide, parallèle au plan XY vers X5, Y7, Z4.8

2. Un déplacement en vitesse travail, parallèle à l'axe Z vers X5, Y7, Z4.2
3. Un déplacement en vitesse rapide, parallèle à l'axe Z vers X5, Y7, Z4.8

La deuxième répétition produira 3 mouvements. La position de X est augmentée de 4 et passe à 9, la position de Y est augmentée de 5 et passe à 12.

1. Un déplacement en vitesse rapide, parallèle au plan XY vers X9, Y12, Z4.8
2. Un déplacement en vitesse travail, parallèle à l'axe Z vers X9, Y12, Z4.2
3. Un déplacement en vitesse rapide, parallèle à l'axe Z vers X9, Y12, Z4.8

La troisième répétition produira 3 mouvements. La position de X est augmentée de 4 et passe à 13, la position de Y est augmentée de 5 et passe à 17.

1. Un déplacement en vitesse rapide, parallèle au plan XY vers X13, Y17, Z4.8
2. Un déplacement en vitesse travail, parallèle à l'axe Z vers X13, Y17, Z4.2
3. Un déplacement en vitesse rapide, parallèle à l'axe Z vers X13, Y17, Z4.8

12.18.3 G82 : Cycle de perçage avec temporisation

Le cycle `G82` est destiné au perçage. Programmer `G82 X- Y- Z- A- B- C- R- L- P-`

donnera :

1. Un mouvement préliminaire, comme il a été traité ci-dessus.
2. Un déplacement de l'axe Z seul en vitesse programmée, vers la position Z programmée.
3. Une temporisation de P secondes.
4. Retrait de l'axe Z en vitesse rapide jusqu'au plan de retrait R.

12.18.4 G83 : Cycle de perçage avec débourrage

Le cycle `G83` est destiné au perçage profond ou au fraisage avec brise-copeaux. Les retraits, au cours de ce cycle, dégagent les copeaux du trou et fragmentent les copeaux longs (qui sont fréquents lors du perçage dans l'aluminum). Ce cycle utilise la valeur Q qui représente un incrément "delta" le long de l'axe Z. Programmer `G83 X- Y- Z- A- B- C- R- L- Q-`

donnera :

1. Un mouvement préliminaire, comme décrit précédemment.
2. Un mouvement de l'axe Z seul, en vitesse travail, sur la position la moins profonde entre, un incrément delta, ou la position de Z programmée.
3. Un mouvement en vitesse rapide au plan de retrait.
4. Une plongée en vitesse rapide dans le même trou, presque jusqu'au fond.
5. Répétition des étapes 2, 3 et 4 jusqu'à ce que la position programmée de Z soit atteinte à l'étape 2.
6. Un mouvement de l'axe Z en vitesse rapide vers le plan de retrait.

C'est une erreur si :

– La valeur de Q est négative ou égale à zéro.

12.18.5 G84 : Cycle de taraudage à droite

Ce code n'est pas encore implémenté dans EMC2. Il est accepté mais son comportement n'est pas défini. Voir G33.1

12.18.6 G85 : Cycle d'alésage, sans temporisation, retrait en vitesse travail

Le cycle G85 est destiné à l'alésage, mais peut être utilisé pour le perçage ou le fraisage. Programmer G85 X- Y- Z- A- B- C- R- L- donnera :

1. Un mouvement préliminaire, comme décrit précédemment.
2. Un déplacement de l'axe Z seul en vitesse travail, vers la position Z programmée.
3. Retrait de l'axe Z en vitesse travail vers le plan de retrait.

12.18.7 G86 : Cycle d'alésage, arrêt de broche, retrait en vitesse rapide

Le cycle G86 est destiné à l'alésage. Ce cycle utilise la valeur P pour une temporisation en secondes. Programmer G86 X- Y- Z- A- B- C- R- L- P-

donnera :

1. Un mouvement préliminaire, comme décrit précédemment.
2. Un déplacement de l'axe Z seul en vitesse travail, vers la position Z programmée.
3. Une temporisation de P secondes.
4. L'arrêt de rotation de la broche.
5. Retrait de l'axe Z en vitesse rapide vers le plan de retrait.
6. Reprise de la rotation de la broche dans la même direction que précédemment.

La broche doit tourner avant le lancement de ce cycle. C'est une erreur si :

– La broche ne tourne pas avant que ce cycle ne soit éxécuté.

12.18.8 G87 : Back Boring

Ce code n'est pas encore implémenté dans EMC2. Il est accepté mais son comportement n'est pas défini.

12.18.9 G88 : Alésage, arrêt de broche, retrait en manuel

Ce code n'est pas encore implémenté dans EMC2. Il est accepté mais son comportement n'est pas défini.

12.18.10 G89 : Cycle d'alésage, temporisation, retrait en vitesse travail

Le cycle G89 est destiné à l'alésage. Il utilise la valeur de P pour une temporisation en secondes. Programmer G89 X- Y- Z- A- B- C- R- L- P-

donnera :

1. Un mouvement préliminaire, comme décrit précédemment.
2. Un déplacement de l'axe Z seul en vitesse travail, vers la position Z programmée.
3. Temporisation de P secondes.
4. Retrait de l'axe Z en vitesse travail vers le plan de retrait.

12.18.11 G90, G91 : Modes de déplacement

L'interprétation du code RS274/NGC peut se faire dans deux modes de déplacements : absolu ou incrémental.

Pour se déplacer en mode absolu, programmer `G90`. En mode absolu, les valeurs d'axes X, Y, Z, A, B, C représentent les positions dans le système de coordonnées courant. Les exceptions à cette règle sont décrites dans la section 12.18.

Pour se déplacer en mode incrémental, programmer `G91`. En mode incrémental, les valeurs d'axes X, Y, Z, A, B, C représentent une valeur à ajouter à la position courante.

Les valeurs I et J représentent toujours des incréments, quel que soit le mode de déplacement actif. La valeur de K représente un incrément à ajouter dans tous les cas sauf un (voir la section 12.18.8), dans lequel la signification du mode de déplacement change.

12.19 G92, G92.1, G92.2, G92.3 : Décalage d'origine du système de coordonnées

Voir la section 10.6 pour une vue générale des systèmes de coordonnées.

Pour donner au point actuel de nouvelles valeurs de coordonnées (sans faire de mouvement), programmer `G92 X- Y- Z- A- B- C-` , où les mots d'axes contiennent les valeurs souhaitées. Au moins un mot d'axe est obligatoire, les autres sont optionnels. Si il n'y a pas de mot d'axe pour un axe donné, les coordonnées de cet axe resteront inchangées. C'est une erreur si :

1. Tous les mots d'axes sont omis.

Quand `G92` est exécuté, l'origine du système de coordonnées courant est déplacée. Pour ce faire, les décalages d'origine sont calculés de sorte que les coordonnées du point courant concordent avec l'origine déplacée comme spécifié dans la ligne du G92. En plus, les paramètres 5211 à 5216 sont positionnés sur le décalage des axes X, Y, Z, A, B et C. Le décalage d'un axe correspond à la distance dont l'origine doit être déplacée afin que la coordonnée du point contrôlé, sur cet axe, ait la valeur spécifiée.

Voici un exemple : Supposons que le point courant soit X=4 dans le système de coordonnées courant et que le décalage actuel de l'axe X soit à zéro, alors `G92 x7` passe le décalage de l'axe X à -3, positionne le paramètre 5211 à -3 et positionne la coordonnée en X du point courant à 7.

Les décalages d'axes sont toujours utilisés quand un mouvement est spécifié en mode de déplacement absolu, en utilisant un des neuf systèmes de coordonnées (ceux désignés par `G54 - G59.3`). Ainsi, les neuf systèmes de coordonnées sont affectés par `G92`.

Le fait d'être en mode de déplacement incrémental est sans effet sur l'action de `G92`.

Des décalages non nuls peuvent déjà être actifs quand `G92` est appelé. Si c'est le cas, la nouvelle valeur de chaque décalage devient A+B, où A est le décalage voulu si l'ancien avait été nul, et B est l'ancien décalage. Par exemple, après le précédent exemple, la valeur de X au point courant est 7. Si `G92 x9` est alors programmé, le nouveau décalage de l'axe X devient -5, qui est calculé par `[[7-9] + -3]`.

Pour repasser un décalage d'axe à zéro, programmer `G92.1` ou `G92.2`. `G92.1` positionne les paramètres 5211 à 5216 à zéro, tandis que `G92.2` conserve leurs valeurs courantes inchangées.

Pour positionner des valeurs de décalage d'axes aux valeurs données dans les paramètres 5211 à 5216, programmer `G92.3`.

Vous pouvez positionner les décalages d'axes dans un programme et ré-utiliser les mêmes dans un autre programme. Programmer `G92` dans le premier programme, ce qui positionnera les paramètres 5211 à 5216. Ne pas utiliser G92.1 dans la suite du premier programme. Les valeurs des paramètres

seront enregistrées lors de la sortie du premier programme et rétablies au chargement du second programme. Utiliser `G92.3` vers le début du deuxième programme, ce qui restaurera les décalages d'axes enregistrés dans le premier. Si d'autres programmes sont lancés entre le programme qui enregistre les décalages et celui qui les restaure, faire une copie du fichier de paramètres écrit par le premier programme et l'utiliser comme fichier de paramètres pour le deuxième programme.

12.20 G93, G94, G95 : Choix des modes de vitesse

Trois modes de vitesse sont reconnus : unités par minute, inverse du temps et unités par tour. Programmer G94 pour passer en mode unités par minute. Programmer G93 pour passer en mode inverse du temps. Programmer G95 pour passer en mode unités par tour.

Dans le mode vitesse en unités par minute, le mot F est interprété pour indiquer que le point contrôlé doit se déplacer à un certain nombre de pouces par minute, de millimètres par minute, ou de degrés par minute, selon l'unité de longueur choisie pour les axes et quels types d'axes doivent se déplacer.

Dans le mode vitesse en unités par tour, le mot F est interprété pour indiquer que le point contrôlé doit se déplacer à un certain nombre de pouces par tour de broche, de millimètres par tour, selon l'unité de longueur utilisée et quels axes doivent être déplacés.

Dans le mode vitesse inverse du temps, le mot F signifie que le mouvement doit être terminé en [1 divisé par la valeur de F] minutes. Par exemple, si la valeur de F est 2.0, les mouvements doivent être terminés en 1/2 minute.

Quand le mode vitesse inverse du temps est actif, le mot F doit apparaître sur chaque ligne contenant un mouvement G1, G2, ou G3. Les mots F qui sont sur des lignes sans G1, G2, ou G3 sont ignorés. Être en mode vitesse inverse du temps est sans effet sur les mouvements G0 (vitesse rapide). C'est une erreur si :

- Le mode vitesse inverse du temps est actif et qu'une ligne avec G1, G2, ou G3 (explicitement ou implicitement) n'a pas de mot F.
- Une nouvelle vitesse n'a pas été spécifiée après un passage en G94 ou G95.

12.21 G96, G97 : Modes de contrôle de la broche

Deux modes de contrôle de la broche sont reconnus : tours par minute, et vitesse de coupe constante. Programmer G96 D- S- pour valider une vitesse de coupe constante de S pieds par minute si G20 est actif, ou mètres par minute si G21 est actif. La vitesse de rotation maximale est indiquée par la valeur de D- en tours par minute.

Programmer G97 pour activer le mode vitesse en tours par minute.

C'est une erreur si :

- S n'est pas spécifié avec G96.
- Une vitesse est spécifiée en mode G96 et la broche ne tourne pas.

12.22 G98, G99 : Options du plan de retrait

Quand la broche se rétracte pendant les cycles préprogrammés, il existe deux options pour indiquer comment elle se rétracte : (1) Retrait perpendiculaire au plan de travail courant jusqu'à la position indiquée par le mot R, ou (2) Retrait perpendiculaire au plan de travail courant jusqu'à la position qui était celle de cet axe juste avant le début du cycle préprogrammé (à moins que cette position ne soit inférieure à celle indiquée par le mot R, auquel cas, c'est cette dernière qui serait utilisée.

Pour utiliser l'option (1), programmer `G99`. Pour utiliser l'option (2), programmer `G98`. Ne pas oublier que le mot R a différentes significations en mode de déplacement absolu et en mode de déplacement incrémental.

Chapitre 13

M Codes

13.1 M0, M1, M2, M30, M60 : Arrêts de programme

Pour stopper temporairement un programme en cours (quelle que soit la position du bouton d'arrêt facultatif), programmer M0.

Pour stopper temporairement un programme en cours (mais seulement si le bouton d'arrêt optionnel est activé), programmer M1.

Il est permis de programmer `M0` et `M1` en mode données manuelles (MDI), mais l'effet ne sera probablement pas perceptible, puisque le comportement normal en mode MDI est de s'arrêter, de toute façon, à la fin de chaque ligne.

Pour procéder à l'échange de porte-pièce avec le chargeur de pièces et stopper temporairement un programme en cours (quel que soit le réglage du bouton d'arrêt facultatif), programmer M60.

Si un programme est stoppé par M0, M1, ou M60, en appuyant sur le bouton de départ cycle, le programme reprend à la ligne suivante.

Pour finir un programme, programmer `M2`. Pour changer le porte-pièce du chargeur et finir un programme, programmer `M30`. Ces deux commandes produisent les effets suivants :

1. Les décalages d'axes sont mis à zéro (comme avec `G92.2`) et les décalages d'origine sont mis aux valeurs par défaut (comme avec `G54`).
2. Le plan de travail actif devient XY (comme avec `G17`).
3. Le mode de déplacement devient absolu (comme avec `G90`).
4. La vitesse travail passe en unités par minute (comme avec `G94`).
5. Les correcteurs de vitesse sont activés (comme avec `M48`).
6. Les compensations d'outil sont désactivées (comme avec `G40`).
7. La broche est arrêtée (comme avec `M5`).
8. Le mode mouvement courant devient G1 (comme avec `G1`).
9. L'arrosage est arrêté (comme avec `M9`).

Plus aucune ligne de code RS274/NGC ne sera exécutée après exécution de la commande M2 ou M30. Presser le départ cycle relance le programme au début du fichier.

13.2 M3, M4, M5 : Contrôle de broche

Pour démarrer la rotation de la broche en sens horaire à la vitesse programmée courante, programmer `M3`.

Pour démarrer la rotation de la broche en sens anti-horaire à la vitesse programmée courante, programmer M4.

Pour arrêter la rotation de la broche, programmer `M5`.

Il est permis d'utiliser `M3` ou `M4` si la vitesse de broche est à zéro. Si cela est fait (ou si le bouton de correction de vitesse est activé mais mis à zéro), la broche ne tournera pas, si, plus tard la vitesse de broche est augmentée (ou que le correcteur de vitesse est augmenté), la broche va se mettre en rotation. Il est permis d'utiliser `M3` ou `M4` quand la broche est déjà en rotation ou d'utiliser `M5` quand la broche est déjà arrêtée.

13.3 M6 : Appel d'outil

Pour changer l'outil actuellement dans la broche par un autre, nouvellement sélectionné en utilisant le mot T, voir la section 15.3), programmer `M6`. Un changement d'outil complet donnera :

- La rotation de la broche est arrêtée.
- L'outil qui a été sélectionné (par le mot T sur la même ligne ou sur n'importe quelle ligne après le changement d'outil précédent), sera placé dans la broche. Le mot T est un nombre entier indiquant le numéro du slot d'outil dans le carrousel (non son index).
- Si l'outil sélectionné n'est pas déjà dans la broche avant le changement d'outil, l'outil qui était dans la broche (s'il y en avait un) va être replacé dans son emplacement dans le chargeur.
- Les coordonnées des axes seront arrêtées dans les mêmes positions absolues qu'elles avaient avant le changement d'outil (mais la broche devra peut-être être réorientée).
- Aucune autre modification ne sera apportée. Par exemple, l'arrosage continue à couler durant le changement d'outil à moins qu'il ne soit arrêté par `M9`.

Le changement d'outil peut inclure des mouvements d'axes pendant son exécution. Il est permis, mais pas utile, de programmer un changement d'outil avec le même outil que celui qui est déjà dans la broche. Il est permis également, si il n'y a pas d'outil dans le slot sélectionné, dans ce cas, la broche sera vide après le changement d'outil. Si le slot zéro a été le dernier sélectionné, il n'y aura pas d'outil dans la broche après le changement.

13.4 M7, M8, M9 : Contrôle d'arrosage

Pour activer l'arrosage par brouillard (gouttelettes), programmer `M7`.

Pour activer l'arrosage fluide, programmer `M8`.

Pour arrêter tous les arrosages, programmer `M9`.

Il est toujours permis d'utiliser une de ces commandes, que les arrosages soient arrêtés ou non.

13.5 M48, M49 : Contrôle des correcteurs de vitesse

Pour autoriser les potentiomètres de corrections de vitesses de broche et celui de vitesse travail, programmer `M48`. Pour les inhiber tous les deux, programmer `M49`. Voir la section 10.3.1 pour plus de détails. Il est permis d'autoriser ou d'inhiber ces potentiomètres quand ils sont déjà autorisés ou inhibés. Ces potentiomètres peuvent aussi être activés individuellement en utilisant les commandes M50 et M51 comme décrit dans les sections 13.6 et 13.7.

13.6 M50 : Contrôle du correcteur de vitesse travail

Pour autoriser le potentiomètre de correction de vitesse travail, programmer `M50` ou `M50 P1`. Pour inhiber ce potentiomètre, programmer `M50 P0`. Quand il est inhibé, le potentiomètre de correction de vitesse n'a plus aucune influence et les mouvements seront exécutés à la vitesse de travail programmée. (à moins que ne soit actif un correcteur de vitesse adaptative).

13.7 M51 : Contrôle du correcteur de vitesse broche

Pour autoriser le potentiomètre de correction de vitesse de la broche, programmer `M51` ou `M51 P1`. Pour inhiber ce potentiomètre programmer `M51 P0`. Quand il est inhibé, le potentiomètre de correction de vitesse de broche n'a plus aucune influence et la broche tournera à la vitesse programmée, en utilisant le mot S comme décrit dans la section 15.2.

13.8 M52 : Contrôle de vitesse adaptative

Pour utiliser une vitesse adaptative, programmer `M52` ou `M52 P1`. Pour stopper l'utilisation d'une vitesse adaptative, programmer `M52 P0`. Quand la vitesse adaptative est utilisée, certaines valeurs externes sont utilisées avec les correcteurs de vitesse de l'interface utilisateur et les vitesses programmées pour obtenir la vitesse travail. Dans EMC2, la HAL pin `motion.adaptive-feed` est utilisée dans ce but. Les valeurs de `motion.adaptive-feed` doivent être dans une échelle comprise entre 0 (pas de vitesse) et 1 (pleine vitesse).

13.9 M53 : Contrôle de coupure de vitesse

Pour autoriser le bouton de coupure de vitesse, programmer `M53` ou `M53 P1`. Pour inhiber ce bouton programmer `M53 P0`. Autoriser la coupure de vitesse permet d'interrompre les mouvements par le biais d'une coupure de vitesse. Dans EMC2, la HAL pin `motion.feed-hold` est utilisée pour cette fonctionnalité. Une valeur de 1 provoque un arrêt des mouvements (si `M53` est actif).

13.10 M62 à M65 : Contrôle de sortie digitale

Pour contrôler un bit de sortie digitale, programmer `M- P-`, où le mot M doit être compris entre 62 et 65, et le mot P compris entre 0 et un maximum défini selon l'implémentation.

M62 Activer la sortie digitale synchronisée avec un mouvement.

M63 Désactiver la sortie digitale synchronisée avec un mouvement.

M64 Activer immédiatement la sortie digitale.

M65 Désactiver immédiatement la sortie digitale.

13.11 M66 : Contrôle d'entrée digitale

Pour contrôler un bit d'entrée digitale, programmer `M66 P- E- L- Q-` , où le mot P et le mot E doivent être compris entre 0 et un maximum défini selon l'implémentation. Un seul des mots P ou E doit être présent. C'est une erreur si ils sont présents tous les deux.

M66 Attente d'une entrée

– Le mot P spécifie le numéro d'une entrée digitale.
– Le mot E spécifie le numéro d'une entrée analogique.
– Le mot L spécifie le type d'attente :

0 - WAIT_MODE_IMMEDIATE - pas d'attente, retour immédiat. La valeur courante de l'entrée est stockée dans le paramètre #5399

1 - WAIT_MODE_RISE - attente d'un front montant sur l'entrée.

2 - WAIT_MODE_FALL - attente d'un front descendant sur l'entrée.

3 - WAIT_MODE_HIGH - attente d'un état logique HIGH sur l'entrée.

4 - WAIT_MODE_LOW - attente d'un état logique LOW sur l'entrée.

– Le mot Q spécifie le timeout pour l'attente. Si le timeout est dépassé, l'attente est interrompue et la variable #5399 positionnée à -1.

M66 attends un nouvel événement sur l'entrée ou l'arrêt de l'exécution du programme, jusqu'à ce que l'événement sélectionné (ou le timeout programmé) ne survienne. C'est une erreur de programmer une valeur de timeout à 0 dans tous les types, sauf le type 0.

C'est également une erreur de programmer M66 avec les deux mots, un mot P et un mot E (ce qui reviendrait à sélectionner à la fois une entrée analogique et une digitale).

13.12 M100 à M199 : Commandes définies par l'utilisateur

Pour invoquer une commande définie par l'utilisateur, programmer M- P- Q- où P et Q sont facultatifs. Le programme externe " Mnnn" dans le répertoire [DISPLAY] PROGRAM_PREFIX est exécuté avec les valeurs P et Q comme étant ses deux arguments. L'exécution du fichier RS274NGC passe en pause jusqu'à ce que le programme invoqué ne se termine.

C'est une erreur si :

– La commande spécifiée n'existe pas.

Chapitre 14

O Codes

Les O-codes permettent le contrôle de flux dans les programmes NC. Chaque block est associé à une adresse, qui est la valeur utilisée après le O. Il faut prendre soin de bien faire correspondre les adresses des codes O.

Le comportement est indéfini si :

– D'autres mots sont utilisés sur une ligne contenant un mot O.
– Un commentaire est utilisé sur une ligne contenant un code O.

14.1 Sous-programmes : "sub", "endsub", "return", "call"

Les sous-programmes s'étendent d'un **`O- sub`** à un **`O- endsub`**. Les lignes, à l'intérieur du sous-programme (le corps du sous-programme), ne sont pas exécutées dans l'ordre, mais elles sont exécutées à chaque fois que le sous-programme est appelé avec un **`O- call`**.

```
O100 sub (sous-programme pour aller à l'origine programme)
G0 X0 Y0 Z0
O100 endsub
(plusieurs lignes)
O100 call
```

À l'intérieur d'un sous-programme, **`O- return`** peut être exécuté, pour retourner immédiatement au code appelant, comme si **`O- endsub`** avait été rencontré.

`O- call` peut prendre jusqu'à 30 arguments optionnels, qui sont passés au sous-programme comme `#1`, `#2`, ..., #N. Les paramètres de #N+1 à #30 ont la même valeur dans le contexte de l'appel. Au retour du sous-programme, les valeurs des paramètres #1 jusqu'à #30 (quel que soit le nombre d'arguments) sont restaurés aux valeurs qu'ils avaient avant l'appel.

Parce que " 1 2 3" est analysé comme le nombre 123, les paramètres doivent être placés entre crochets. L'appel de sous-programme suivant, s'effectue avec 3 arguments :

```
O200 call [1] [2] [3]
```

Les corps de sous-programme ne peuvent pas être imbriqués. Ils ne peuvent être appelés qu'après avoit été définis. Ils peuvent être appelés depuis d'autres fonctions et peuvent s'appeler eux même récursivement, s'il est judicieux de le faire. Le niveau maximum d'imbrication des sous-programmes est de 10.

Les sous-programmes n'ont pas de "valeur de retour", mais ils peuvent changer la valeur des paramètres au dessus de #30 et ces changements sont visibles depuis le code appelant. Les sous-programmes peuvent aussi changer la valeur des paramètres nommés globaux.

14.2 Boucles : “do”, “while”, “endwhile”, “break”, “continue”

La boucle **while** a deux structures possibles : while/endwhile et do/while. Dans chaque cas, la boucle est quittée quand la condition du “while” devient fausse.

```
(dessine la forme d'une dent de scie)
F100
#1 = 0
O101 while [#1 lt 10]
G1 X0
G1 Y[#1/10] X1
#1 = [#1+1]
O101 endwhile
```

À l'intérieur d'une boucle while, **O- break,** quitte immédiatement la boucle et **O- continue,** saute immédiatement à la prochaine évaluation de la condition du `while`. Si elle est vraie, la boucle recommence au début. Si elle est fausse, la boucle est quittée.

14.3 Conditionnel : “if”, “else”, “endif”

Le **if** conditionnel exécute un groupe d'instructions si sa condition est vraie et un autre groupe si elle est fausse.

```
(Ajuste la vitesse travail en fonction d'une variable)
O102 if [#2 GT 5]
F100
O102 else
F200
O102 endif
```

14.4 Indirection

L'adresse de O- peut être donnée par un paramètre ou un calcul.

```
O[#101+2] call
```

14.5 Calcul des valeurs dans les mots O

Dans les mots O-, les paramètres (section 11.3.2), les expressions (section 11.3.4), les opérateurs binaires (section 11.3.5) et les fonctions (tableau 11.3), sont particulièrement intéressants.

Chapitre 15

Autres Codes

15.1 F : Réglage de la vitesse travail

Pour régler la vitesse d'avance, programmer `F-` . L'application de la vitesse est telle que décrite dans la section 10.2.5, à moins que le mode vitesse inverse du temps ne soit actif, dans ce cas, la vitesse est telle que décrite dans la section 12.20.

15.2 S : Réglage de la vitesse de rotation de la broche

Pour régler la vitesse en tours par minute (tr/mn) de la broche, programmer `S-` . La broche va tourner à cette vitesse quand elle sera programmée pour tourner. Il est permis de programmer un mot S que la broche tourne ou non. Si le potentiomètre de correction de vitesse broche est autorisé et n'est pas positionné sur 100%, la vitesse de broche sera différente de celle programmée. Il est permis de programmer S0, la broche ne tournera pas. C'est une erreur si :

– La valeur de S est négative.

Comme décrit dans la section 12.18.5, si un `cycle préprogrammé G84` (taraudage) est actif et que les potentiomètres de vitesse et d'avance sont autorisés, celui qui a le réglage le plus bas sera utilisé. La vitesse de rotation et d'avance resterons synchronisées. Dans ce cas, la vitesse peut différer de celle programmée, même si le potentiomètre de correction de vitesse travail est sur 100%.

15.3 T : Choix de l'outil

Pour sélectionner un outil, programmer `T-`, où la valeur de T correspond au numéro du slot d'outil dans le carrousel. L'outil ne sera appelé et changé que quand un `M6` sera programmé (voir la section 13.3). Le mot T peut apparaitre sur la même ligne que le `M6` ou sur une ligne précédente. Il est permis, mais normalement inutile, qu'un mot T apparaisse à plus de deux lignes avant, sans changement d'outil. Le carrousel peut bouger, seulement le plus récent mot T ne prendra effet qu'au prochain changement d'outil. Il est permis de programmer `T0`, aucun outil ne sera sélectionné. C'est utile pour avoir la broche vide. C'est une erreur si :

– Un valeur négative de T est utilisée.
– Une valeur de T supérieure au nombre de slot d'outils dans le carrousel est utilisée.

Sur certaines machines, le carrousel se déplace lorsque le mot T est programmé, avec l'usinage en cours. Sur ces machines, programmer T plusieurs lignes de texte avant le changement d'outil permet de gagner du temps. Une pratique de programmation courante pour ces types de machines, consiste à placer le mot T pour le prochain outil sur la ligne suivant le changement d'outil. Cela laisse au carrousel tout le temps pour se positionner.

Chapitre 16

Ordre d'exécution

L'ordre d'exécution des éléments d'une ligne est essentielle à la sécurité et l'efficacité d'une machine. Les éléments sont exécutés dans l'ordre indiqué ci-dessous, si ils se trouvent sur la même ligne.

1. Commentaire (message inclu)
2. Positionnement du mode de vitesses (G93, G94).
3. Réglage de la vitesse travail (F).
4. Réglage de la vitesse de rotation de la broche (S).
5. Sélection de l'outil (T).
6. Appel d'outil (M6).
7. Marche/Arrêt broche (M3, M4, M5).
8. Marche/Arrêt arrosages (M7, M8, M9).
9. Activation/Inhibition correcteurs de vitesse (M48, M49).
10. Temporisation (G4).
11. Choix du plan de travail (G17, G18, G19).
12. Choix de l'unité de longueur (G20, G21).
13. Activation/Désactivation de la compensation de rayon d'outil (G40, G41, G42)
14. Activation/Désactivation de la compensation de longueur d'outil (G43, G49)
15. Sélection du système de coordonnées (G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3).
16. Réglage du mode de trajectoire (G61, G61.1, G64)
17. Réglage du mode de déplacement (G90, G91).
18. Réglage du type de retrait (G98, G99).
19. Prise d'origine machine (G28, G30) ou établissement du système de coordonnées (G10) ou encore, réglage des décalages d'axes (G92, G92.1, G92.2, G94).
20. Effectuer les mouvements (G0 à G3, G33, G80 à G89), tels que modifiés (éventuellement) par G53.
21. Arrêt (M0, M1, M2, M30, M60).

Chapitre 17

G-Code : Bonnes pratiques

17.1 Utiliser un nombre de décimales approprié

Utilisez au plus 3 chiffres après la virgule pour l'usinage en millimètres et au plus 4 chiffres après la virgule pour l'usinage en pouces. En particulier, les contrôles de tolérance des arcs sont faits pour .001 et .0001 selon les unités actives.

17.2 Utiliser les espaces de façon cohérente

Le G-code est plus lisible quand au moins un espace apparaît avant les mots. S'il est permis d' insérer des espaces blancs au milieu des chiffres, il faut éviter de le faire.

17.3 Préférer le "format centre" pour les arcs

Les arcs en format centre (qui utilisent `I- J- K-` au lieu de `R-`) se comportent de façon plus précise que ceux en format rayon, particulièrement pour des angles proche de 180 et 360 degrés.

17.4 Placer les codes modaux importants au début des programmes

Lorsque l'exécution correcte de votre programme dépend de paramètres modaux, n'oubliez pas de les mettre au début du programme. Des modes incorrects peuvent provenir d'un programme précédent ou depuis des entrées manuelles.

Une bonne mesure préventive consiste à placer la ligne suivante au début de tous les programmes :

```
G17 G21 G40 G49 G54 G80 G90 G94
```

(plan XY, mode mm, annulation de la compensation de rayon, et de longueur, sytème de coordonnées numéro 1, arrêt des mouvements, déplacements absolus, mode vitesse/minute)

Peut-être que le code modal le plus important est le réglage des unités machine. Si les codes G20 ou G21, ne sont pas inclus, selon les machines l'échelle d'usinage sera différente. D'autres valeurs comme le plan de retrait des cycles préprogrammés peuvent être importantes.

17.5 Ne pas mettre trop de choses sur une ligne

Ignorer le contenu de la section 16 et ne pas écrire de ligne de code qui laisse la moindre ambiguïté. De même, ne pas régler un paramètre et l'utiliser sur la même ligne, même si la sémantique est bien définie. (Exception : Mise à jour d'une variable à une nouvelle valeur, comme `#1=[#1+#2]`)

17.6 Ne pas utiliser les numéros de ligne

Les numéros de ligne n'apportent rien. Quand des numéros de ligne sont rapportés dans les messages d'erreur, ces numéros font référence aux numéros de lignes à l'intérieur du programme, pas aux valeurs des mots N.

17.7 Lorsque plusieurs systèmes de coordonnées sont déplacés, envisager le mode vitesse inverse du temps

Parce que la signification d'un mot F en mètres par minute varie selon les axes à déplacer et parce que la quantité de matière enlevée ne dépend pas que de la vitesse travail, il peut être plus simple d'utiliser G93, vitesse inverse du temps pour atteindre l'enlévement de matière souhaîté.

Chapitre 18

Fichier d'outils et compensations

18.1 Fichier d'outils

Les longueurs et diamètres d'outils peuvent être lus dans un fichier d'outils (voir la section 10.4) ou provenir d'un mot spécifié pour activer la compensation d'outil.

18.2 Compensation d'outil

La compensation d'outil peut causer beaucoup de problèmes aux meilleurs programmeurs de code nc. Mais elle peut aussi être une aide puissante quand elle est utilisée pour aider un opérateur à obtenir une pièce à la cote. En réglant la longueur et le diamètre des outils dans une table d'outils, les décalages peuvent être utilisés pendant un cycle d'usinage qui tient compte des variations de taille de l'outil, ou pour des déviations mineures des trajectoires programmées. Et ces changements peuvent être faits sans que l'opérateur n'ait à changer quoi que ce soit dans le programme.

Tout au long de ce module, vous trouverez occasionnellement des références à des fonctions canoniques, là où il est nécessaire pour le lecteur de comprendre comment fonctionne une compensation d'outil dans une situation spécifique. Ces références ont pour but de donner au lecteur une idée de la séquences plutôt que d'exiger qu'il comprenne la façon dont les fonctions canoniques elles-mêmes fonctionnent dans EMC.

18.3 Compensation de longueur d'outil

Les compensations de longueur d'outil sont données comme des nombres positifs dans la table d'outils. Une compensation d'outil est programmée en utilisant G43 Hn, où n est le numéro d'index de l'outil souhaité dans la table d'outil. Il est prévu que toutes les entrées dans la table d'outils soit positives. La valeur de H est vérifiée, elle doit être un entier non négatif quand elle est lue. L'interprèteur se comporte comme suit :

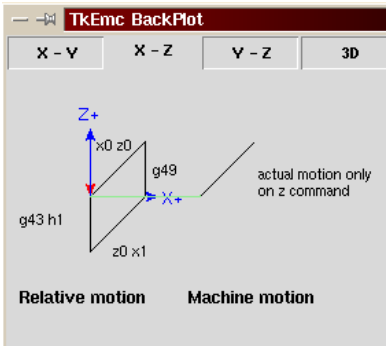
1. Si G43 Hn est programmé, un appel à la fonction USE_TOOL_LENGTH_OFFSET(longueur) est fait (où longueur est l'écart de longueur, lu dans la table d'outils, de l'outil indexé n), tool_length_offset est repositionné dans le modèle de réglages de la machine et la valeur de current_z dans le modèle est ajustée. Noter que n n'a pas besoin d'être le même que le numéro de slot de l'outil actuellement dans la broche.

2. Si G49 est programmé, USE_TOOL_LENGTH_OFFSET(0.0) est appelée, tool_length_offset est remis à 0.0 dans le modèle de règlages de la machine et la valeur courante de current_z dans le modèle est ajustée. L'effet de la compensation de longueur d'outil est illustrée dans la capture ci-dessous. Noter que la longueur de l'outil est soustraite de Z pour que le point contrôlé programmé

corresponde à la pointe de l'outil. Il faut également noter que l'effet de la compensation de longueur est immédiat quand on voit la position de Z comme une coordonnée relative mais il est sans effet sur la position actuelle de la machine jusqu'à ce qu'un mouvement en Z soit programmé.

Programme de test de longueur d'outil.
Tool #1 is 1 inch long.

```
N01 G1 F15 X0 Y0 Z0
N02 G43 H1 Z0 X1
N03 G49 X0 Z0
N04 G0 X2
N05 G1 G43 H1 G4 P10 Z0 X3
N06 G49 X2 Z0
N07 G0 X0
```



Avec ce programme, dans la plupart des cas, la machine va appliquer le décalage sous forme d'une rampe pendant le mouvement en xyz suivant le mot G43.

18.4 Compensation de rayon d'outil

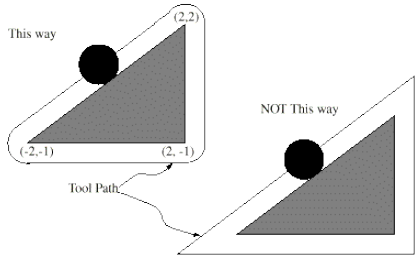
La compensation de rayon d'outil (également appelée compensation de diamètre d'outil) à été ajoutée aux spécifications RS-274D à la demande d'utilisateurs, car elle est extrêmement utile, mais son implémentation a été assez mal pensée. L'objectif de cette fonctionnalité est de permettre aux programmeurs de "virtualiser" la trajectoire de l'outil, de sorte que la machine puisse pendant toute l'exécution, déterminer la bon décalage a apporter à la position de l'outil pour respecter les cotes, en s'appuyant sur les diamètres d'outils existants. Si un outil est réaffuté, son diamètre sera légérement plus petit que celui d'origine, il faudra également en tenir compte.

Le problème est pour donner à la machine la trajectoire exacte où l'outil doit usiner, sur le côté intérieur d'un chemin imaginaire, ou sur le côté extérieur. Comme ces trajectoires ne sont pas nécessairement fermées (même si elles peuvent l'être), il est quasiment impossible à la machine de connaître de quel côté du profil elle doit compenser l'outil. Il a été décidé qu'il n'y aurait que deux choix possibles, outil à "gauche" du profil à usiner et outil à "droite" du profil à usiner. Ce qui doit être interprété à gauche ou à droite du profil à usiner en suivant l'outil le long du profil.

18.4.1 Compensation de rayon d'outil, détails

Les possibilités de compensation de rayon d'outil de l'interpréteur, autorise le programmeur à spécifier si l'outil doit passer à gauche ou à droite du profil à usiner. Ce profil peut être un contour ouvert ou fermé, dans le plan XY composé de segments en arcs de cercles et en lignes droites. Le contour peut être le pourtour d'une pièce brute ou, il peut être une trajectoire exacte suivie par un outil mesuré avec précision. La figure ci-dessous, montre deux exemples de trajectoires d'usinage d'une pièce triangulaire, utilisant la compensation de rayon d'outil.

Dans les deux exemples, le triangle gris représente le matériau restant après usinage et la ligne extérieure représente le chemin suivi par le centre de l'outil. Dans les deux cas le triangle gris est conservé. Le parcour de gauche (avec les coins arrondis) est le chemin généralement interprété. Dans la méthode de droite (celle marquée Not this way), l'outil ne reste pas en contact avec les angles vifs du triangle gris.



Des mouvements sur l'axe Z peuvent avoir lieu pendant que le contour est suivi dans le plan XY. Des portions du contour peuvent être franchies avec l'axe Z en retrait au dessus de la pièce pendant la poursuite du parcour et jusqu'au point où l'usinage doit reprendre, l'axe Z plongera de nouveau en position. Ces dégagement de zones non usinées peuvent être faits en vitesse travail (G1), en rapide (G0), en vitesse inverse du temps (G93) ou en avance en unités par minute (G94) toutes peuvent être utilisées avec la compensation de rayon d'outil. Sous G94, la vitesse sera appliquée à la pointe de l'outil coupant, non au contour programmé.

Instructions de programmation

- Pour activer la compensation de rayon d'outil, programmer soit, G41 (pour maintenir l'outil à gauche du profil à usiner) soit, G42 (pour maintenir l'outil à droite du profil). Dans la figure 18.4.1, par exemple, si G41 était programmé, l'outil devrait tourner en sens horaire autour du triangle et dans le sens contraire si G42 était programmé.
- Pour désactiver la compensation de rayon d'outil, programmer G40.
- Si G40, G41, ou G42 est programmé dans la même ligne qu'un mouvement d'axe, la compensation de rayon sera activée ou désactivée avant que le mouvement ne soit fait. Pour que le mouvement s'effectue en premier, il doit être programmé séparément et avant.

La valeur de D

L'interpréteur actuel requiert une valeur D sur chaque ligne contenant un mot G41 ou G42. La nombre D doit être un entier positif. Il représente le numéro de slot de l'outil, dont le rayon (la moitié du diamètre d'outil indiqué dans la table d'outils) sera compensé. Il peut aussi être égal à zéro, dans ce cas, la valeur du rayon sera aussi égale à zéro. Tous les slots de la table d'outils peuvent être sélectionnés de cette façon. Le nombre D n'est pas nécessairement le même que le numéro de slot de l'outil monté dans la broche.

Table d'outils

La compensation de rayon d'outil utilise les données fournies par la table d'outils de la machine. Pour chaque slot d'outil dans le carrousel, la table d'outils contient le diamètre de l'outil rangé à cet

emplacement (ou la différence entre le diamètre nominal de l'outil placé dans ce slot et son diamètre actuel). La table d'outils est indexée par les numéros de slot. Reportez vous à la page des “Fichiers d'outils” pour savoir comment remplir ces fichiers.

Deux type de contour

L'interpréteur contrôle la compensation pour deux types de contour :

– 1) Le contour donné dans le code NC correspond au bord extérieur du matériau après usinage. Nous l'appellerons "contour sur le profil du matériau".
– 2) Le contour donné dans le code NC correspond à la trajectoire suivie par le centre d'un outil de rayon nominal. Nous l'appellerons "contour sur le chemin d'outil".

L'interpréteur ne dispose d'aucun paramètre pour déterminer quel type de contour est utilisé, mais la description des contours est différente (pour la même géométrie de pièce) entre les deux types, les valeurs des diamètres dans la table d'outils seront également différentes pour les deux types.

Contour sur le profil du matériau

Lorsque le contour est sur le profil du matériau, c'est ce tracé qui est décrit dans le programme NC. Pour un contour sur le profil du matériau, la valeur du diamètre dans la table d'outils correspond au diamètre réel de l'outil courant. Cette valeur dans la table doit être positive. Le code NC pour ce type de contour reste toujours le même à l'excéption du diamètre de l'outil (actuel ou nominal).

Exemple 1 :

Voici un programme NC qui usine le matériau en suivant le profil d'un des triangles de la figure précédente. Dans cet exemple, la compensation de rayon est celle du rayon actuel de l'outil, soit 0.5 . La valeur pour le diamètre dans la table d'outil est de 1.0 .

```
N0010 G41 G1 X2 Y2 (active la compensation et fait le mouvement d'entrée)
N0020 Y-1 (suit la face droite du triangle)
N0030 X-2 (suit la base du triangle)
N0040 X2 Y2 (suit l'hypoténuse du triangle)
N0050 G40 (désactive la compensation)
```

Avec ce programme, l'outil suit cette trajectoire : un mouvement d'entrée, puis la trajectoire montrée dans la partie gauche de la figure, avec un déplacement de l'outil en sens horaire autour du triangle. Noter que les coordonnées du triangle de matériau apparaissent dans le code NC. Noter aussi que la trajectoire inclus trois arcs qui ne sont pas explicitement programmés, ils sont générés automatiquement.

Contour sur le chemin d'outil

Lorsque le contour est sur le chemin d'outil, la trajectoire décrite dans le programme correspond au chemin que devra suivre le centre de l'outil. Le bord de l'outil, à un rayon de là, (excepté durant les mouvements d'entrée) suivra la géométrie de la pièce. La trajectoire peut être créée manuellement ou par un post-processeur, selon la pièce qui doit être réalisée. Pour l'interpréteur, le chemin d'outil doit être tel que le bord de l'outil reste en contact avec la géométrie de la pièce, comme montré à gauche de la figure 7. Si une trajectoire du genre de celle montrée sur la droite de la figure 7 est utilisée, dans laquelle l'outil ne reste pas en permanence au contact avec la géométrie de la pièce, l'interpréteur ne pourra pas compenser correctement si un outil en dessous de son diamètre nominal est utilisé.

Pour un contour sur le chemin d'outil, la valeur pour le diamètre de l'outil dans la table d'outils devra être un petit nombre positif si l'outil sélectionné est légérement surdimensionné. La valeur du

diamètre sera un petit nombre négatif si l'outil est légérement sous-dimensionné. Si un diamètre d'outil est négatif, l'interpréteur compense de l'autre côté du contour programmé et utilise la valeur absolue donnée au diamètre. Si l'outil courant est à son diamètre nominal, la valeur dans la table d'outil doit être à zéro.

Exemple de contour sur le chemin d'outil

Supposons que le diamètre de l'outil courant monté dans la broche est de 0.97 et le diamètre utilisé en générant le chemin d'outil a été de 1.0 . Alors la valeur de diamètre dans la table d'outils pour cet outil est de -0.03. Voici un programme NC qui va usiner l'extérieur d'un triangle de la figure 7.

```
N0010 G1 X1 Y4.5 (mouvement d'alignement)
N0020 G41 G1 Y3.5 (active la compensation et premier mouvement d'entrée)
N0030 G3 X2 Y2.5 I1 (deuxième mouvement d'entrée)
N0040 G2 X2.5 Y2 J-0.5 (usinage le long de l'arc en haut du chemin d'outil)
N0050 G1 Y-1 (usinage le long du côté droit du chemin d'outil)
N0060 G2 X2 Y-1.5 I-0.5 (usinage de l'arc en bas à droite)
N0070 G1 X-2 (usinage de la base du chemin d'outil)
N0080 G2 X-2.3 Y-0.6 J0.5 (usinage de l'arc en bas à gauche)
N0090 G1 X1.7 Y2.4 (usinage de l'hypoténuse)
N0100 G2 X2 Y2.5 I0.3 J-0.4 (usinage de l'arc en haut à droite)
N0110 G40 (désactive la compensation)
```

Avec ce programme, l'outil suit cette trajectoire : un mouvement d'alignement, deux mouvements d'entrée, puis il suit une trajectoire légérement intérieure au chemin d'outil montré sur la figure 7 en tournant en sens horaire autour de la pièce. Cette compensation est à droite de la trajectoire programmée, même si c'est G41 qui est programmé, en raison de la valeur négative du diamètre.

Erreurs de programmation et limitations

Les messages en rapport avec la compensation de rayon d'outil, délivrés par l'interpréteur sont les suivants :

1. Impossible de changer les décalages d'axes avec la compensation de rayon d'outil
2. Impossible de changer d'unité avec la compensation de rayon d'outil
3. Impossible d'activer la compensation de rayon d'outil en dehors du plan XY
4. Action impossible, la compensation de rayon d'outil est déjà active
5. Impossible d'utiliser G28 ou G30 avec la compensation de rayon d'outil
6. Impossible d'utiliser G53 avec la compensation de rayon d'outil
7. Impossible d'utiliser le plan XZ avec la compensation de rayon d'outil
8. Impossible d'utiliser le plan YZ avec la compensation de rayon d'outil
9. Coin concave avec la compensation de rayon d'outil
10. Interférence de l'outil avec une partie finie de la pièce avec la compensation de rayon d'outil[1]
11. Mot D sur une ligne sans mot de commande G41 ni G42
12. Index de rayon d'outil trop grand
13. Le rayon de l'outil n'est pas inférieur au rayon de l'arc avec la compensation de rayon
14. Deux G-codes du même groupe modal sont utilisés.

Pour certains de ces messages, des explications sont données plus loin.

Changer d'outil alors que la compensation de rayon d'outil est active n'est pas considéré comme une erreur, mais il est peu probable que cela soit fait intentionnellement. Le rayon d'outil utilisé

[1] Le terme anglais "gouging" indique une interférence entre l'outil et une partie finie de la pièce ou la paroi d'une cavité. Par extension, le terme est parfois repris pour une interférence entre le porte-outil ou la broche et la pièce.

lors de l'établissement de la compensation continuera à être utilisé jusqu'à la désactivation de la compensation, même si un nouvel outil est effectivement utilisé.

Quand la compensation de rayon d'outil est active, il est physiquement possible de faire un cercle, dont le rayon est la moitié du diamètre de l'outil donné dans la table d'outils, il sera tangent avec l'outil en tout point de son contour.

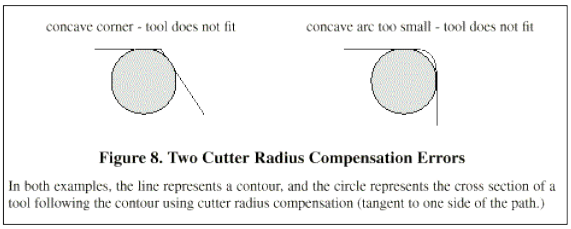


Figure 8. Two Cutter Radius Compensation Errors

In both examples, the line represents a contour, and the circle represents the cross section of a tool following the contour using cutter radius compensation (tangent to one side of the path.)

En particulier, l'interpréteur traite les coins concaves et les arcs concaves plus petits que l'outil, comme des erreurs, le cercle ne peut pas être maintenu tangent avec le contour dans ces situations. Cette détection de défaut, ne limite pas les formes qui peuvent être usinées, mais elle requiert que le programmeur précise la forme exacte à usiner (ou le chemin d'outil qui doit être suivi) et non une approximation. A cet égart, l'interpréteur utilisé par EMC diffère des interpréteurs utilisés dans beaucoup d'autres contrôleurs, qui passent ces erreurs sous silence et laissent l'outil interférer avec la partie finie de la pièce (gouging) ou arrondissent des angles qui devraient être vifs. Il n'est pas nécessaire, de déplacer l'outil entre la désactivation de la compensation et sa réactivation, mais le premier mouvement suivant la réactivation sera considéré comme premier mouvement, comme déjà décrit plus tôt.

Il n'est pas possible de passer d'un index de rayon d'outil à un autre alors que la compensation est active. Il est également impossible de basculer la compensation d'un côté à l'autre avec la compensation active. Si le prochain point de destination XY est déjà dans le périmètre d'action de l'outil quand la compensation est activée, le message indiquant une interférence outil/surface finie, s'affichera quand la ligne du programme qui donne cette destination sera atteinte. Dans ce cas, l'outil a déjà usiné dans le matériau, là où il n'aurait pas dû...

Si le numéro de slot programmé par le mot D est supérieur au nombre d'emplacements disponibles dans le carrousel, un message d'erreur sera affiché. Dans l'implémentation actuelle, le nombre d'emplacements maximum est de 68.

Le message d'erreur "Deux G-codes du même groupe modal sont utilisés" est un message générique utilisé pour plusieurs jeux de G-codes. Il s'applique à la compensation de rayon d'outil, il signifie que plus d'un code G40, G41 ou G42 apparaît sur la même ligne de progamme NC, ce qui n'est pas permis.

Premier mouvement

L'algorithme utilisé lors du premier déplacement, quand c'est une ligne droite, consiste à tracer une droite, depuis le point d'arrivée, tangente à un cercle dont le centre est le point actuel, et le rayon, celui de l'outil. Le point de destination de la pointe de l'outil se trouve alors au centre d'un cercle de même rayon, tangeant à la ligne droite tracée précédemment. C'est montré sur la figure 9. Si le point programmé est situé à l'intérieur de la première section d'outil (le cercle de gauche), une erreur sera signalée.

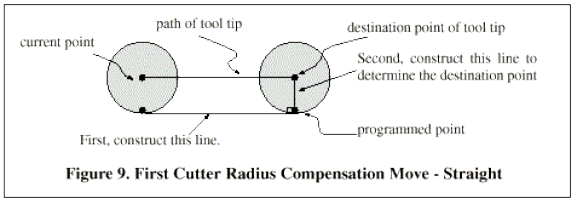


Figure 9. First Cutter Radius Compensation Move - Straight

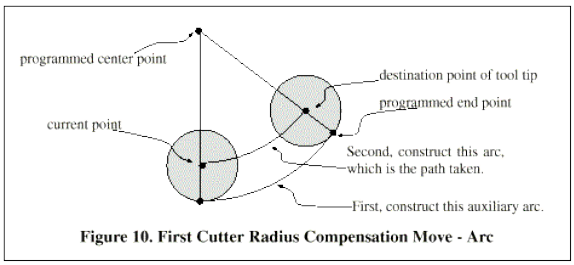


Figure 10. First Cutter Radius Compensation Move - Arc

Si le premier mouvement après que la compensation de rayon d'outil a été activée est un arc, l'arc qui sera généré est dérivé d'un arc auxiliaire qui, a son centre identique à celui du point central programmé, passe par le point final de l'arc programmé et, est tangent à l'outil à son emplacement courant. Si l'arc auxiliaire ne peut pas être construit, une erreur sera signalée. L'arc généré déplacera l'outil pour qu'il reste tangeant à l'arc auxiliaire pendant tout le mouvement. C'est montré sur la figure 10.

Indépendamment du fait que le premier déplacement est une droite ou un arc, l'axe Z peut aussi se déplacer en même temps. Il se déplacera linéairement, comme c'est le cas quand la compensation de rayon n'est pas utilisée. Les mouvements des axes rotatifs (A, B et C) sont autorisés avec la compensation de rayon d'outil, mais leur utilisation serait vraiment très inhabituelle.

Après les mouvements d'entrée en compensation de rayon d'outil, l'interpréteur maintiendra l'outil tangent au contour programmé et du côté approprié. Si un angle aigü se trouve dans le chemin, un arc est inséré pour tourner autour de l'angle. Le rayon de cet arc sera de la moitiè du diamètre de l'outil donné dans la table d'outils.

Quand la compensation de rayon est désactivée, aucun mouvement de sortie particulier n'est fait. Le mouvement suivant sera ce qu'il aurait été si la compensation n'avait jamais été activée et que le mouvement précédent ait placé l'outil à sa position actuelle.

Programmation des mouvements d'entrée

En général, un mouvement d'alignement et deux mouvements d'entrée sont demandés pour commencer la compensation correctement. Cependant, si le contour programmé comporte des pointes et des angles aigüs, un seul mouvement d'entré (plus, éventuellement, un mouvement de pré-entrée) est demandé. La méthode générale, qui fonctionne dans toutes les situations, est décrite en premier. Elle suppose que le programmeur connait déjà le contour et son but est d'ajouter le mouvement d'entrée.

Méthode générale

La méthode générale de programmation comprend un mouvement d'alignement et deux mouvements d'entrée. Les mouvements d'entrée expliqués ci-dessus, seront repris comme exemple. Voici le code correspondant :

```
N0010 G1 X1 Y4.5 (mouvement d'alignement vers le point C)
N0020 G41 G1 Y3.5 (active la compensation et fait le premier mouvement d'entrée vers le point
B)
N0030 G3 X2 Y2.5 I1 (fait le second mouvement d'entrée vers le point A)
```

Voir la figure 11. La figure montre les deux mouvements d'entrée mais pas le mouvement d'alignement.

En premier, choisir un point A sur le contour où il convient d'attacher un arc d'entrée. Spécifier un arc à l'extérieur du contour qui commence au point B et s'achève au point A, tangent au contour (et aller dans la même direction que celle prévue pour tourner autour du contour). Le rayon doit être supérieur à la moitié du diamètre donné dans la table d'outils. Ensuite, tirer une ligne tangente à l'arc, du point B au point C, placé de telle sorte que la ligne BC fasse plus d'un rayon de long.

Après que la construction soit terminée, le code est écrit dans l'ordre inverse de celui de la construction. La compensation de rayon d'outil est activée après le mouvement d'alignement et avant le premier mouvement d'entrée. Dans le code précédent, la ligne N0010 fait le mouvement d'alignement, la ligne N0020 active la compensation et fait le premier mouvement d'entrée et la ligne N0030 fait le second mouvement d'entrée.

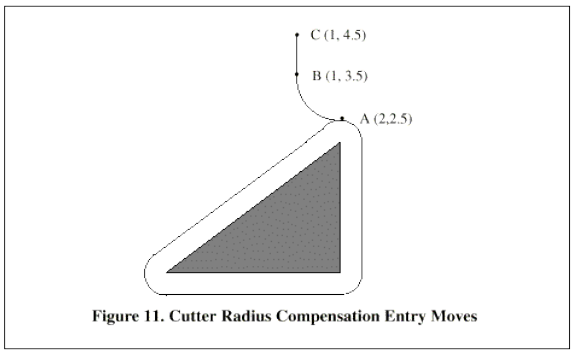


Figure 11. Cutter Radius Compensation Entry Moves

Dans cet exemple, l'arc AB et la ligne BC sont très larges, ce n'est pas nécessaire. Pour un contour sur chemin d'outil, le rayon de l'arc AB demande juste à être légérement plus grand que la variation

maximale du rayon de l'outil par rapport à son rayon nominal. Egalement, pour un contour sur chemin d'outil, le côté choisi pour la compensation doit être celui utilisé si l'outil est surdimensionné. Comme mentionné précédemment, si l'outil est sousdimensionné, l'interpréteur basculera de l'autre côté.

Méthode simple

Si le contour est sur le profil du matériau et qu'il comprends des angles aigüs quelque part sur le contour, une méthode simple pour faire l'entrée est possible. Voir la figure 12.

Premièrement, choisir un angle aigü, par exemple D. Ensuite, décider comment on va tourner autour du matériau depuis le point D. Dans notre exemple nous maintiendrons l'outil à gauche du profil et nous avancerons vers F. Prolonger la ligne FD (si le segment suivant du contour est un arc, prolonger la tangente à l'arc FD depuis D) pour diviser la surface extérieure au contour proche de D en deux parties. S'assurer que le centre de l'outil est actuellement dans la partie du même côté de la ligne prolongée que le matériau. Sinon, déplacer l'outil dans cette partie. Par exemple, le point E représente la position courante du centre de l'outil. Comme il est du même côté de la ligne FD prolongée que le triangle gris du matériau, aucun mouvement supplémentaire n'est nécessaire. Maintenant écrire la ligne de code NC qui active la compensation et faire le mouvement vers le point D

```
N0010 G41 G1 X2 Y2 (active la
compensation et fait le mouvement
d'entrée)
```

Cette méthode fonctionnera également avec avec un angle aigü sur un contour sur chemin d'outil, si l'outil est surdimensionné, mais elle échouera si il est sousdimensionné.

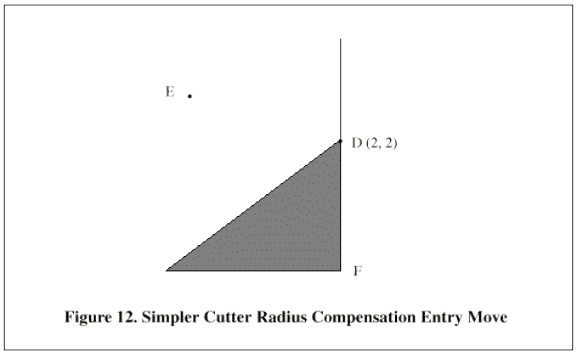


Figure 12. Simpler Cutter Radius Compensation Entry Move

Autres points où est exécutée la compensation de rayon d'outil

Le jeu complet de fonctions canoniques comprend des fonctions qui activent et désactivent la compensation de rayon d'outil, de sorte qu'elle puisse être activée quand le contrôleur exécute une de ces fonctions. Dans l'interpréteur, cependant, ces commandes ne sont pas utilisées. La compensation est assurée par l'interpréteur et reflétée dans les sorties des commandes, c'est l'interpréteur qui continuera à diriger les mouvements du centre de l'outil. Cela simplifie le travail du contrôleur de mouvement tout en rendant le travail de l'interpréteur un peu plus difficile.

Algorithmes pour compensation de rayon d'outil

L'interpréteur permet que les mouvements d'entrée et de sortie soient des arcs. Le comportement pour les mouvements intermédiaires est le même, excepté que certaines situations sont traitées comme des erreurs par l'interpréteur alors qu'elles ne le sont pas sur d'autres contrôleurs de machine.

Données relatives à la compensation de rayon d'outil :

L'interpréteur du modèle de machine conserve trois données pour la compensation de rayon d'outil : Le réglage lui même (gauche, droite ou arrêt), program_x, et program_y. Les deux dernières représentent les positions en X et en Y données dans le code NC quand la compensation est active. Quand elle est désactivée, les deux entrées sont fixées à de très petites valeurs (10 e-20) dont la valeur symbolique (dans un #define) est "unknown". L'interpréteur du modèle de machine utilise, les items current_x et current_y qui représentent, le centre de la pointe de l'outil (dans le système de coordonnées courant), à tous moments.

Exemples de Jon Elson

Toutes les informations spécifiques au système se réfèrent au programme EMC du NIST, mais doit aussi s'appliquer aux plus modernes contrôleurs CNC. Ma méthode de vérification de ces programmes est d'abord de sélectionner l'outil zéro, de sorte que les commandes de compensation soient ignorées. Ensuite, je colle une feuille de papier sur une plaque tenue de niveau dans l'étau, une sorte de platine. J'installe une recharge de stylo à ressort dans la broche. C'est une recharge standard de stylo à bille en métal avec un ressort, dans un emballage de 1/2" de diamètre. Elle à un ressort pour la faire rentrer dans le corps du stylo, et un 'collet' à l'arrière qui permet à la pointe de se rétracter malgrès le ressort, mais qui la laisse centrée à quelques centièmes de pouce près. Je charge le programme avec l'outil zéro sélectionné, et il trace une ligne à l'extérieur de la pièce. (voir la figure suivante) Alors, je sélectionne un outil avec le diamètre de l'outil que j'envisage d'utiliser et je lance le programme une nouvelle fois. (Noter que la coordonnée Z dans le programme ne doit pas être changée pour éviter de plonger le stylo au travers du plateau ;-) Maintenant, je dois voir si la compensation G41 ou G42 que je spécifie passe sur le côté voulu de la pièce. Sinon, je modifie avec la compensation du côté opposé, et j'édite la compensation opposée dans le programme, puis j'essaye à nouveau. Maintenant, avec l'outil sur le côté correct de la pièce, je peut vérifier si quelque part sur le parcours l'outil est 'trop gros' pour usiner les surfaces concaves. Ma vieille Allen-Bradley 7320 était très indulgente sur ce point, mais EMC ne tolère rien. Si vous avez la moindre concavité où deux lignes se rencontrent à moins de 180 degrés avec un outil de taille définies, EMC va s'arrêter là, avec un message d'erreur. Même si le gougeage est de .0001" de profondeur. Alors, je fais toujours l'approche sur le mouvement d'entrée et le mouvement de sortie juste sur un coin de la pièce, en fournissant un angle de plus de 180 degrés, afin qu'EMC ne hurle pas. Cela exige une grande attention lors de l'ajustement des points de départ et de sortie, qui ne sont pas compensés par le rayon d'outil, mais ils doivent être choisis avec un rayon approximatif bien réfléchi.

Les commandes sont :

G40 - Annuler la compensation de rayon d'outil
G41 - Activer la compensation, outil à gauche du profil
G42 - Activer la compensation, outil à droite du profil

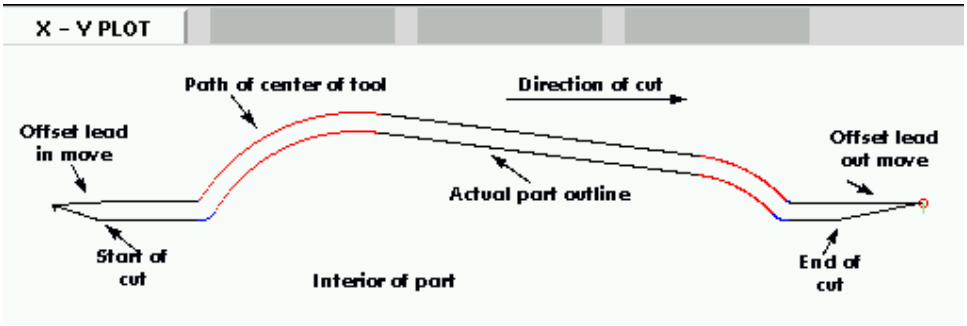
Voici un petit fichier qui usine le côté d'une pièce avec de multiples arcs convexes et concaves et plusieurs lignes droites. La plupart de ces commandes ont été tracées depuis Bobcad/CAM, mais les lignes N15 et N110 ont été ajoutées par moi et certaines coordonnées dans ce contour ont été bricolées un peu par moi.

```
N10 G01 G40 X-1.3531 Y3.4
N15 F10 G17 G41 D4 X-0.7 Y3.1875 (ligne d'entrée)
N20 X0. Y3.1875
N40 X0.5667 F10
N50 G03 X0.8225 Y3.3307 R0.3
```

```
N60 G02 X2.9728 Y4.3563 R2.1875
N70 G01 X7.212 Y3.7986
N80 G02 X8.1985 Y3.2849 R1.625
N90 G03 X8.4197 Y3.1875 R0.3
N100 G01 X9.
N110 G40 X10.1972 Y3.432 (ligne de sortie)
N220 M02
```

La ligne 15 contient G41 D4, qui signifie que le diamètre de l'outil est celui de l'outil #4 dans la table d'outils, il sera utilisé pour décaler la broche de 1/2 diamètre, qui est, bien sûr, le rayon d'outil. Noter que la ligne avec la commande G41 contient le point final du mouvement dans lequel la compensation de rayon est interpolée. Cela signifie qu'au début de ce mouvement, il n'y a aucun effet de compensation et à la fin, l'outil est décalé de 100% du rayon de l'outil sélectionné. Immédiatement après le G41 il y a D4, signifiant que le décalage sera le rayon de l'outil N°4 dans la table d'outils. Noter que les DIAMETRES d'outil sont entrés dans la table d'outils. (le diamètre de l'outil de Jon est de 0.4890)

Mais, noter qu'à la ligne 110, où il y a la commande G40, l'interpolation de la compensation d'outil est en dehors de ce mouvement. La manière d'obtenir ce réglage, les mouvements des lignes 15 et 110 sont presque exactement parallèles à l'axe X et la différence dans les coordonnées Y est à la ligne où l'outil est appelé, en dehors de la compensation d'outil.



Certaines autres choses sont à noter, le programme commence avec G40, pour désactiver les compensations éventuellement actives. Cela évite un tas d'ennuis quand le programme s'arrête à cause d'une erreur de concavité, mais laisse la compensation désactivée. Noter aussi, en ligne 15, G17 est utilisé pour spécifier le plan de travail XY pour les interpolations circulaires. J'ai utilisé le format rayon pour les spécifications des arcs plutôt que la forme I, J. EMC est trés pointilleux au sujet des rayons qu'il calcule à partir du format des coordonnées I, J et il doit trouver le début et la fin du mouvement avec 10^-11 unités internes, de sorte qu'il y a beaucoup de problèmes avec des arcs arbitraires. Normalement, si vous avez un arc de 90 degrés, centré sur (1.0,1.0) avec un rayon de 1", tout ira bien, mais si le rayon ne peut pas être exprimé exactement et avec juste le nombre de chiffres significatifs, ou si l'arc à un nombre étrange de degrès, alors les problèmes commencent avec EMC. Le mot R supprime tous ce désordre et il est beaucoup plus facile de travailler avec lui, de toute façon. Si l'arc est de plus de 180 degrés, R doit être négatif.

18.5 Source d'informations à propos de la compensation d'outil

This unit borrows heavily from the published works of Tom Kramer and Fred Proctor at NIST and the cutter compensation web page of Jon Elson.

Papers by Tom Kramer and Fred Proctor
http ://www.isd.mel.nist.gov/personnel/kramer/publications.html

http ://www.isd.mel.nist.gov/personnel/kramer/pubs/RS274NGC_22.pdf
http ://www.isd.mel.nist.gov/personnel/kramer/pubs/RS274VGER_11.pdf

Pages by Jon Elson
http ://artsci.wustl.edu/~jmelson/
http ://206.19.206.56/diacomp.htm
http ://206.19.206.56/lencomp.htm

Chapitre 19

Différences entre le g-code d'EMC2 et le RS274NGC

19.1 Différences changeant la signification d'un programme correctement écrit en RS274NGC

19.1.1 Position après un changement d'outil

Dans EMC2, la machine ne revient pas à sa position initiale après un changement d'outil. Ce choix a été fait parce que le nouvel outil pourrait être plus long que l'ancien, rendant le retour de la machine à sa position initiale, impossible sans heurter la pièce.

19.1.2 Les décalages sont dans les unités du fichier ini

Dans EMC2, les valeurs stockées dans les paramètres des points d'origine G28 et G30, des systèmes de coordonnées P1. . . P9 et du décalage G92 sont dans les unités définies dans le fichier ini. Cette différence existe parce que, sinon, la signification d'un emplacement changeait selon que G20 ou G21 était actif lorsque G28, G30, G10, L2, ou G92.3 était programmé.

19.1.3 Les longueurs/diamètres de la table d'outils sont dans les unités du fichier ini

Dans EMC2, les longueurs (compensation) et diamètres de la table d'outils sont seulement dans les unités du fichier ini. Cette différence existe parce que autrement, la longueur d'un outil et son diamètre aurait changé selon que G20 ou G21 aurait été actif dans les modes G43, G41, G42. Ce qui aurait rendu impossible, sans modification de la table d'outils, l'exécution de g-code avec la machine en unités non natives, même pour un g-code bien écrit (ligne en début avec G20 ou G21 et mêmes unités tout au long du programme).

19.1.4 G84, G87 non implémentés

G84 et G87 ne sont pas actuellement implémentés, mais serons peut être ajoutés à de futures versions d'emc2.

19.1.5 G28, G30 avec des mots d'axes

Quand `G28` ou `G30` est programmé avec un mot d'axe présent, EMC2 déplace cet axe seul. Il est commun sur les autres contrôles machine. Pour déplacer certains axes vers un point intermédiaire puis les déplacer tous vers un point pré-défini, écrire deux lignes de gcode :

```
G0 X- Y- (Les axes se déplacent vers le point intermédiaire)
G28 (Les axes se déplacent vers leurs origines)
```

19.1.6 M62, M63 non implémentés

M62 et M63 ne sont pas encore implémentés, mais serons peut être ajoutés à de futures versions d'emc2..

19.1.7 G0 réduit les longs mouvements angulaires pour les rendre inférieurs à un tour

Comme décrit précédemment, G0 réduit les mouvements angulaires à moins d'un tour. Ce qui peut produire des trajectoires différentes de celles produites avec RS274NGC.

19.2 Différences ne changeant pas la signification des programmes bien écrits en RS274NGC

19.2.1 G33, G76 codes de taraudage

Ces codes ne sont pas définis dans RS274NGC.

19.2.2 G38.2

La pointe de la sonde n'est pas rétractée après un mouvement G38.2. Ce mouvement de retrait sera peut être ajouté dans de futures versions d'emc2.

19.2.3 G38.3. . . G38.5

Ces codes ne sont pas définis dans RS274NGC

19.2.4 O-codes

Ces codes ne sont pas définis dans RS274NGC

19.2.5 M50. . . M53 survitesses

Ces codes ne sont pas définis dans RS274NGC

19.2.6 G43, G43.1

19.2.6.1 Longueurs d'outil négatives

Les spécifications du RS274NGC disent "il est prévu que" toutes les longueurs d'outils soient positives. Toutefois, G43 fonctionne aussi avec des longueurs d'outils négatives.

19.2.6.2 Outils de tour

La compensation de longueur d'outil G43 peut décaler l'outil dans les deux dimensions X et Z. Cette fonctionnalité est très utile sur les tours.

19.2.6.3 Longueurs d'outil dynamiques

EMC2 permet la spécification d'une longueur d'outil calculée par `G43.1 I K`.

19.2.7 G41.1, G42.1

EMC2 permet la spécification dans le gcode d'un diamètre d'outil et sur un tour, de son orientation. Le format est `G41.1/G42.1 D- L-`, où `D` est le diamètre et `L` (si spécifié) est l'orientation de l'outil de tour.

19.2.8 G43 sans mot H

Dans ngc, cela n'est pas permis. Dans EMC2, il règle la cote de longueur du décalage pour l'outil courant. Si aucun outil n'est chargé actuellement, c'est une erreur. Cette différence a été apportée pour que l'utilisateur n'ait pas à spécifier le numéro d'outil à deux endroits pour chaque changement d'outil et parce que c'est plus cohérent avec la manière de travailler de G`41/G42` quand le mot D n'est pas spécifié.

19.2.9 Axes U, V et W

EMC2 gère des machines avec un maximum de 9 axes en définissant un jeu supplémentaire de 3 axes linéaires nommés U, V et W.

Chapitre 20

Systèmes de coordonnées et décalage G92

20.1 Introduction

Vous avez vu comme il est pratique d'utiliser la compensation de longueur d'outil, elle évite au programmeur d'avoir à connaitre en tout temps la hauteur de l'outil quand il écrit un programme. De la même manière, il est très intéressant d'utiliser un point de référence sur le brut ou la pièce et de faire travailler le programme à partir de ce point, sans avoir à tenir compte d'où est placée la pièce sur la machine pendant l'usinage.

Ce chapitre va introduire les décalages et comment ils sont utilisés par EMC. Ce qui inclus :

– Les coordonnées machine (G53)
– Les neuf décalages (G54 à G59.3)
– Un jeu de décalages globaux (G92).

20.2 Commande en coordonnées machine : G53

Indépendamment de tout décalage pouvant être actif, un G53 dans une ligne de code indique à l'interpréteur de se déplacer aux positions réelles des axes (absolues), commandées dans la ligne. Par exemple :

g53 g0 x0 y0 z0

déplacera le mobile depuis la position actuelle vers la position où les coordonnées des trois axes seront à zéro. Vous pouvez utiliser cette commande si vous avez une position fixe pour le changement d'outil ou si votre machine dispose d'un changeur automatique. Vous pouvez aussi utiliser cette commande pour dégager la zone de travail et accéder à la pièce dans l'étau.

G53 est une commande non modale. Elle doit être utilisée sur chaque ligne où un mouvement basé sur les coordonnées machine est souhaité.

20.3 Décalages pièce (G54 à G59.3)

Le décalage d'origine est utilisé pour séparer le point de référence de la pièce, de l'origine machine, créant ainsi un système de coordonnées (relatif), propre à chaque pièce et séparé du système de coordonnées machine (absolu). Il permet, entre autre, dans le cas de pièces multiples mais semblables, de créer en décalant ses origine, le système de coordonnées de chaque pièce, le programme restant

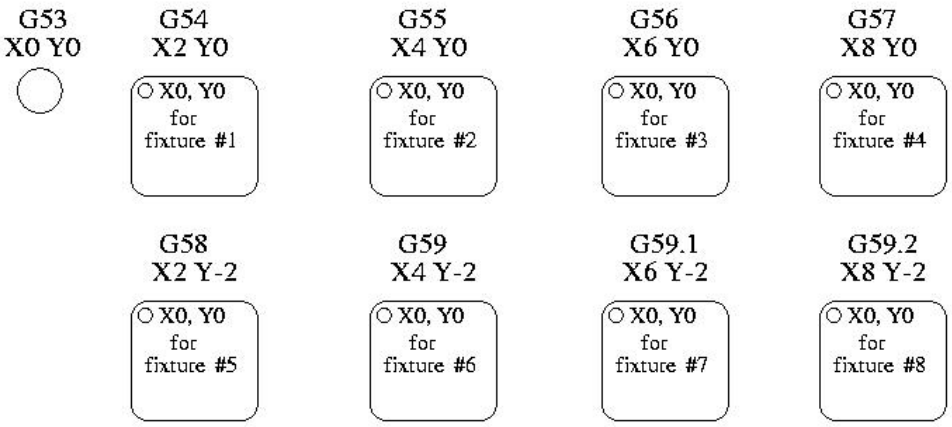


FIG. 20.1 – Décalages pour 8 ilots

le même. Un cas typique d'utilisation de cette fonctionnalité, pour usiner huit ilots identiques sur la même pièce est illustré sur la figure 20.1

Les valeurs des décalages sont enregistrées dans le fichier VAR qui est requis par le fichier INI durant le démarrage d'EMC. Dans l'exemple ci-dessous qui utilise G55 la valeur de chacun des axes de G55 est enregistrée dans une variable numérotée.

```
5241    0.000000
5242    0.000000
5243    0.000000
5244    0.000000
5245    0.000000
5246    0.000000
```

Dans le schéma d'un fichier VAR, la première variable contient la valeur du décalage de l'axe X, la seconde variable le décalage de l'axe Y et ainsi de suite pour les six axes. Il y a une série de variables de ce genre pour chacun des décalages pièce.

Chacune des interfaces graphiques offre un moyen de fixer les valeurs de ces décalages. Vous pouvez également définir ces valeurs en modifiant le fichier VAR lui-même, puis faire un [reset], pour qu' EMC lise les nouvelles valeurs. Pour notre exemple, nous allons éditer directement le fichier pour que G55 prenne les valeurs suivantes :

```
5241    2.000000
5242    1.000000
5243   -2.000000
5244    0.000000
5245    0.000000
5246    0.000000
```

Vous pouvez voir que les positions zéro de G55 sont passées en X=2, Y=1 et Z=-2 éloignées donc de l'origine absolue (machine).

Une fois que les valeurs sont assignées, un appel de G55 dans une ligne de programme décalera le point de référence zéro, l'origine, vers les valeurs enregistrées. La ligne suivante décalera chacun des axes vers sa nouvelle position d'origine. Contrairement à G53, les commandes G54 à G59.3 sont modales. Elles agissent sur toutes les lignes de g-code du programme après qu'une ait été rencontrée. Voyons le programme qui pourrait être écrit à l'aide de la figure20.1, il suffira d'un seul point de référence pour chacune des pièces pour faire tout le travail. Le code suivant est proposé

comme exemple pour faire un rectangle, il utilisera les décalages G55 que nous avons expliqué ci-dessus.

```
G55 G0 x0 y0 z0
g1 f2 z-0.2000
x1
y1
x0
y0
g0 z0
g54 x0 y0 z0
m2
```

“Mais,” dites vous, “pourquoi y a-t-il un G54 vers la fin?” C’est une pratique courante de quitter le système de coordonnées G54 avec l’ensemble des valeurs d’axes à zéro afin de laisser un code modal basé sur les positions machine absolues. Nous le faisons avec cette commande qui met la machine à zéro. Il aurait été possible d’utiliser G53 et d’arriver au même endroit, mais la commande n’aurait pas été modale, les commandes suivantes auraient voulu retourner dans le système de coordonnées du G55 toujours actif.

G54 utilise les réglages du système de coordonnées 1
G55 utilise les réglages du système de coordonnées 2
G56 utilise les réglages du système de coordonnées 3
G57 utilise les réglages du système de coordonnées 4
G58 utilise les réglages du système de coordonnées 5
G59 utilise les réglages du système de coordonnées 6
G59.1 utilise les réglages du système de coordonnées 7
G59.2 utilise les réglages du système de coordonnées 8
G59.3 utilise les réglages du système de coordonnées 9

20.3.1 Système de coordonnées par défaut

Une autre variable dans le fichier VAR joue un rôle important dans les décalages, c’est la variable 5220. Dans les fichiers par défaut, sa valeur est fixée à 1,00000. Ce qui signifie que lorsque EMC démarre, il doit utiliser le premier système de coordonnées comme système par défaut. Si vous définissez celui-ci à 9,00000 le neuvième système décalé sera utilisé par défaut au démarrage du système et aux réinitialisations. Toute valeur autre qu’un entier compris entre 1 et 9 provoquera une erreur au démarrage d’EMC.

20.3.2 Réglage des décalages avec G10

La commande G10 L- peut être utilisée pour modifier les valeurs des décalages dans un système de coordonnées. (add here)

20.4 G92 Décalages d’axes

G92 est la plus incomprise et la plus maligne des commandes programmables avec EMC. La façon dont elle fonctionne a un peu changé entre les premières versions et l’actuelle. Ces changements ont sans doute déconcerté de nombreux utilisateurs. Elle devrait être vue comme une commande produisant un décalage temporaire, qui s’applique à tous les autres décalages.

En réponse à une critique de cette commande, Ray Henry l'a étudiée en comparant la façon dont les auteurs de l'interpréteur s'attendaient à ce qu'elle fonctionne et la façon dont elle fonctionnait sur sa mini-fraiseuse Grizzly. Dans les paragraphes ci-dessous, entre guillemets, vous trouverez des extraits de son document, disponible sur http://www.linuxcnc.org.

20.4.1 Les commandes G92

Ce jeu de commandes inclus :

G92 Cette commande, utilisée avec des mots d'axes, fixe les valeurs des variables de décalage.

G92.1 Cette commande met à zéro les valeurs des variables de g92.

G92.2 Cette commande suspend, sans les mettre à zéro, les variables de g92.

G92.3 Cette commande applique les valeurs de décalage qui ont été suspendues.

Lorsque les commandes sont utilisées comme décrit ci-dessus, elles fonctionneront à peu près comme vous vous y attendiez.

> L'utilisateur doit bien comprendre le fonctionnement des valeurs de g92. Elles sont basées sur l'emplacement de chaque axe au moment où la commande g92 est invoquée. Le document du NIST est clair, " pour faire en sorte que le point actuel ait les coordonnées" x0, y0 et z0 nous utiliserons g92 x0 y0 z0. G92 *ne fonctionne pas depuis le système de coordonnées machine absolues. Il fonctionne à partir de l'emplacement actuel.*
>
> G92 travaille également à partir d'un emplacement actuel déjà modifié par tout autre décalage actif au moment où la commande g92 est invoquée. Lors de test des différences entre les décalages de travail et les décalages réels, il a été constaté qu'un décalage g54 pouvait annuler un g92 et ainsi, donner l'apparence qu'aucun décalage n'était actif. Toutefois, le g92 était toujours actif, pour toutes les coordonnées et il a produit les décalages attendus pour tous les autres systèmes de coordonnées.
>
> Il est probable que l'absence de contact d'origine machine et d'une bonne procédure de prise d'origine machine se traduira par de très grandes erreurs dans l'application des valeurs de g92 si elles existent dans le fichier VAR. Beaucoup d'utilisateurs d'EMC n'ont pas de contact d'origine machine sur leurs machines. Pour eux, l'origine machine devrait être trouvée en déplaçant chaque axe à l'emplacement supposé et en utilisant la commande de prise d'origine. Lorsque chaque axe est dans une position connue, la commande de prise d'origine recalculera comment les valeurs de g92 doivent être appliquées pour produire des résultats cohérents. Sans séquence de prise d'origine machine, les valeurs sont appliquées à la position de la machine au démarrage d'EMC.

20.4.2 Réglage des valeurs de G92

Il y a au moins deux façons d'établir les valeurs de G92.

- Par un clic droit de la souris sur les afficheurs de position de tkemc, une fenêtre s'ouvre dans laquelle il est possible de saisir une valeur.
- Par la commande G92.

Toutes les deux, fonctionnent depuis l'emplacement courant de l'axe auquel le déplacement doit être appliqué.

> Programmer g92 x- y- z- a- b- c- fixe les valeurs des variables de G92 de sorte que chaque axe prenne la valeur associée à son nom. Ces valeurs sont assignées à la position courante des axes. Ces résultats satisfont les paragraphes un et deux du document du NIST.
>
> Les commandes G92 fonctionnent à partir de la position courante de l'axe, à laquelle elles ajoutent ou soustraient des valeurs pour donner à la position courante la valeur assignée par la commande g92. Elles prennent effet même si d'autres décalages sont déjà actifs.

Ainsi, si l'axe X est actuellement en position X=2.0000, un G92 x0 fixera un décalage de -2.0000, de sorte que l'emplacement actuel de X devienne X=0.0000. Un nouveau G92 X5.0000 fixera un décalage de 3.0000 et l'affichage indiquera une position courante X=5.0000.

Maintenant, un G92 X5 fixera un décalage de 0.0000 et l'affichage ne changera pas.

20.4.3 Prudence avec G92

Parfois, les valeurs de décalage d'un G92 restent bloquées dans le fichier VAR. Quand ça arrive, une ré-initialisation ou un redémarrage peut les rendre de nouveau actifs. Les variables sont numérotées

5211 0.000000
5212 0.000000
5213 0.000000
5214 0.000000
5215 0.000000
5216 0.000000

où 5211 est le numéro du décalage de l'axe X et ainsi de suite. Si vous voyez des positions inattendues à la suite d'une commande de déplacement, ou même des chiffres inattendus dans l'affichage de la position lorsque vous démarrez, regardez ces variables dans le fichier VAR pour vérifier si elles contiennent des valeurs. Si c'est le cas, les mettre à zéro devrait solutionner le problème.

> Avec ces tests, nous pouvons voir qu'après une réinitialisation, G92 retourne aux conditions qu'il avait au démarrage de l'interpréteur. Le lecteur doit noter que nous avons établi ... qu'aucune de ces valeurs n'est écrite durant un démarrage normal si aucun G92 n'a été fixé au démarrage, aucune ne pourra être lue lors d'un redémarrage.
>
> Il est possible que ce soit le coeur du problème que certains ont vécu avec les différences entre l'ancien et le nouvel interpréteur. C'est possible, mais je laisse le soin à d'autres de faire des essais, pour vérifier que l'ancien interpréteur et les programmes écrivent les variables immédiatement et de constater ensuite que ces valeurs sont lues lors d'un redémarrage.

D'autre part, si les valeurs de G92 existent dans le fichier VAR lorsque EMC a démarré

> ... À partir d'EMC avec les valeurs de g92 dans le fichier de variables, EMC appliquera ces valeurs à partir de l'emplacement actuel de chaque axe. Si la prise d'origine machine a été faite et que l'origine machine correspond bien au zéro des axes, tout sera correct. Une fois que la prise d'origine a été faite en utilisant les contacts d'origine machine ou en déplaçant chaque axe à une position connue et en appliquant la commande de prise d'origine, les commandes g92 et leurs valeurs fonctionnent comme prévu.
>
> Ces tests n'ont pas étudié l'effet produit par la lecture des variables dans le fichier alors qu'il contient des valeurs. Cela pourrait poser des problèmes si les décalages g92 avaient été enlevés avec g92.1 alors que le fichier de variables contenait encore les valeurs précédentes.

C'est cette complexité qui nous amène à dire que les valeurs de G92 doivent être considérées comme provisoires. Elles devraient être utilisées pour définir des décalages globaux et à court terme. Les commandes de décalage d'origine G54 à 59.3 devraient être utilisées chaque fois que des décalages durables et prévisibles sont nécessaires.

20.5 Exemple de programme utilisant les décalages d'axes

Cet exemple de projet de gravure fraise un jeu de quatre cercles de rayon .1 pouce dans une forme grossière d'étoile au centre du cercle. Nous pouvons configurer individuellement les formes de la façon suivante :

```
G10 L2 P1 x0 y0 z0 (assure que g54 a mis la machine à zéro)
g0 x-.1 y0 z0
g1 f1 z-.25
g3 x-.1 y0 i.1 j0
g0 z0
m2
```

Nous pouvons émettre une série de commandes pour créer des décalages pour les quatre autres cercles comme cela.

```
G10 L2 P2 x0.5 (offsets g55 x value by 0.5 inch)
G10 L2 P3 x-0.5 (offsets g56 x value by -0.5 inch)
G10 L2 P4 y0.5 (offsets g57 y value by 0.5 inch)
G10 L2 P5 y-0.5 (offsets g58 y value by -0.5 inch)
```

Nous mettons ces ensemble dans le programme suivant.

```
(Un programme de fraisage de cinq petits cercles dans un losange)
G10 L2 P1 x0 y0 z0 (assure que g54 a mis la machine à zéro)
G10 L2 P2 x0.5 (offsets g55 x value by 0.5 inch)
G10 L2 P3 x-0.5 (offsets g56 x value by -0.5 inch)
G10 L2 P4 y0.5 (offsets g57 y value by 0.5 inch)
G10 L2 P5 y-0.5 (offsets g58 y value by -0.5 inch)
g54 g0 x-.1 y0 z0 (center du cercle)
g1 f1 z-.25
g3 x-.1 y0 i.1 j0
g0 z0
g55 g0 x-.1 y0 z0 (premier décalage circulaire)
g1 f1 z-.25
g3 x-.1 y0 i.1 j0
g0 z0
g56 g0 x-.1 y0 z0 (second décalage circulaire)
g1 f1 z-.25
g3 x-.1 y0 i.1 j0
g0 z0
g57 g0 x-.1 y0 z0 (troisième décalage circulaire)
g1 f1 z-.25
g3 x-.1 y0 i.1 j0
g0 z0
g58 g0 x-.1 y0 z0 (décalage circulaire)
g1 f1 z-.25
g3 x-.1 y0 i.1 j0
g54 g0 x0 y0 z0
m2
```

Maintenant vient le moment où l'on applique une série de décalages G92 à ce programme. Vous verrez qu'il c'est fait dans chaque cas de z0. Si la machine étaient à la position de zéro, un g92 z1.0000 placé en tête de programme le décalera d'un pouce. Vous pouvez également modifier l'ensemble du dessin dans le plan XY en ajoutant quelques décalages x et y avec g92. Si vous faites cela, vous devez ajouter une commande G92.1 juste avant le m2 qui termine le programme. Si vous ne le faites pas, d'autres programmes que vous pourriez lancer après celui-ci, utiliseront également les décalages g92. En outre, cela permettrait d'éviter d'écrire les valeurs de g92 lorsque vous arrêtez EMC et donc, d'éviter de les recharger quand vous démarrez à nouveau le programme.

Chapitre 21

Cycles préprogrammés

Les cycles préprogrammés G81 à G89 sont implémentés comme décrit dans cette section.

Tous les cycles préprogrammés sont effectués dans le respect du plan de travail courant. N'importe lequel des trois plans de travail XY, YZ, ZX peut être choisi. Dans cette section, la plupart des descriptions supposeront que le plan de travail XY est le plan courant. Le comportement reste analogue pour les plans de travail YZ ou XZ.

Les mots d'axes rotatifs sont autorisés (sous conditions) dans les cycles préprogrammés, mais il est préférable de les omettre. Si les mots d'axes rotatifs sont utilisés, leurs valeurs doivent rester les mêmes que celles de la position courante, de sorte qu'ils ne tournent pas.

Tous les cycles préprogrammés utilisent X-, Y-, R-, et Z- dans le code NC. Ces valeurs sont utilisées pour déterminer les positions de X, Y, R, et Z. La position de R- (signifiant rétraction) est perpendiculaire au plan de travail courant (axe Z pour le plan XY, axe X pour le plan YZ, axe Y pour le plan XZ). Quelques cycles préprogrammés utilisent des arguments supplémentaires.

Dans les cycles préprogrammés, un nombre est appelé "sticky" (collant) si, quand le même cycle est répété sur plusieurs lignes de code en colonne, le nombre doit être indiqué la première fois, mais qu'il devient optionnel pour le reste des lignes suivantes. Les nombres "sticky" conservent leur valeur tant qu'ils ne sont pas explicitement programmés avec une nouvelle valeur. La valeur de R- est toujours "sticky".

En mode de déplacements incrémentaux (G91) : quand le plan courant est XY, les valeurs X-, Y-, est R- sont traitées comme incrémentales à partir de la position courante et Z- comme un incrément depuis la position précédent le mouvement impliquant l'axe Z. Quand le plan YZ ou XZ est le plan courant, le traitement des mots d'axes est analogue. En mode de déplacements absolus, les valeurs de X-, Y-, R-, et Z- sont des positions absolues dans le système de coordonnées courant.

La valeur L- est optionnelle, elle représente le nombre de répétitions. L=0 n'est pas permis. Si les répétitions sont utilisées, elles le sont normalement en mode de déplacements incrémentaux, de sorte que la même séquence de mouvements puisse être répétée à plusieurs endroits, égalements espacés, le long d'un ligne droite. En mode de déplacements absolus, L>1 signifie "faire le même cycle au même endroit, plusieurs fois". L'omission du mot L revient à spécifier L=1. La valeur de L- n'est pas "sticky".

Avec L>1 en mode incrémental et XY comme plan courant, les positions X et Y sont déterminées en ajoutant les valeurs X- et Y- de la commande à celles de la position courante, pour le premier trajet ou, ensuite, à celles de la position finale du précédent trajet, pour les répétitions. Les valeurs de R- et de Z- ne changent pas durant toutes les répétitions.

La hauteur du mouvement de retrait à la fin de chaque répétition (appellée "plan de retrait" dans les descriptions suivantes) est déterminée par le passage en mode : retrait sur la position initiale de Z, si elle est au dessus de la valeur de R- et que le mode de retrait est G98, OLD_Z, sinon, à la position de R-.

21.1 Mouvement préliminaire

Tout au début de l'exécution d'un cycle préprogrammé, avec le plan courant XY, si la position actuelle de Z est en dessous de la position de retrait R, l'axe Z va à la position R. Ceci n'arrive qu'une fois, sans tenir compte de la valeur de L.

En plus, au début du premier cycle et à chaque répétition, un ou deux des mouvements suivants sont faits :

1. Un déplacement en ligne droite, parallèle au plan XY, vers le position programmée. 2. Un déplacement en ligne droite, de l'axe Z seul vers la position de retrait R, si il n'est pas déjà à cette position R.

Si un des plans XZ ou YZ est actif, le mouvement préliminaire et intermédiaire est analogue.

21.2 G80 Révocation des codes modaux

G80 Programmer G80 pour s'assurer qu'aucun mouvement d'axe ne surviendra sans G-code modal.

Dans l'interpréteur d'EMC, G80 est un code modal révoqué par tout autre g-code. Les résultats des lignes suivantes sont identiques :

```
N1000 G90 G81 X1 Y1 Z1.5 R2.8 (cycle préprogrammé en mode de déplacements absolus)
N1001 G80 (révoque G81)
N1002 G0 X0 Y0 Z0 (active les mouvements en vitesse rapide et déplace le mobile en X0, Y0 et Z0)
```

produit le même déplacement et le même état final de la machine que :

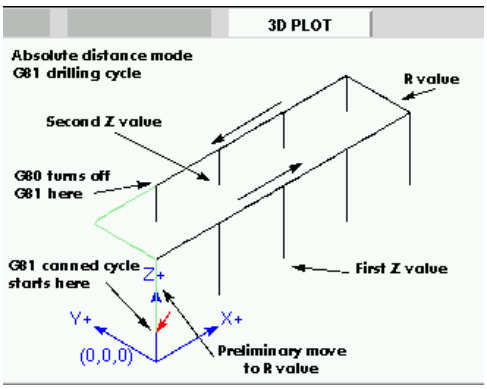
```
N1000 G90 G81 X1 Y1 Z1.5 R2.8 (cycle préprogrammé en mode de déplacements absolus)
N1001 G0 X0 Y0 Z0 (active les mouvements en vitesse rapide et déplace le mobile en X0, Y0 et Z0)
```

L'avantage du premier, la ligne du G80 révoque clairement le cycle G81. Avec ce premier programme, le programmeur doit revenir en mode mouvement avec G0, comme c'est fait sur la ligne suivante, ou tout autre mot G de mouvement.

Exemple 0 : Utilisation d'un cycle préprogrammé avec un code de mouvement modal

Si un cycle préprogrammé n'est pas révoqué avec G80 ou un autre mot G de mouvement, le cycle préprogrammé attend de se répéter en utilisant le prochaine ligne de code contenant un (ou plusieurs) mot d'axe X, Y ou Z. Le fichier suivant perce (G81) un ensemble de huit trous. Noter la position de Z change après les quatre premiers trous.

```
N100 G90 G0 X0 Y0 Z0 (coordonnées d'origine)
N110 G1 X0 G4 P0.1
N120 G81 X1 Y0 Z0 R1 (cycle de perçage)
N130 X2

N140 X3
N150 X4
N160 Y1 Z0.5
N170 X3
N180 X2
N190 X1
N200 G80 (révocation du cycle G81)
N210 G0 X0 (mouvement en vitesse rapide)
N220 Y0
N220 Z0
N220 M2 (fin du programme)
```



L'utilisation de G80 à la ligne N200 est optionnel puisqu'il y a un G0 sur la ligne suivante qui révoque le cycle G81. Mais utiliser G80, comme l'exemple 1 le montre, donne une meilleure lisibilité au code du cycle préprogrammé. Sans lui, il est moins évident que les lignes entre N120 et N200 appartiennent au cycle G81.

Si vous utilisez G80 et que vous ne placez pas un code de mouvement modal juste derrière vous pourrez avoir un de ces messages :

```
Cannot use axis commands with G80
Coordinate setting given with G80
```

Ils servent à vous rappeler que vous devez écrire un nouveau mot de mouvement.

21.3 Cycle G81

Le cycle G81 est destiné au perçage.

0. Un mouvement préliminaire, comme il a été traité ci-dessus.

1. Un déplacement de l'axe Z seul à la vitesse programmée, vers la position Z programmée.

2. Retrait de l'axe Z en vitesse rapide jusqu'au plan de retrait R.

Exemple 1 : G81 en position absolue

Supposons que la position courante soit, X1, Y2, Z3 dans le plan XY, la ligne de codes suivante est interprétée :

G90 G81 G98 X4 Y5 Z1.5 R2.8

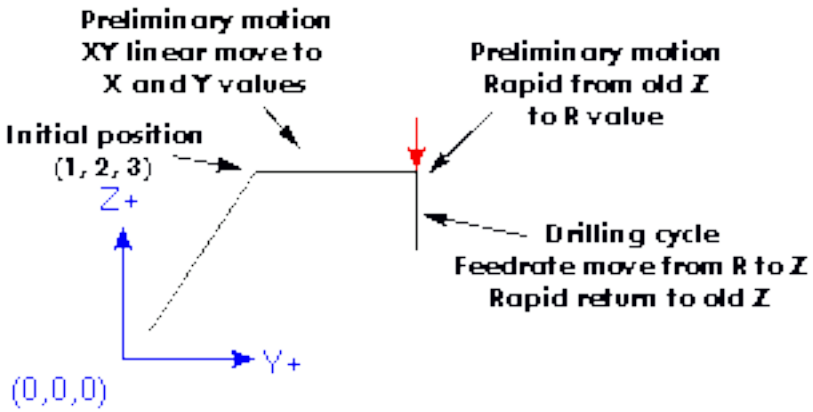
Le mode de déplacements absolus est appelé (G90), le plan de retrait est positionné sur OLD_Z (G98), l'appel du cycle de perçage G81 va lancer ce cycle une fois. La position X deviendra celle demandée, X4. La position de Y deviendra celle demandée, Y5. La position de Z deviendra celle demandée, Z1.5. La valeur de R fixe le plan de retrait de Z à 2.8. La valeur de OLD_Z est 3. Les mouvements suivants vont se produire :

1. Un mouvement en vitesse rapide, parallèle au plan XY vers X4, Y5, Z3

2. Un mouvement en vitesse rapide, parallèle à l'axe Z vers X4, Y5, Z2.8

3. Un mouvement en vitesse travail, parallèle à l'axe Z vers X4, Y5, Z1.5

4. Un mouvement en vitesse rapide, parallèle à l'axe Z vers X4, Y5, Z3



Exemple 2 : Supposons que la position courante soit, X1, Y2, Z3 dans le plan XY, la ligne de codes suivante est interprétée :

G91 G81 G98 X4 Y5 Z-0.6 R1.8 L3

Le mode de déplacements incrémentaux est appelé (G91), le plan de retrait est positionné sur OLD_Z (G98), l'appel du cycle de perçage G81 demande 3 répétitions du cycle. La valeur demandée de X est 4, la valeur demandée de Y est 5, la valeur demandée de Z est -0.6 et le retrait R est à 1.8. La position initiale de X sera 5 (1+4), la position initiale de Y sera 7 (2+5), le plan de retrait sera positionné sur 4.8 (1.8+3) et Z positionné sur 4.2 (4.8-0.6). OLD_Z est à 3.

Le premier mouvement en vitesse rapide le long de l'axe Z vers X1, Y2, Z4.8), puisque OLD_Z est inférieur au plan de retrait.

La première répétition produira 3 mouvements.

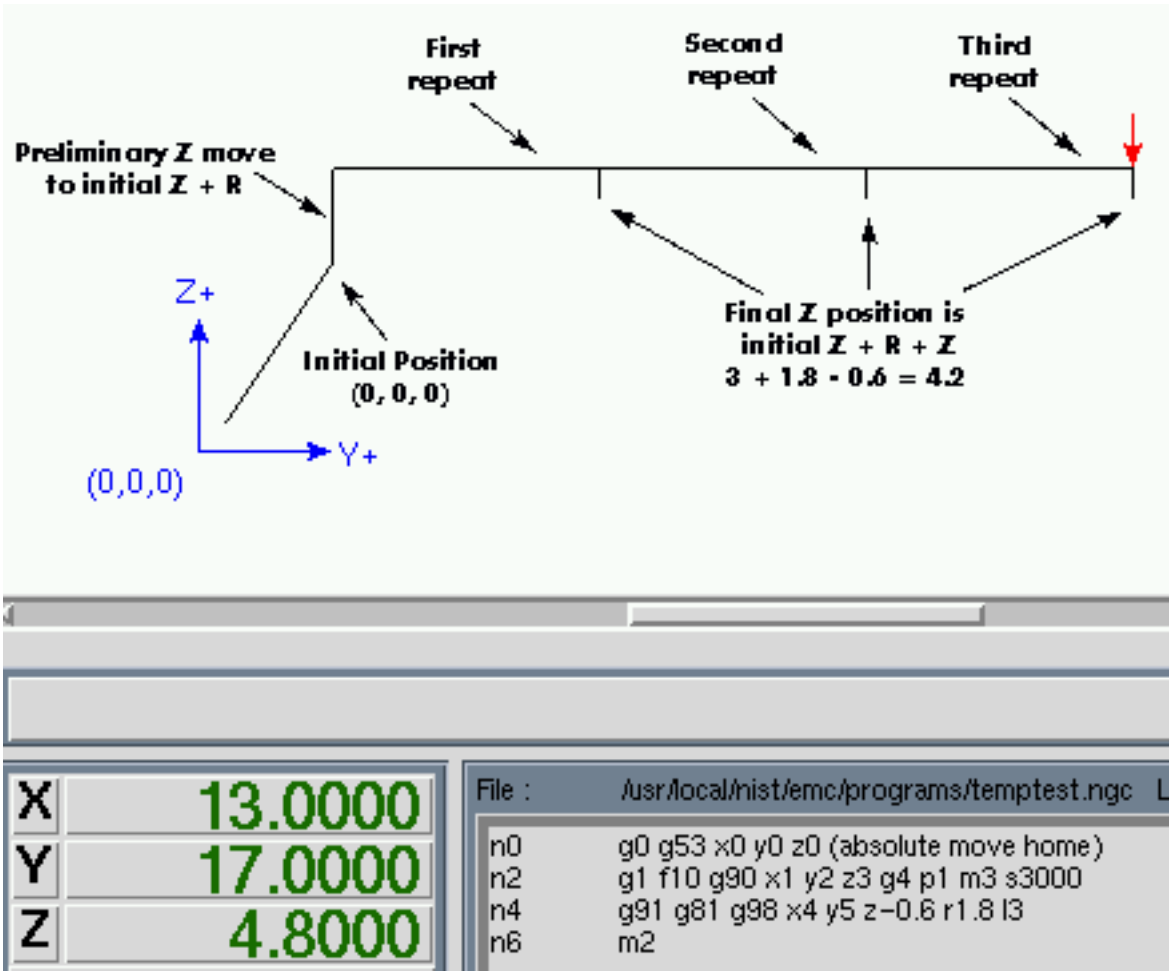
1. Un déplacement en vitesse rapide, parallèle au plan XY vers X5, Y7, Z4.8
2. Un déplacement en vitesse travail, parallèle à l'axe Z vers X5, Y7, Z4.2
3. Un déplacement en vitesse rapide, parallèle à l'axe Z vers X5, Y7, Z4.8

La deuxième répétition produira 3 mouvements. La position de X est augmentée de 4 et passe à 9, la position de Y est augmentée de 5 et passe à 12.

1. Un déplacement en vitesse rapide, parallèle au plan XY vers X9, Y12, Z4.8
2. Un déplacement en vitesse travail, parallèle à l'axe Z vers X9, Y12, Z4.2
3. Un déplacement en vitesse rapide, parallèle à l'axe Z vers X9, Y12, Z4.8

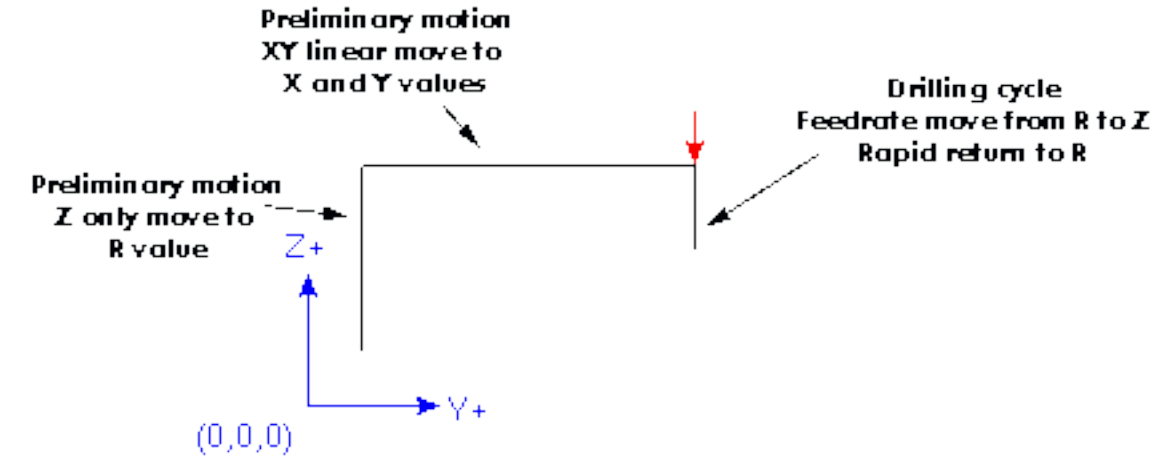
La troisième répétition produira 3 mouvements. La position de X est augmentée de 4 et passe à 13, la position de Y est augmentée de 5 et passe à 17.

1. Un déplacement en vitesse rapide, parallèle au plan XY vers X13, Y17, Z4.8
2. Un déplacement en vitesse travail, parallèle à l'axe Z vers X13, Y17, Z4.2
3. Un déplacement en vitesse rapide, parallèle à l'axe Z vers X13, Y17, Z4.8



Exemple 3 - G81 en position relative

Supposons maintenant que vous exécutez le premier g81 de la ligne de code, mais de (0, 0, 0) plutôt que de (1, 2, 3). G90 G81 G98 X4 Y5 Z1.5 R2.8 Depuis OLD_Z est inférieur à la valeur de R, il n'ajoute rien au mouvement, mais puisque la valeur initiale de Z est inférieure à la valeur spécifiée dans R, un premier mouvement de Z sera effectué durant le mouvement préliminaire.

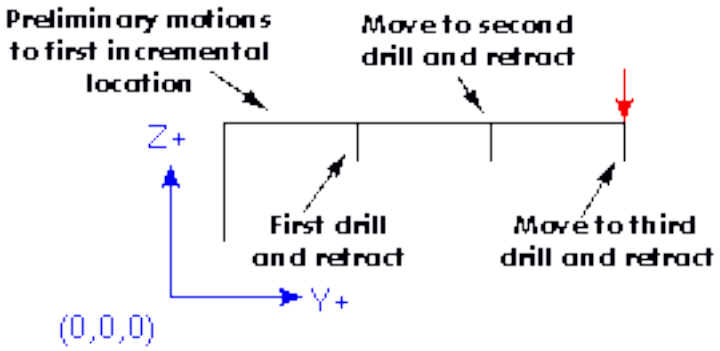


Exemple 4 - G81 en absolu avec R > Z

Il s'agit de la trajectoire pour le second block de code de g81.

G91 G81 G98 X4 Y5 Z-0.6 R1.8 L3

Cette trajectoire commence en (0, 0, 0), l'interpréteur ajoute les valeurs initiales Z0 et R 1.8 et déplace le mobile en vitesse rapide vers cet emplacement. Après ce premier déplacement initial de Z, la répétition fonctionne de manière identique à celle de l'exemple 3 avec le mouvement final de Z à 0.6 en dessous de la valeur de R.



Exemple 5 - Position relative R > Z

21.4 Cycle G82

Le cycle G82 est destiné au perçage.

0. Un mouvement préliminaire, comme il a été traité ci-dessus.

1. Un déplacement de l'axe Z seul en vitesse programmée, vers la position Z programmée.

2. Une temporisation de P secondes.

3. Retrait de l'axe Z en vitesse rapide jusqu'au plan de retrait R.

Les mouvements du cycle G82 ressemblent à ceux de g81 avec une temporisation supplémentaire en fin de mouvement Z. La longueur de cette temporisation, exprimée en secondes, est spécifiée par un mot P# sur la ligne du G82.

G90 G82 G98 X4 Y5 Z1.5 R2.8 P2

Sera équivalent à l'exemple 3 ci-dessus mais avec une temporisation de 2 secondes en fond de trou.

21.5 Cycle G83

Le cycle G83 est destiné au perçage profond ou au fraisage avec brise-copeaux. Les retraits, au cours de ce cycle, dégagent les copeaux du trou et fragmentent les copeaux longs (qui sont fréquents lors du perçage dans l'aluminum). Ce cycle utilise la valeur Q qui représente un incrément "delta" le long de l'axe Z.

0. Un mouvement préliminaire, comme décrit précédemment.

1. Un mouvement de l'axe Z seul, en vitesse travail, sur la position la moins profonde entre, un incrément delta, ou la position de Z programmée.

2. Un mouvement en vitesse rapide au plan de retrait.

3. Une plongée en vitesse rapide dans le même trou, presque jusqu'au fond.

3. Répétition des étapes 2, 3 et 4 jusqu'à ce que la position programmée de Z soit atteinte à l'étape 2.

4. Un mouvement de l'axe Z en vitesse rapide vers le plan de retrait.

21.6 Cycle G84

Ce code n'est pas encore implémenté dans EMC2. Il est accepté mais son comportement n'est pas défini. Voir G33.1

The G84 cycle is intended for right-hand tapping.

0. Preliminary motion, as described above.

1. Start speed-feed synchronization.

2. Move the Z-axis only at the current feed rate to the Z position.

3. Stop the spindle.

4. Start the spindle counterclockwise.

5. Retract the Z-axis at the current feed rate to clear Z.

6. If speed-feed synch was not on before the cycle started, stop it.

7. Stop the spindle.

8. Start the spindle clockwise.

21.7 Cycle G85

Le cycle G85 est destiné à l'alésage, mais peut être utilisé pour le perçage ou le fraisage.

0. Un mouvement préliminaire, comme décrit précédemment.

1. Un déplacement de l'axe Z seul en vitesse travail, vers la position Z programmée.

2. Retrait de l'axe Z en vitesse travail vers le plan de retrait.

21.8 Cycle G86

Le cycle G86 est destiné à l'alésage. Ce cycle utilise la valeur P pour une temporisation en secondes.

0. Un mouvement préliminaire, comme décrit précédemment.

1. Un déplacement de l'axe Z seul en vitesse travail, vers la position Z programmée.

2. Une temporisation de P secondes.

3. L'arrêt de rotation de la broche.

4. Retrait de l'axe Z en vitesse rapide vers le plan de retrait.

5. Reprise de la rotation de la broche dans la même direction que précédemment.

21.9 Cycle G87

Ce code n'est pas encore implémenté dans EMC2. Il est accepté mais son comportement n'est pas défini.

The G87 cycle is intended for back boring.

The situation is that you have a through hole and you want to counter bore the bottom of hole. To do this you put an L-shaped tool in the spindle with a cutting surface on the UPPER side of its base. You stick it carefully through the hole when it is not spinning and is oriented so it fits through the hole, then you move it so the stem of the L is on the axis of the hole, start the spindle, and feed the tool upward to make the counter bore. Then you stop the tool, get it out of the hole, and restart it.

This cycle uses I and J values to indicate the position for inserting and removing the tool. I and J will always be increments from the X position and the Y position, regardless of the distance mode setting. This cycle also uses a K value to specify the position along the Z-axis of the top of counterbore. The K value is an absolute Z-value in absolute distance mode, and an increment (from the Z position) in incremental distance mode.

0. Preliminary motion, as described above.

1. Move at traverse rate parallel to the XY-plane to the point indicated by I and J.

2. Stop the spindle in a specific orientation.

3. Move the Z-axis only at traverse rate downward to the Z position.

4. Move at traverse rate parallel to the XY-plane to the X,Y location.

5. Start the spindle in the direction it was going before.

6. Move the Z-axis only at the given feed rate upward to the position indicated by K.

7. Move the Z-axis only at the given feed rate back down to the Z position.

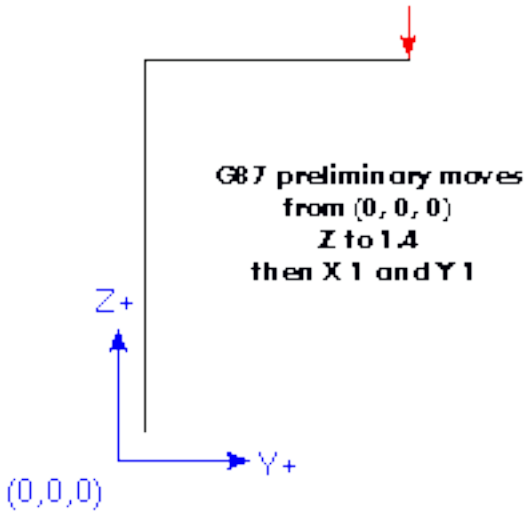
8. Stop the spindle in the same orientation as before.

9. Move at traverse rate parallel to the XY-plane to the point indicated by I and J.

10. Move the Z-axis only at traverse rate to the clear Z.

11. Move at traverse rate parallel to the XY-plane to the specified X,Y location.

12. Restart the spindle in the direction it was going before.

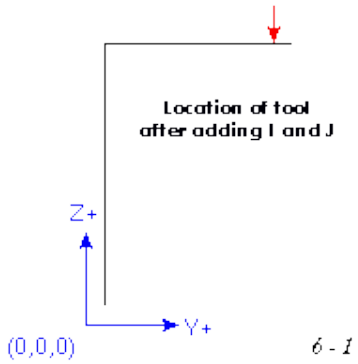


Exemple 6 - Backbore

Example six uses a incremental distances from (0, 0, 0) so the preliminary moves look much like those in example five but they are done using the G87 backbore canned cycle.

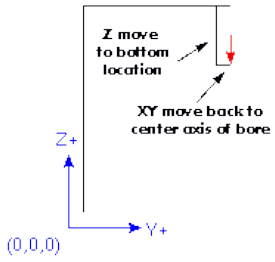
G91 G87 M3 S1000 X1 Y1 Z-0.4 R1.4 I-0.1 J-0.1 K-0.1

You will notice that the preliminary moves shift the tool to directly above the center axis of the existing bore.
Next it increments that location by the I and J values. I offsets X with a plus value being added to the current X. J does the same for the Y axis.
For our example block both I and J are negative so they move back from the hole axis along the path just made by the tool. The amount of offset required should be just enough that the tool tip will slide down through the bore.



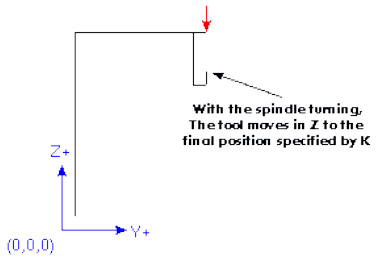
6 - 1

Next the canned cycle moves the tool down in z and at the bottom location represented in the block by the Z 0.4 value it moves the tool back to the center of the bore.



Now the g87 canned cycle turns the spindle on and moves back up into the bore at the programmed feed rate. This is the real cutting action of this canned cycle. With the proper tool in a boring bar this cycle will produce a chamfer on the bottom side of the bore. G87 can also be used to produce a larger diameter bore on the bottom side of the bore.

When the tool has reached the K position it is returned to the bottom location, the spindle is stopped and oriented and follows the earlier path back out of the bore to the initial position above.



This canned cycle assumes spindle orientation which has not been implemented in the EMC to date. The proper alignment of the tool tip to the oriented spindle is critical to the successful insertion of the tool through the hole to be backbored.

21.10 Cycle G88

Ce code n'est pas encore implémenté dans EMC2. Il est accepté mais son comportement n'est pas défini.

The G88 cycle is intended for boring. This cycle uses a P value, where P specifies the number of seconds to dwell.

0. Preliminary motion, as described above.

1. Move the Z-axis only at the current feed rate to the Z position.

2. Dwell for the given number of seconds.

3. Stop the spindle turning.

4. Stop the program so the operator can retract the spindle manually.

5. Restart the spindle in the direction it was going. It is unclear how the operator is to manually move the tool because a change to manual mode resets the program to the top. We will attempt to clarify that step in this procedure.

21.11 Cycle G89

Le cycle G89 est destiné à l'alésage. Il utilise la valeur de P pour une temporisation en secondes.

0. Un mouvement préliminaire, comme décrit précédemment.

1. Un déplacement de l'axe Z seul en vitesse travail, vers la position Z programmée.

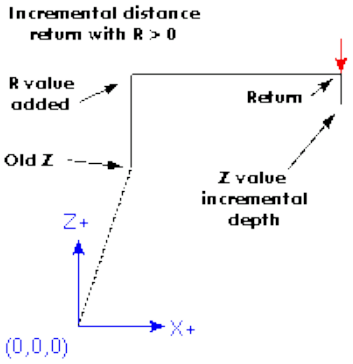
2. Temporisation de P secondes.

3. Retrait de l'axe Z en vitesse travail vers le plan de retrait.

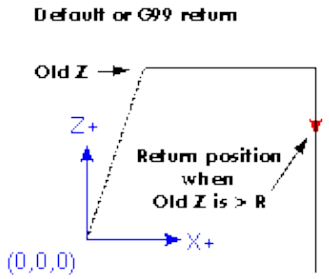
21.12 Options G98 et G99

Quand la broche se rétracte pendant les cycles préprogrammés, il existe deux options pour indiquer comment elle se rétracte : (1) Retrait perpendiculaire au plan de travail courant jusqu'à la position indiquée par le mot R, ou (2) Retrait perpendiculaire au plan de travail courant jusqu'à la position qui était celle de cet axe juste avant le début du cycle préprogrammé (à moins que cette position ne soit inférieure à celle indiquée par le mot R, auquel cas, c'est cette dernière qui serait utilisée.

Pour utiliser l'option (1), programmer G99. Pour utiliser l'option (2), programmer G98. Ne pas oublier que le mot R a différentes significations en mode de déplacement absolu et en mode de déplacement incrémental.



Utilisation de G98



Utilisation de G99

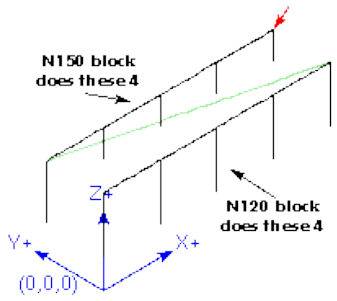
21.13 Pourquoi utiliser les cycles préprogrammés ?

Il y a au moins deux raisons, la première est l'économie de code. Un simple trou demande plusieurs lignes de code pour être exécuté.

Exemple 7 : Nous avons montré plus haut, comment les cycles préprogrammés peuvent être utilisés pour produire 8 trous avec dix lignes de code. Le programme ci-dessous permet de produire le même jeu de 8 trous en utilisant cinq lignes pour le cycle préprogrammé. Il ne suit pas exactement le même parcours et ne perce pas dans le même ordre que l'exemple précédent, mais le programme a été écrit de manière économique, une bonne pratique qui devrait être courante avec les cycles préprogrammés.

Exemple 7 - perçage de huit trous, réécrit.

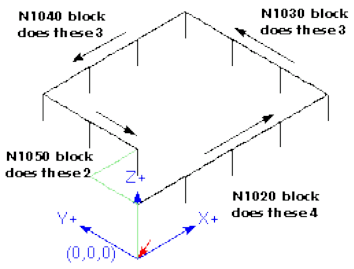
```
n100 g90 g0 x0 y0 z0 (coordonnées d'origine)

n110 g1 f10 x0 g4 p0.1

n120 g91 g81 x1 y0 z-1 r1 l4(cycle de perçage)

n130 g90 g0 x0 y1

n140 z0

n150 g91 g81 x1 y0 z-.5 r1 l4(cycle de perçage)

n160 g80 (révocation du cycle G81)

n170 m2 (fin du programme)
```



Exemple 8 - Douze trous dans un carré

Cet exemple montre l'utilisation du mot L pour répéter une série incrémentale de cycles de perçage pour des blocks de code successifs dans le même mode mouvements G81. Ici, nous produisons 12 trous au moyen de cinq lignes de code dans le mouvement modal.

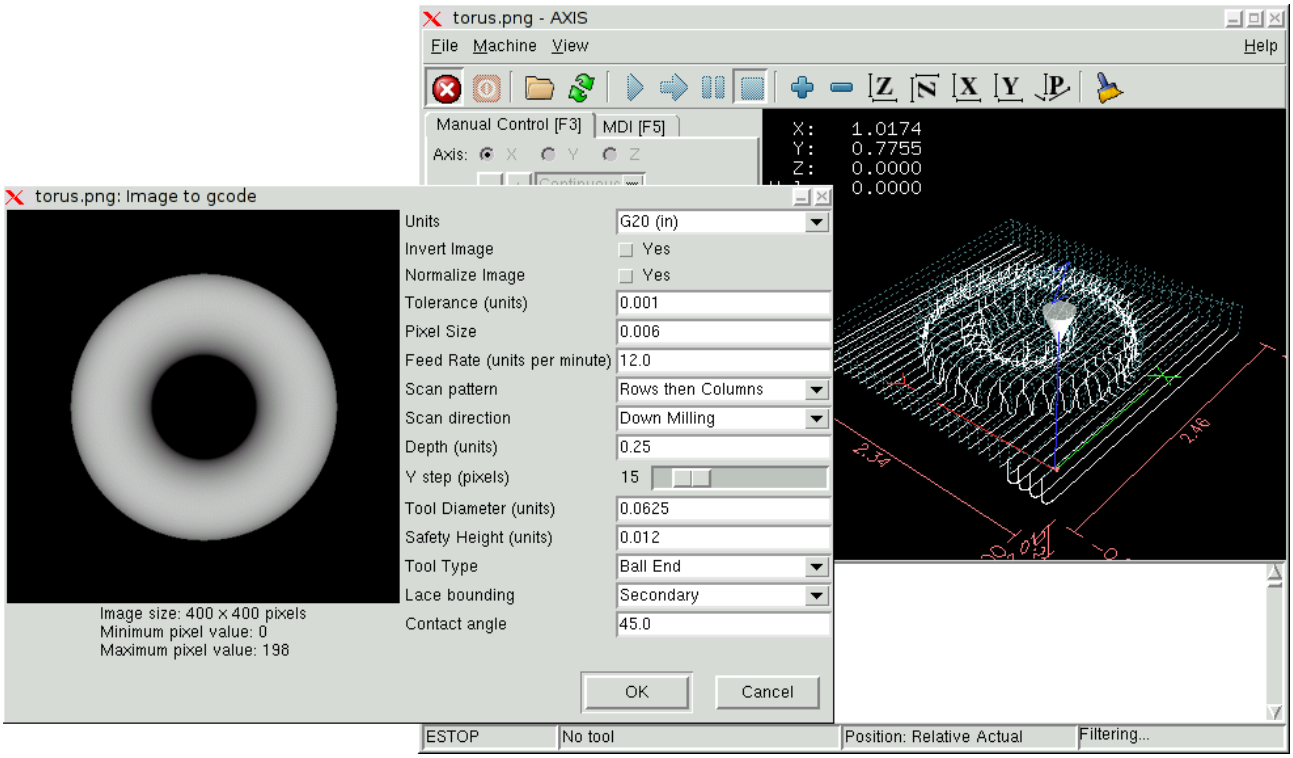
```
N1000 G90 G0 X0 Y0 Z0 (coordinées d'origine)
N1010 G1 F50 X0 G4 P0.1
N1020 G91 G81 X1 Y0 Z-0.5 R1 L4 (cycle de
perçage)
N1030 X0 Y1 R0 L3 (répétition)
N1040 X-1 Y0 L3 (répétition)
N1050 X0 Y-1 L2 (répétition)
N1060 G80 (révocation du cycle G81)
N1070 G90 G0 X0 (retour vers l'origine en
vitesse rapide)
N1080 Y0
N1090 Z0
N1100 M2 (fin du programme)
```



La deuxième raison d'utiliser les cycles préprogrammés, c'est qu'il produisent un mouvement préliminaire et retournent à une position prévisible et contrôlable, quel que soit le point de départ du cycle.

Chapitre 22

Image-to-gcode : Usiner un “depth maps”



22.1 Qu’est-ce qu’un “depth map” ?

Il s’agit d’une image en échelle de gris dont la luminosité de chaque pixel corresponds à la profondeur (ou hauteur) de chaque point de l’objet.

22.2 Intégrer image-to-gcode dans l’interface utilisateur d’AXIS

Ajoutez les lignes suivantes dans la section : `[FILTER]` de votre fichier .ini pour qu’AXIS invoque automatiquement image-to-gcode à l’ouverture d’une image .png, .gif, ou .jpg :

```
PROGRAM_EXTENSION = .png,.gif,.jpg Grayscale Depth Image
png = image-to-gcode
gif = image-to-gcode
jpg = image-to-gcode
```

Le fichier de configuration : `sim/axis.ini` est déjà configuré de cette façon.

22.3 Utilisation d’image-to-gcode

image-to-gcode peut être démarré soit en ouvrant une image dans AXIS, soit en invoquant image-to-gcode dans une console, de la manière suivante :

```
image-to-gcode torus.png > torus.ngc
```

Ajustez les réglages dans la colonne de droite, puis pressez OK pour créer le g-code. Selon la taille de l’image et les options choisies, le traitement peut durer de quelques secondes à quelques minutes. Quand une image est appelée, le gcode sera automatiquement chargé et visualisé dans AXIS une fois le traitement terminé. Dans AXIS, faites ”Recharger” pour afficher de nouveau l’écran d’options d’ image-to-gcode, vous pourrez ainsi travailler en boucle.

22.4 Les différentes options

22.4.1 Unités

Spécifie quelle unité sera utilisée dans le g-code généré G20 (pouces) ou G21 (mm), ce sera également l’unité utilisée par toutes les options marquées : **(units)**.

22.4.2 Invert Image

Si “no”, le pixel noir sera le point le plus bas et le pixel blanc le point le plus haut. Si “yes”, le pixel noir sera le point le plus haut et le pixel blanc le point le plus bas.

22.4.3 Normalize Image

Si “yes”, le pixel le plus sombre est ramené au noir, le pixel le plus lumineux est ramené au blanc.

22.4.4 Expand Image Border

Si “None”, l’image entrée sera utilisée telle-quelle, les détails les plus aux bords de l’image pourraient être coupés. Si “White” ou “Black”, alors une bordure de pixels égale au diamètre de l’outil sera ajoutée sur tout le pourtour pour éviter ce risque.

22.4.5 Tolerance (unités)

Quand une série de points est proche d’une ligne droite au point d’être dans la **tolerance**, elle sera traitée comme une ligne droite en sortie. Augmenter la tolérance peut donner de meilleures performances de contourage avec emc, mais peut aussi estomper ou gommer les détails les plus fins de l’image.

22.4.6 Pixel Size (unités)

Il y a beaucoup d’unités pour un pixel dans l’image entrée. Habituellement ce nombre est beaucoup plus petit que 1.0. Par exemple, pour usiner un objet de 50x50mm depuis une image de 400x400 pixels, utiliser un ”pixel size” de 0.125, parce que 50 / 400 = 0.125.

22.4.7 Plunge Feed Rate (unités par minute)

Vitesse du mouvement de plongée initial.

22.4.8 Feed Rate (unités par minute)

Vitesse d’avance pour le reste de l’usinage.

22.4.9 Spindle Speed (RPM)

Vitesse de rotation de la broche, en tours/mn

22.4.10 Scan Pattern

Modèles de balayage possibles :

– Rangées
– Colonnes
– Rangées puis colonnes
– Colonnes puis rangées

22.4.11 Scan Direction

Directions de balayage possibles :

– Positive : le fraisage commence à de petites valeurs de X ou Y et se poursuit avec des valeurs croissantes.
– Négative : le fraisage commence à des valeurs élevées de X ou Y et se poursuit avec des valeurs décroissantes.
– Alternative : le fraisage commence aux valeurs de X ou Y où s’est terminé le dernier mouvement. Cela réduit les déplacements ”en l’air”.
– Up Milling : le fraisage commence en points bas et se poursuit vers les points hauts.
– Down Milling : le fraisage commence en points hauts et se poursuit vers les points bas.

22.4.12 Depth (unités)

Le dessus du bloc est toujours à **Z=0**. La profondeur d’usinage dans le matériau est de **Z=-depth.**

22.4.13 Step Over (pixels)

Distance entre rangées ou colonnes adjacentes. Pour trouver le nombre en pixels pour une distance donnée en unités, calculez : **distance/pixel size** et arrondissez au nombre le plus proche**.** Par exemple : si **pixel size=.006** et le pas souhaité sur la **distance=.015**, alors utilisez un Step Over de 2 ou 3 pixels, parce que **.015/.006=2.5.**

22.4.14 Tool Diameter

Le diamètre du taillant de l'outil.

22.4.15 Safety Height

La hauteur à laquelle les mouvements de traversée. image-to-gcode considère toujours le dessus du matériau comme étant : **Z=0**.

22.4.16 Tool Type

La forme du taillant de l'outil. Les formes possibles sont :

- Hémisphèrique
- Plate
- Vé à 45 degrés
- Vé à 60 degrés

22.4.17 Lace bounding

Contrôle si les zones relativement plates le long d'une colonne ou d'une rangée peuvent être ignorées. Ces options n'ont de sens que pour un fraisage dans les deux directions. Trois choix sont possibles :

- None : toutes les rangées et les colonnes seront entièrement fraisées.
- Secondary : lors du fraisage dans la deuxième direction, les zones qui ne présentent pas une forte pente dans cette direction seront ignorées.
- Full : lors du fraisage dans la première direction, les zones qui présentent une forte pente dans la deuxième direction seront ignorées. Lors du fraisage dans la deuxième direction, les zones qui ne présentent pas une forte pente dans cette direction seront ignorées.

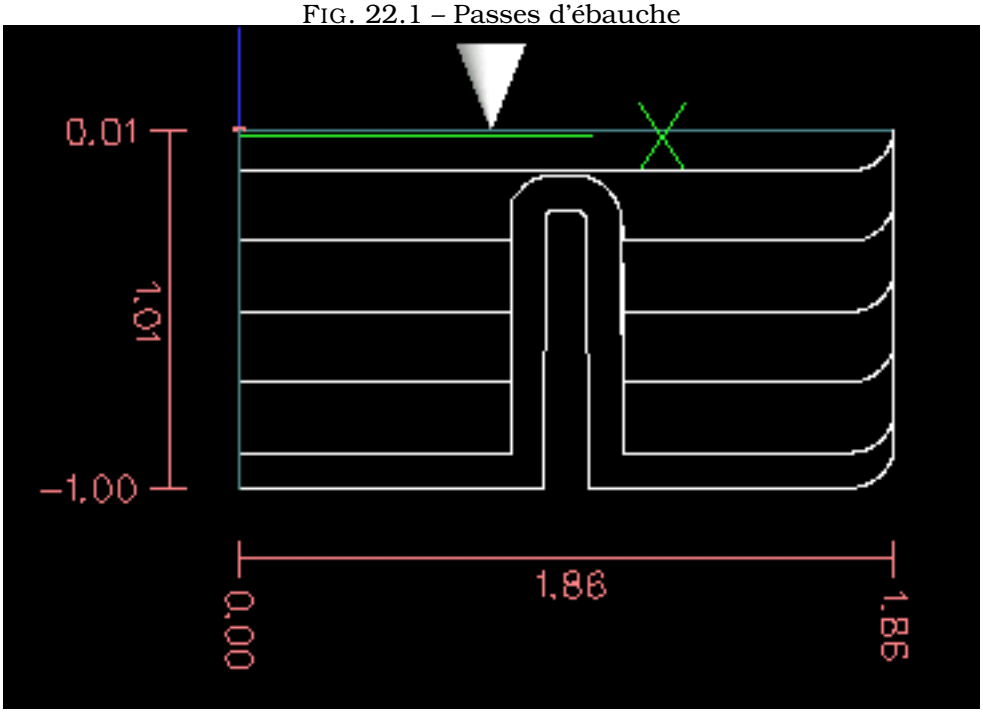
22.4.18 Contact angle

Quand **Lace bounding** n'est pas None, les pentes qui présentent une pente supérieure à **Contact angle** seront considérées comme de “fortes” pentes et celles en dessous de cet angle considérées comme de faible pentes.

22.4.19 Roughing offset and depth per pass

Image-to-gcode peut optionnellement produire des passes d'ébauche. La profondeur des passes d'ébauche successives est fixée par “Roughing depth per pass”. Par exemple, entrer 0.2 pour une première passe d'ébauche d'une profondeur de 0.2, la seconde passe d'ébauche aura une profondeur de 0.4 et ainsi de suite, jusqu'à-ce que la profondeur totale Depth de l'image soit atteinte. Aucune des passes d'ébauche n'usinera plus près de la partie finale que Roughing Offset. La figure 22.1 montre une grande profondeur verticale à usiner. Sur cette image, la profondeur des passes d'ébauche est de 0.2 pouces et Roughing Offset de 0.1 pouces.

FIG. 22.1 – Passes d’ébauche



Chapitre 23

Section légale

Chapter 24

Legal Section

Copyright Terms

GNU Free Documentation License

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.

Index