LinuxCNC V2.9.0~pre0, 06 Apr 2022

Contents

Part I

Contents

Part II

About LinuxCNC

Chapter 1

Introduction

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

Chapter 2

LinuxCNC History

== Origin

EMC (the Enhanced Machine Controller) was created by NIST , the National Institute of Standards and Technology, which is an agency of the Commerce Department of the United States government.

NIST first became interested in writing a motion control package as a test platform for concepts and standards. Early sponsorship from General Motors resulted in an adaptation of the fledgling version of EMC using PMAC intelligent control boards running under a ”real time” version of Windows NT and controlling a large milling machine.

As is required of all work product of US federal government employees, the resulting software and the report about it are required to be in the public domain and a report about it was duly published, including on the Internet. It was there that Matt Shaver discovered EMC. He contacted NIST and entered into discussions with Fred Proctor about adapting the code for use in controlling less expensive hardware to be used for upgrades and replacements of CNC controls that were obsolete or just plain dead. NIST was intrigued because they too wanted something less expensive. In order to launch a cooperative effort, a formal agreement was created which guaranteed that the resulting code and design would remain in the public domain.

Early considerations focused on replacing the expensive and temperamental ”real time” Windows NT system. It was proposed that a relatively new (at the time) real time extension of the Linux operating system be tried. This idea was pursued with success. Next up was the issue of the expensive intelligent motion control boards. By this time the processing power of a PC was considered great enough to directly take control of the motion routines. A quick search of available hardware resulted in the selection of a Servo-To-Go interface board as the first platform for letting the PC directly control the motors. Software for trajectory planning and PID loop control was added to the existing user interface and RS274 interpreter. Matt successfully used this version to upgrade a couple of machines with dead controls and this became the EMC system that first caught the attention of the outside world. Mention of EMC on the rec.crafts.metalworking USENET newsgroup resulted in early adopters like Jon Elson building systems to take advantage of EMC.

NIST set up a mailing list for people interested in EMC. As time went on, others outside NIST became interested in improving EMC. Many people requested or coded small improvements to the code. Ray Henry wanted to refine the user interface. Since Ray was reluctant to try tampering with the C code in which the user interface was written, a simpler method was sought. Fred Proctor of NIST suggested a scripting language and wrote code to interface the Tcl/Tk scripting language to the internal NML communications of EMC. With this tool Ray went on to write a Tcl/Tk program that became the predominant user interface for EMC at the time.

For NIST’s perspective, see this paper written by William Shackleford and Frederick Proctor, describing the history of EMC and its transition to open source.

By this time interest in EMC as beginning to pick up substantially. As more and more people attempted installation of EMC, the difficulty of patching a Linux kernel with the real time extensions and of

compiling the EMC code became glaringly obvious. Many attempts to document the process and write scripts were attempted, some with moderate success. The problem of matching the correct version of the patches and compilers with the selected version of Linux kept cropping up. Paul Corner came to the rescue with the BDI (brain dead install) which was a CD from which a complete working system (Linux, patches, and EMC) could be installed. The BDI approach opened the world of EMC to a much larger user community. As this community continued to grow, the EMC mailing list and code archives were moved to SourceForge and the LinuxCNC web site was established.

With a larger community of users participating, EMC became a major focus of interest at the on-going CNC exhibits at NAMES and NAMES became the annual meeting event for EMC. For the first couple of years, the meetings just happened because the interested parties were at NAMES. In 2003 the EMC user community had its first announced public meeting. It was held the Monday after NAMES in the lobby of the arena where the NAMES show was held. Organization was loose, but the idea of a hardware abstraction layer (HAL) was born and the movement to restructure the code for ease of development (EMC2) was proposed.

2.1 Name Change

In the spring of 2011, the LinuxCNC Board of Directors was contacted by a law firm representing EMC Corporation (www.emc.com) about the use of ”EMC” and ”EMC2” to identify the software offered on linuxcnc.org. EMC Corporation has registered various trademarks relating to EMC and EMC² (EMC with superscripted numeral two).

After a number of conversations with the representative of EMC Corporation, the final result is that, starting with the next major release of the software, linuxcnc.org will stop identifying the software using ”emc” or ”EMC”, or those terms followed by digits. To the extent that the LinuxCNC Board of Directors controls the names used to identify the software offered on linuxcnc.org, the board has agreed to this.

As a result, it was necessary to choose a new name for the software. Of the options the board considered, there was consensus that ”LinuxCNC” is the best option, as this has been our website’s name for years.

In preparation for the new name, we have received a sub-license of the LINUX® trademark from the Linux Foundation (www.linuxfoundation.org), protecting our use of the LinuxCNC name. (LINUX® is the registered trademark of Linus Torvalds in the U.S. and other countries.)

The rebranding effort included the linuxcnc.org website, the IRC channels, and versions of the software and documentation since version 2.5.0.

2.2 Additional Info

NIST published a paper describing the RS274NGC language and the abstract machining center it controls, as well as an early implementation of EMC. The paper is also available at http://linuxcnc.org/files/RS274NGCv3.pdf .

NIST also published a paper on the history of EMC and its transition to open source. The paper is also available at http://linuxcnc.org/files/Use-of-Open-Source-Distribution-for-a-Machine-Tool-Controller.pdf

Part III

Using LinuxCNC

Chapter 3

General Info

3.1 Avant-propos

LinuxCNC est souple et modulaire. Ces attributs l'ont fait apparaître à certains comme un brouillon de petits morceaux confus, ils se sont demandé pourquoi il en était ainsi. Cette page tentent de répondre à cette question avant que vous lecteurs, ne plongiez dedans pour vous faire votre propre idée.

EMC a débuté à l'institut national des standards et des technologies des Etats Unis, le NIST. Il a mûri comme un logiciel fonctionnant sur le système d'exploitation Unix. Unix le rendait différent. Très tôt des développeurs Unix ont apporté une série d'idées concernant l'écriture du code, c'est devenu une écriture selon «la tradition d'Unix». Les premiers auteurs de LinuxCNC ont suivi cette voie.

Eric S. Raymond, dans son livre The Art of Unix Programming, résume la philosophie Unix par la philosophie largement utilisée en ingéniérie, le principe KISS Keep it Simple, Stupid Reste Simple, Crétin ou Sois Simple et Concis. Puis il décrit sa vision selon laquelle cette philosophie globale s'applique en tant que norme culturelle Unix, bien qu'on trouve sans surprise de graves entorses à la plupart des règles Unix suivantes:

- Règle de modularité: Ecrire des éléments simples reliés par de bonnes interfaces.
- Règle de clarté: La Clarté vaut mieux que l'ingéniosité.
- Règle de composition: Concevoir des programmes qui peuvent être reliés à d'autres programmes.
- Règle de séparation: Séparer les règles du fonctionnement; Séparer les interfaces du mécanisme.[1]

Monsieur Raymond offre d'autres règles mais ces quatre là décrivent les caractéristiques essentielles du système de contrôle de mouvement LinuxCNC.

La règle de Modularité est critique. Tout au long de ces manuels, vous trouverez des discussions à propos de l'interpréteur ou à propos des planificateurs de tâche ou de mouvement ou encore à propos de HAL. Chacun d'eux est un module ou un ensemble de modules. Cette modularité vous permettra de ne connecter entre elles que les parties dont vous avez besoin pour le bon fonctionnement de votre machine.

La règle de clarté est essentielle. LinuxCNC est en perpétuelle évolution, il n'est pas terminé et ne le sera jamais. Il est assez complet pour piloter toutes les machines que nous avons voulu qu'il pilote. Une bonne partie de cette évolution est atteinte parce que les utilisateurs et les développeurs peuvent voir le travail des autres et construire sur ce qui est déjà fait.

La règle de composition nous permet de concevoir et de construire un contrôleur à partir des nombreux modules existants, en les rendant connectables entre eux. Nous obtenons cette connectivité en appliquant une interface standard à tous les modules et en suivant ce standard.

[1] Trouvé sur http://fr.wikipedia.org/wiki/Philosophie_d%27Unix, 09/09/2008

La règle de séparation exige que chaque petite chose soit faite par une partie distincte. En séparant les fonctions, le dépannage est rendu plus aisé, le remplacement de modules par d'autres peut être fait à l'intérieur du système et la comparaison s'effectuer facilement.

Qu'apporte la fameuse «tradition d'Unix» à vous, utilisateurs de LinuxCNC. Elle signifie que vous pourrez faire des choix sur la façon d'utiliser le système. Beaucoup de ces choix affecteront les parties intégrées à la machine, mais beaucoup également affecteront la manière dont vous utiliserez votre machine. Au cours de votre lecture, vous trouverez différents endroits où vous pourrez faire des comparaisons. Finalement vous pourrez dire «J'utiliserai cette interface plutôt que telle autre» ou, «J'écrirai cette nouvelle partie de telle manière plutôt que de telle autre.» Tout au long de ces manuels nous décrirons l'étendue des possibilités de LinuxCNC actuellement disponibles.

Puisque vous commencez votre voyage dans l'utilisation de LinuxCNC nous vous proposons ces deux citations de précaution.

- Pour paraphraser les paroles de Doug Gwyn sur UNIX: "LinuxCNC n'a pas été conçu pour empêcher ses utilisateurs de commettre des actes stupides, car cela les empêcherait aussi de réaliser des actes ingénieux."
- De même les paroles de Steven King: "LinuxCNC est convivial. Cependant Unix ne précise pas vraiment avec qui."

3.2 LinuxCNC

3.2.1 Introduction

Ce document est centré sur l'utilisation de LinuxCNC, il est plutôt destiné aux lecteurs l'ayant déjà installé et configuré. Quelques informations sur l'installation sont données dans les chapitres suivants. La documentation complète sur l'installation et la configuration se trouve dans le manuel de l'intégrateur.

3.2.2 Comment fonctionne LinuxCNC

LinuxCNC est un peu plus que juste un autre programme de fraiseuse CNC. Il est capable de contrôler des machines-outils, des robots ou d'autres automatismes. Il est capable de contrôler des servomoteurs, des moteurs pas à pas, des relais ainsi que d'autres mécanismes relatifs aux machines-outils.

Il y a quatre principales composantes du logiciel LinuxCNC:

- un contrôleur de mouvement (EMCMOT),
- une contrôleur d'entrées/sorties discrètes (EMCIO),
- un exécuteur des tâches qui les coordonne (EMCTASK),
- et les interfaces utilisateur graphiques.

En outre il existe une couche appelée HAL (couche d'abstraction du matériel) qui permet la configuration de LinuxCNC sans avoir besoin de recompiler.

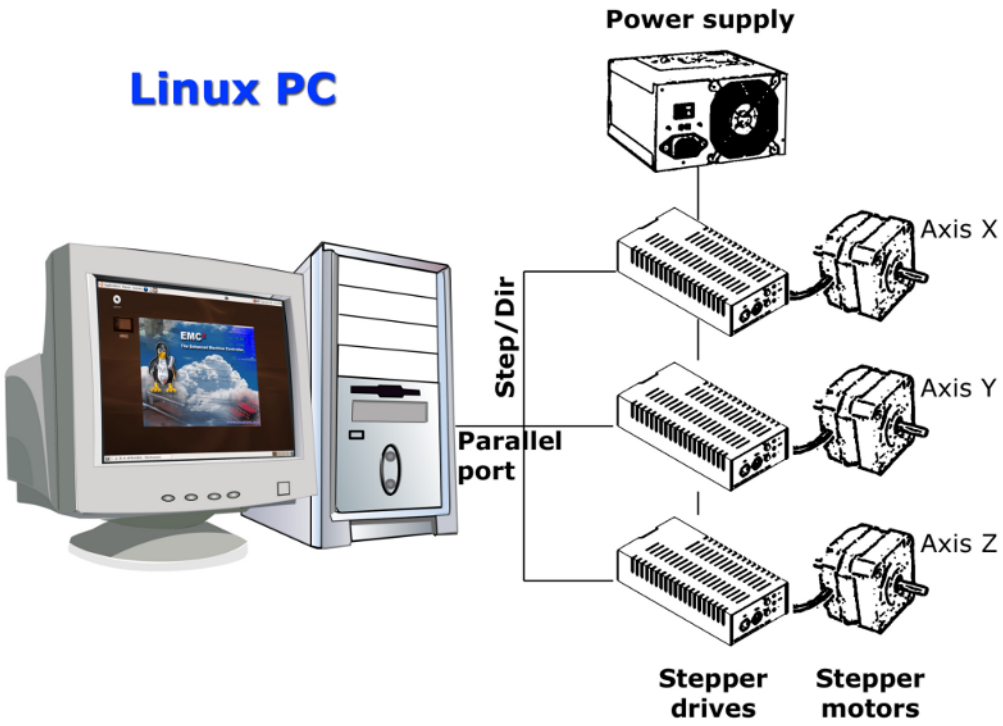


Figure 3.1: Machine simple contrôlée par LinuxCNC

La figure ci-dessus montre un diagramme bloc représentant une machine 3 axes typique comme LinuxCNC les aime. Cette figure montre un système basé sur des moteurs pas à pas. Le PC, tournant sous Linux contrôle les interfaces de puissance des moteurs pas à pas en leur envoyant des signaux au travers du port parallèle. Ces signaux (impulsionnels) font que la puissance adéquate est fournie aux moteurs. LinuxCNC peut également contrôler des servomoteurs via une interface de puissance pour servomoteurs ou utiliser le port parallèle étendu connecté à une carte de contrôle externe. Quand nous examinerons chacun des composants qui forment un système LinuxCNC, nous nous référerons à cette machine typique.

3.2.3 Interfaces utilisateur graphiques

L'interface graphique est la partie de LinuxCNC qui interagit avec l'opérateur de la machine. LinuxCNC est fourni avec plusieurs interfaces utilisateurs graphiques:

- Axis, l'interface utilisateur standard.

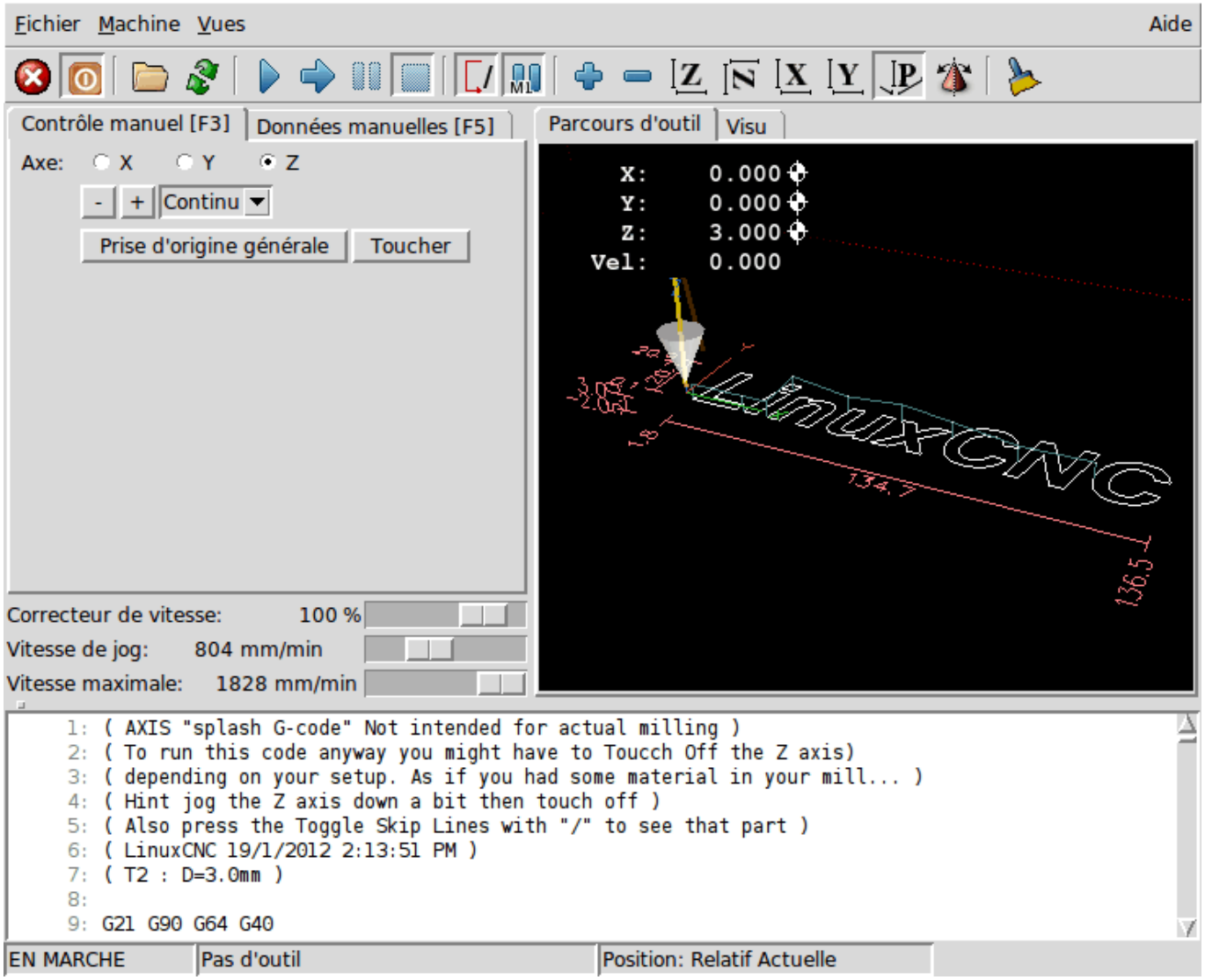


Figure 3.2: L'interface graphique AXIS

- Touchy, une interface graphique pour écran tactile.

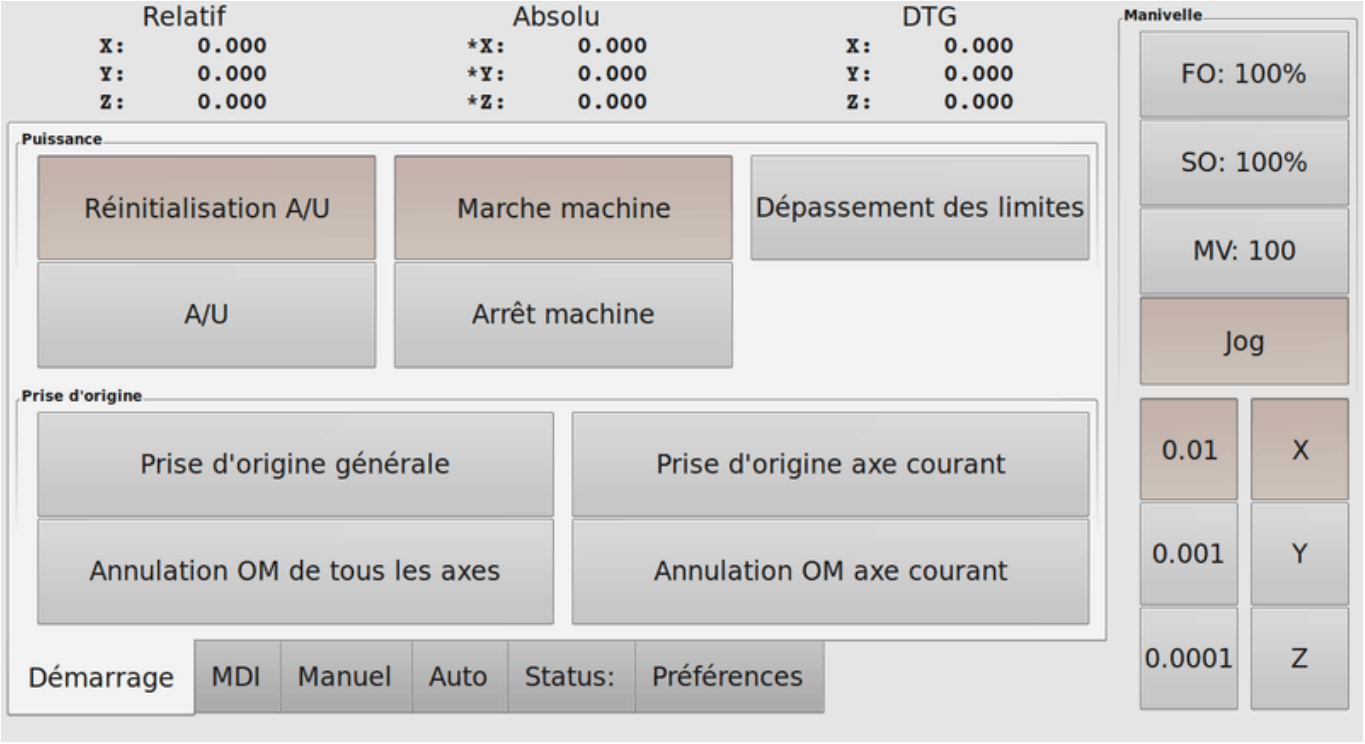


Figure 3.3: L'interface graphique Touchy

- NGCGUI, une interface graphique gérant les sous-programmes. Elle permet très simplement de créer des programme G-code. Elle supporte surtout la concaténation de fichiers de sous-programmes, ce qui permet de construire des programmes G-code complets sans aucune programmation.

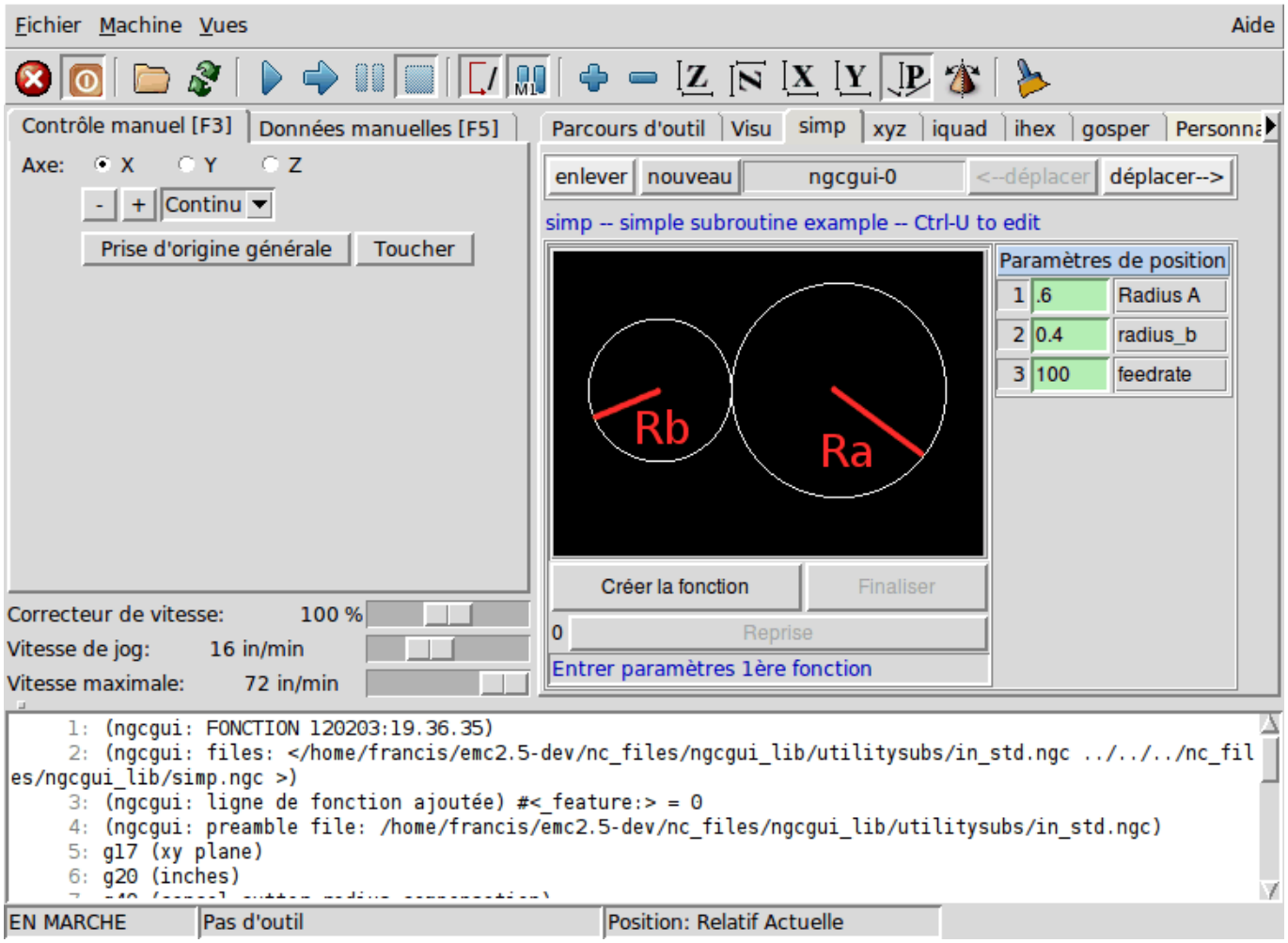


Figure 3.4: L'interface graphique NGCGUI intégrée dans Axis

- TkLinuxCNC, une autre interface basée sur Tcl/Tk. C'est l'interface la plus populaire après Axis

Figure 3.5: L'interface graphique tklinuxcnc

- Xemc, un programme X-Windows
- halui, une interface utilisateur basée sur HAL, qui permet de contrôler LinuxCNC en utilisant des boutons et des interrupteurs
- linuxcncrsh, une interface utilisateur basée sur telnet, qui permet d'envoyer des commandes à partir d'ordinateurs distants de celui de LinuxCNC

3.2.4 Panneaux de contrôle virtuels

- PyVCP, un panneau de contrôle virtuel basé sur Python, il peut être intégré dans l'interface graphique Axis ou utilisé en autonome.

- GladeVCP, un panneau de contrôle virtuel basé sur Glade, il peut être intégré dans l'interface graphique Axis ou utilisé en autonome.

3.2.5 Langues

LinuxCNC utilise des fichiers traduits pour les interfaces utilisateur. Il fonctionne dans plusieurs langues et démarre dans la langue de la session ouverte par l'utilisateur au démarrage du PC. Si votre langue n'a pas encore été traduite contactez un développeur sur l'IRC ou sur la mailing liste si vous pouvez aider à la traduction.

3.2.6 Penser comme un opérateur sur CNC

Ce manuel ne prétend pas vous apprendre à utiliser un tour ou une fraiseuse. Devenir un opérateur expérimenté prends beaucoup de temps et demande beaucoup de travail. Un auteur a dit un jour, Nous apprenons par l'expérience, si on la possède toute. Les outils cassés, les étaux attaqués et les cicatrices sont les preuves des leçons apprises. Une belle finition, des tolérances serrées et la prudence pendant le travail sont les preuves des leçons retenues. Aucune machine, aucun programme ne peut remplacer l'expérience humaine.

Maintenant que vous commencez à travailler avec le programme LinuxCNC, vous devez vous placer dans la peau d'un opérateur. Vous devez être dans le rôle de quelqu'un qui a la charge d'une machine. C'est une machine qui attendra vos commandes puis qui exécutera les ordres que vous lui donnerez. Dans ces pages, nous donnerons les explications qui vous aideront à devenir un bon opérateur de CNC avec LinuxCNC. Vous aurez besoin de bonnes informations ici, devant vous, c'est là que les pages suivantes prendront tout leur sens.

3.2.7 Modes opératoires

Quand LinuxCNC fonctionne, il existe trois différents modes majeurs pour entrer des commandes. Les modes Manuel, Auto et MDI. Passer d'un mode à un autre marque une grande différence dans le comportement de LinuxCNC. Des choses spécifiques à un mode ne peuvent pas être faites dans un autre. L'opérateur peut faire une prise d'origine sur un axe en mode manuel mais pas en mode auto ou MDI. L'opérateur peut lancer l'exécution complète d'un programme de G-codes en mode auto mais pas en mode manuel ni en MDI.

En mode manuel, chaque commande est entrée séparément. En termes humains, une commande manuelle pourrait être active l'arrosage ou jog l'axe X à 250 millimètres par minute. C'est en gros, équivalent à basculer un interrupteur ou à tourner la manivelle d'un axe. Ces commandes sont normalement contrôlées en pressant un bouton de l'interface graphique avec la souris ou en maintenant appuyée une touche du clavier. En mode auto, un bouton similaire ou l'appui d'une touche peut être utilisé pour charger ou lancer l'exécution complète d'un programme de G-codes stocké dans un fichier. En mode d'entrée de données manuelles (MDI) l'opérateur peut saisir un bloc de codes est dire à la machine de l'exécuter en pressant la touche Return ou Entrée du clavier.

Certaines commandes de mouvement sont disponibles et produisent les mêmes effets dans tous les modes. Il s'agit des commandes Abandon, Arrêt d'Urgence et Correcteur de vitesse travail . Ces commandes se dispensent d'explications.

L'interface utilisateur graphique AXIS supprime certaines distinctions entre Auto et les autres modes en rendant automatique la disponibilité des commandes, la plupart du temps. Il rend également floue la distinction entre Manuel et MDI parce que certaines commandes manuelles comme Toucher, sont également implémentées en envoyant une commande MDI. Il fait cela en changeant automatiquement le mode qui est nécessaire pour l'action que l'utilisateur a demandé.

3.3 Concepts importants pour l’utilisateur

3.3.1 La configuration machine

Le dessin suivant montre les directions de déplacement de l’outil et la position des fins de course de limite sur une fraiseuse classique. Noter le diagramme cartésien représentant les directions de déplacement de l’outil (Tool Direction). La direction de déplacement de la table et en opposition du système de coordonnées cartésiennes. Le système de coordonnées cartésiennes représente le sens de déplacement de l’outil. C’est toujours les déplacements de l’outil qui doivent être programmés pour que l’outil se déplace dans les directions correctes par rapport au matériel.

Noter également la position des fins de course et le sens d’activation de leurs cames. Plusieurs combinaisons sont possibles, par exemple il est possible, à l’inverse du dessin, de placer un seul fin de course fixe au milieu de la table et deux cames mobiles pour l’actionner. Dans ce cas les limites seront inversées, +X sera à droite de la table et -X à gauche. Cette inversion ne change rien du point de vue du sens de déplacement de l’outil.

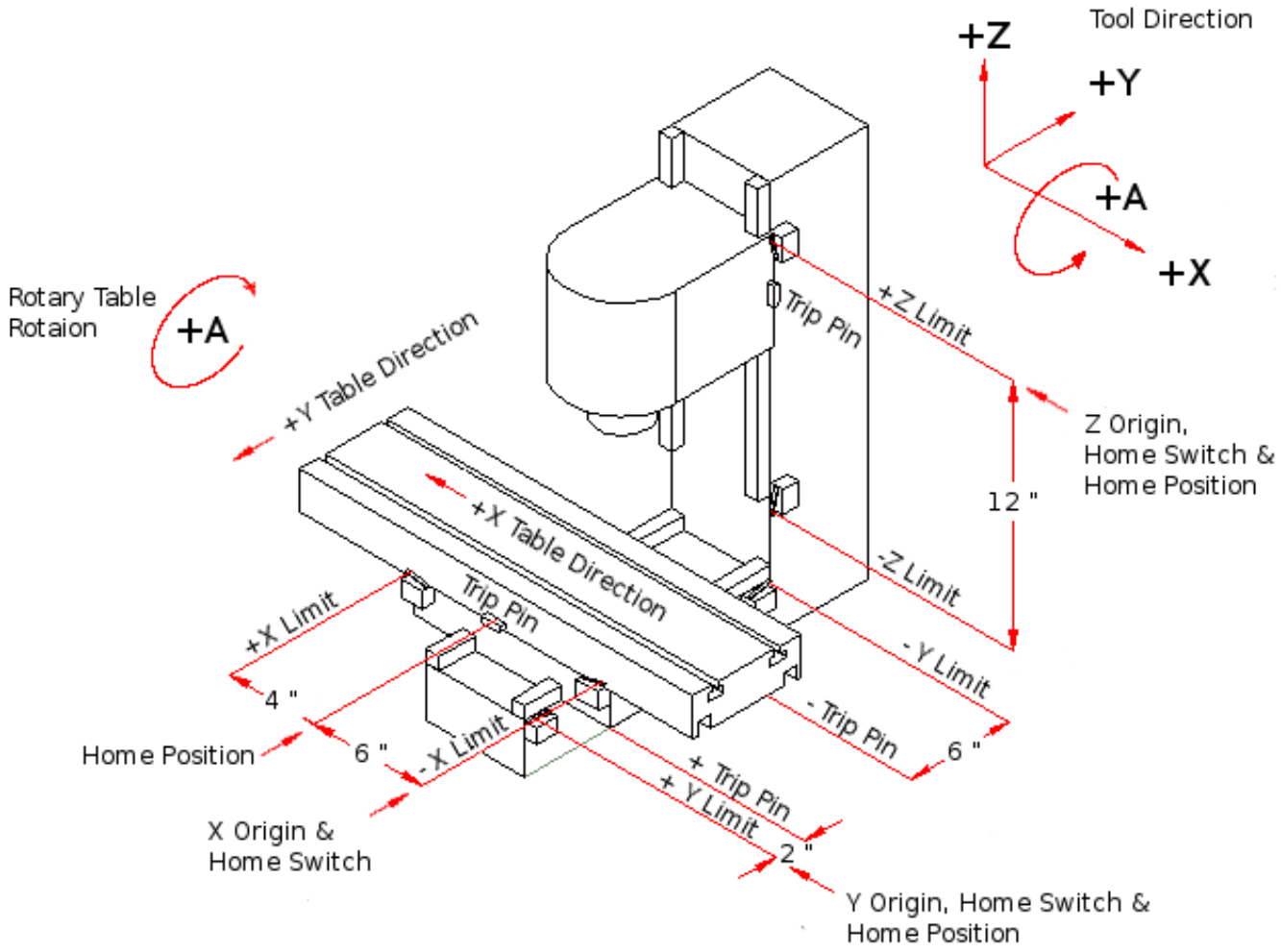


Figure 3.6: Configuration typique d’une fraiseuse

Le dessin suivant montre les directions de déplacement de l’outil et la position des fins de course de limite sur un tour classique.

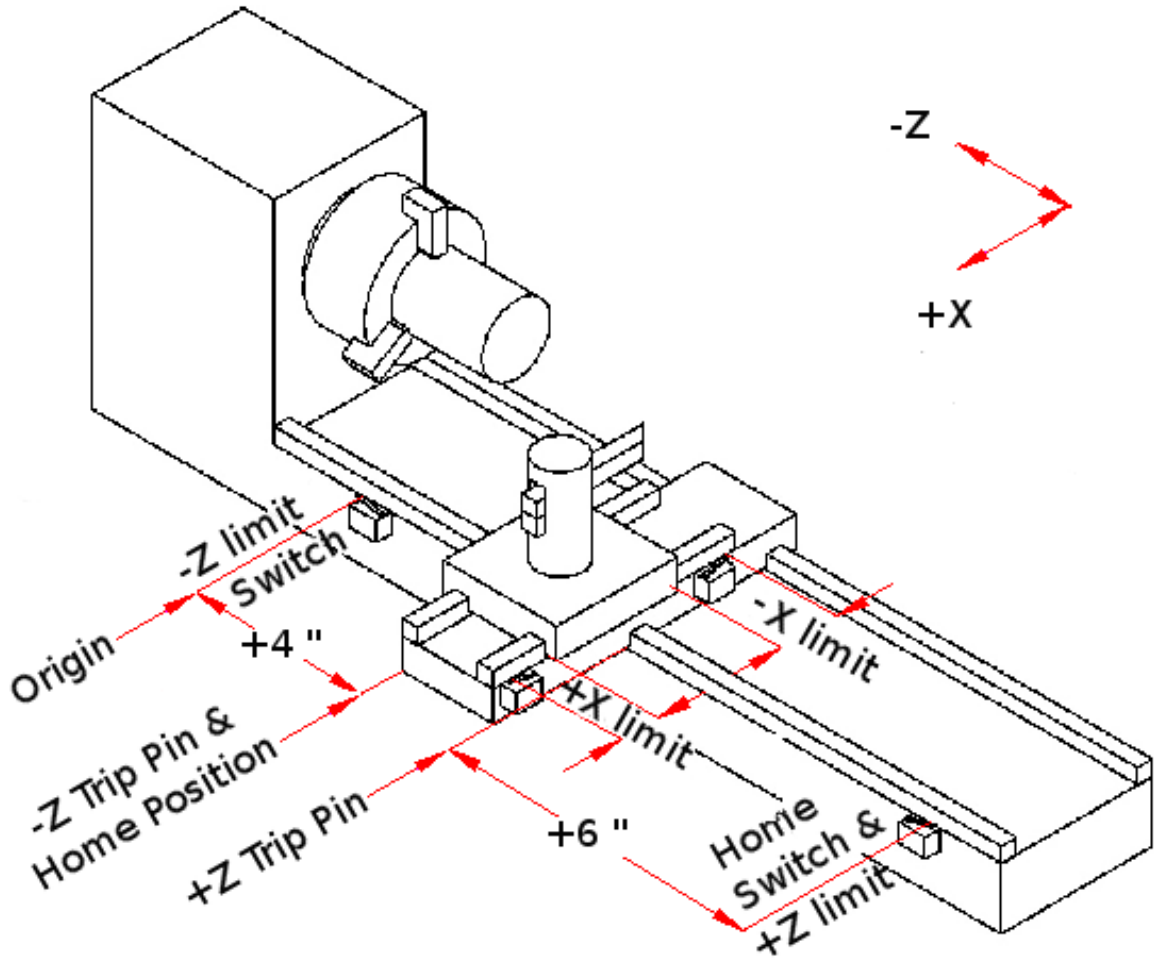


Figure 3.7: Configuration typique d'un tour

3.3.2 Contrôle de trajectoire

3.3.2.1 La planification de trajectoire

La planification de trajectoire est en général, le moyen qui permet à LinuxCNC de suivre le chemin spécifié par le programme G-code, tout en restant dans les limites permises par la machine.

Un programme en G-code ne peut jamais être exactement suivi. Par exemple imaginez que vous spécifiez dans une ligne du programme les mouvements suivants:

```
G1 X10 F100 (G1 un mouvement linéaire, X10 la destination, F100 la vitesse)
```

En réalité, la totalité du mouvement ne peut pas être effectuée à F100, puisque la machine commence le mouvement à une vitesse nulle, elle doit accélérer pour se déplacer vers X=10, puis décélérer pour revenir à une vitesse nulle en fin de mouvement. Parfois une portion du mouvement se fera bien à F100, mais pour beaucoup de mouvements, spécialement les petits mouvements, la vitesse spécifiée ne sera jamais atteinte.

Les accélérations et décélérations de base décrite ici ne sont pas complexes et ne nécessite pas de compromis. Les contraintes des axes de la machine sont placés dans le fichier INI, comme la vitesse maximum de l'axe et l'accélération ne devant pas être dépassées par le planificateur de trajectoire.

3.3.2.2 Le suivi du parcours

Un problème plus compliqué est posé par le suivi du parcours. Quand vous programmez un angle droit en G-code, le planificateur de trajectoire peut suivre différents parcours, tous sont bons dans certains cas; il peut décélérer et s'arrêter exactement sur les coordonnées du sommet de l'angle, puis accélérer dans la direction perpendiculaire. Il peut également faire ce qui est appelé le mode trajectoire continue, qui consiste à maintenir la vitesse d'avance en passant vers le sommet de l'angle, ce qui nécessite d'arrondir l'angle de façon à respecter les contraintes machine. Vous pouvez remarquer qu'il y a dans ce cas un compromis: vous pouvez ralentir pour avoir un meilleur suivi du parcours, ou conserver une vitesse d'avance élevée au détriment de la finesse des angles, du fait d'un moins bon suivi du parcours. Selon les particularités de l'usinage, du matériau, de l'outillage, etc., le programmeur devra décider du bon compromis.

3.3.2.3 La programmation du planificateur

Les commandes de contrôle de trajectoire sont les suivantes:

G61
: (mode trajectoire exacte) G61 indique au planificateur de suivre exactement la trajectoire prévue.

G61.1
: (mode Arrêt exact) G61.1 demande au planificateur de s'arrêter exactement à la fin de chaque segment. Le parcours sera suivi avec exactitude mais les arrêts complets de l'avance peuvent se révéler destructeurs pour la pièce ou l'outillage, selon les particularités de l'usinage.

G64
: (mode trajectoire continue sans tolérance) Le mode G64 est le mode par défaut au démarrage de LinuxCNC. G64 est juste une trajectoire continue, le Détecteur naive CAM n'est pas activé. G64 et G64 P0 indiquent au planificateur de sacrifier la précision de suivi du parcours pour conserver une vitesse d'avance élevée. Ce mode est nécessaire pour certains types de matériaux ou d'outillages pour lesquels l'arrêt exact est dangereux. Il peut très bien fonctionner tant que le programmeur garde à l'esprit que le parcours d'outil pourra être plus arrondi que celui indiqué par le programme. Dans le cas d'un mouvement en G0 (rapide) avec G64, faire preuve de prudence sur les mouvements de dégagement et prévoir suffisamment de distance pour éviter les obstacles selon les capacités d'accélérations de la machine.

G64 Px.xxx
: (mode trajectoire continue avec tolérance) Ce mode active le Détecteur naive CAM" et active le mode trajectoire continue avec tolérance. Si vous utilisez le millimètre comme unité et programmez G64 P1.27, vous dites au planificateur que vous souhaitez une vitesse d'avance continue, mais qu'aux coins programmés vous voulez un ralentissement suffisant pour que le parcours de l'outil puisse rester à moins de 1.27mm du parcours programmé. L'amplitude exacte du ralentissement dépend de la géométrie de l'angle programmé et des contraintes machine, mais la seule chose dont le programmeur ait à se soucier est la tolérance, ce qui lui donne le contrôle complet des compromis du suivi de parcours. La tolérance de ce mode peut être modifiée tout au long du programme si nécessaire. Attention: spécifier un G64 P0 aura le même effet qu'un G64 seul (voir ci-dessus), c'est rendu nécessaire pour conserver la compatibilité ascendante avec les anciens programmes G-code. Voir le chapitre sur le G-code pour plus d'information sur G64 P-Q-.

Trajectoire continue sans tolérance
: Le point contrôlé touchera chaque mouvement spécifié à au moins un point. La machine ne pourra jamais se déplacer à une vitesse d'avance telle qu'elle ne puisse pas s'arrêter avec précision à la fin du mouvement en cours (ou du prochain mouvement, si vous mettez en pause lorsque la trajectoire est déjà commencée). La distance avec le point final du mouvement est aussi grande que nécessaire pour maintenir la meilleure vitesse d'avance possible pendant le parcours.

Détecteur Naive Cam

Les mouvements successifs en G1, concernant uniquement les axes XYZ, dont la déviation par rapport à une ligne droite est inférieur à P, sont fusionnés en une seule ligne droite. Ce mouvement fusionné remplace les mouvements individuels en G1 pour obtenir une nouvelle trajectoire avec tolérance. Entre les mouvements successifs, le point contrôlé ne passera jamais à plus de P- du point final du mouvement en cours. Le point contrôlé touchera au moins un point de chacun des mouvements. La machine ne pourra jamais se déplacer à une vitesse ne lui permettant pas de venir s'arrêter exactement à la fin du mouvement actuel (ou du prochain mouvement, si vous mettez en pause lorsque la trajectoire est déjà commencée). En mouvement G2/3 dans le plan G17 (XY) quand la déviation maximale entre un arc et une ligne droite est plus petite que la tolérance G64 Q- l'arc est brisé en deux lignes (du début de l'arc à son milieu et du milieu à la fin de l'arc). Ces deux tronçons sont ensuite soumis à l'algorithme Naïve cam des lignes. Ainsi, les cas ligne-arc, arc-arc et arc-ligne, comme les cas ligne-ligne bénéficient du traitement Détecteur naive CAM. Les performances de contourage sont accrues grâce à la simplification de la trajectoire.

Dans la figure suivante la ligne bleue représente la vitesse machine actuelle. La ligne rouge représente la capacité d'accélération de la machine. La ligne horizontale sous chaque tracé est le mouvement planifié. Le tracé supérieur montre comment le planificateur de trajectoire ralenti la machine quand des petits mouvements sont rencontrés. Ceci pour rester dans les limites fixées par les paramètres d'accélération de la machine et être capable de s'arrêter exactement à la fin du prochain mouvement. Le tracé du bas montre l'effet du détecteur Naive Cam pour combiner les mouvements et fournir une amélioration conséquente dans le suivi de la vitesse programmée.

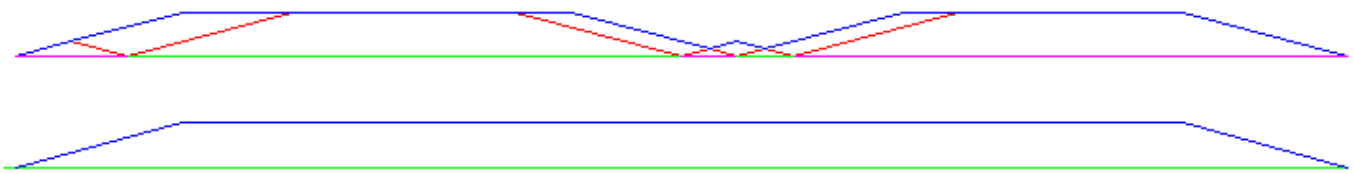


Figure 3.8: Détecteur Naive Cam

3.3.2.4 Planification des mouvements

Assurez-vous que les mouvements soient assez longs pour convenir à votre machine/matériel. Principalement en raison de la règle selon laquelle "la machine ne pourra jamais se déplacer à une vitesse ne lui permettant pas de venir s'arrêter complètement à la fin du mouvement actuel", il y a une longueur minimale de déplacement permettant à la machine d'atteindre la vitesse demandée avec un réglage d'accélération donné.

Les phases d'accélération et de décélération utilisent chacune la moitié de la variable MAX_ACCELERATION du fichier .ini. Avec une trajectoire continue c'est exactement inversé, ce qui fait que l'accélération totale de l'axe est égal à la variable MAX_ACCELERATION. Dans d'autres cas, l'accélération actuelle de la machine est un peu inférieure à celle du fichier ini.

Pour maintenir la vitesse d'avance, le mouvement doit être plus long que la distance qui lui est nécessaire pour accélérer de zéro à la vitesse souhaitée, puis de décélérer pour s'arrêter. En utilisant A comme étant 1/2 de la variable MAX_ACCELERATION du fichier ini et F comme étant la vitesse d'avance en unités par seconde, le temps d'accélération sera t_a = F/A et la distance d'accélération sera d_a = F*t_a/2. Les temps et distance de décélération sont les mêmes, ce qui fait que la distance critique d = d_a + d_d = 2 * d_a = F^2/A.

Par exemple, pour une vitesse d'avance de 25mm par seconde et une accélération de 250 mm/sec^2, la distance critique sera de 10^2/100 = 100/100 = 1mm. Pour une vitesse d'avance de 5mm par seconde, la distance critique ne serait que de 5^2/100 = 25/100 = 0.25mm.

3.3.3 G-code

3.3.3.1 Par défaut

Quand LinuxCNC démarre pour la première fois beaucoup de G et M codes sont chargés par défaut. Les codes actifs courants sont visibles dans l'interface Axis, dans l'onglet Données manuelles dans le champ G-codes actifs Ces codes G et M définissent le comportement de LinuxCNC et il est important de bien comprendre la signification de chacun avant de démarrer LinuxCNC. Ces codes par défaut peuvent être modifiés lors du lancement d'un fichier de G-codes puis laissés dans différents états qui seront identiques lors d'une nouvelle session de LinuxCNC. La bonne pratique consiste à mettre dans le préambule de chaque fichier de G-codes les codes nécessaires pour le travail demandé et ne pas supposer que ceux par défaut conviendront. Imprimer la page des références rapides du G-code peut aider à se rappeler la signification de chacun d'eux.

3.3.4 Vitesse d'avance

Si vous avez un tour ou un axe rotatif, pour savoir comment la vitesse d'avance s'applique selon que l'axe est linéaire ou rotatif, lire et comprendre la section vitesse d'avance du manuel de l'utilisateur.

3.3.5 Compensation de rayon d'outil

La compensation de rayon d'outil (G41/G42) nécessite que l'outil puisse usiner tout au long de la trajectoire programmée sans interférer avec les mouvements d'entrée ou de sortie. Si c'est impossible avec le diamètre de l'outil courant, une erreur est signalée. Un diamètre d'outil inférieur est peut être utilisable sans erreur pour le même parcours. Ce qui signifie que quand ce type de problème se présente, il est possible de programmer un outil plus petit pour usiner le même parcours sans erreur. Voir la section compensation de rayon d'outil pour plus d'informations.

3.3.6 Prise d'origine machine

Après le démarrage de LinuxCNC chaque axe doit être référencé sur son point d'origine machine avant tout mouvement ou commande MDI.

Pour déroger à ce comportement par défaut, ou pour utiliser l'interface Mini, il est possible d'ajuster l'option NO_FORCE_HOMING = 1 dans la section [TRAJ] du fichier ini.

3.3.7 Changement d'outil

Il existe plusieurs options pour effectuer un changement d'outil. Voir la section [EMCIO] dans le manuel de l'intégrateur pour les informations sur la configuration de ces options. Voir également les sections G28 et G30 du manuel de l'utilisateur.

3.3.8 Systèmes de coordonnées

Les systèmes de coordonnées peuvent être déroutant au premier abord. Avant de démarrer une machine CNC, il est important de bien comprendre les bases des systèmes utilisés par LinuxCNC. Pour explorer plus en profondeur les systèmes de coordonnées utilisés par LinuxCNC, voir la section xxxxx de ce manuel.

3.3.8.1 G53 Coordonnées machine

Quand vous réalisez une prise d'origine de plusieurs axes de LinuxCNC, vous passez G53, les coordonnées système, à 0 pour chacun des axes concernés.

- La prises d'origine ne modifient en rien les autres systèmes de coordonnées, ni les compensations d'outil.

La seule façon de se déplacer en mode G53, en coordonnées machine, c'est de programmer un G53 sur la même ligne que celle d'un mouvement. En fonctionnement normal, vous êtes dans le système de coordonnées G54.

3.3.8.2 G54 à 59.3 Coordonnées utilisateur

Normalement vous utilisez le système de coordonnées G54. Quand un décalage est appliqué au système de coordonnées utilisateur courant, dans Axis, une petite sphère bleue avec des rayons est affichée à l'emplacement de l'origine machine quand la visu affiche Position: Relative Actuelle. Si votre décalage utilise temporairement les coordonnées machine, depuis le menu Machine ou en programmant G10 L2 P1 X0 Y0 Z0 à la fin du programme G-Code. Modifiez la valeur du mot P en fonction du système de coordonnées dont vous voulez effacer le décalage.

- Les décalages stockés dans un système de coordonnées utilisateur sont conservés à l'arrêt de LinuxCNC.
- Dans Axis, utiliser le bouton Toucher décalera le système de coordonnées utilisateur choisi.

3.3.8.3 Quand vous êtes perdu

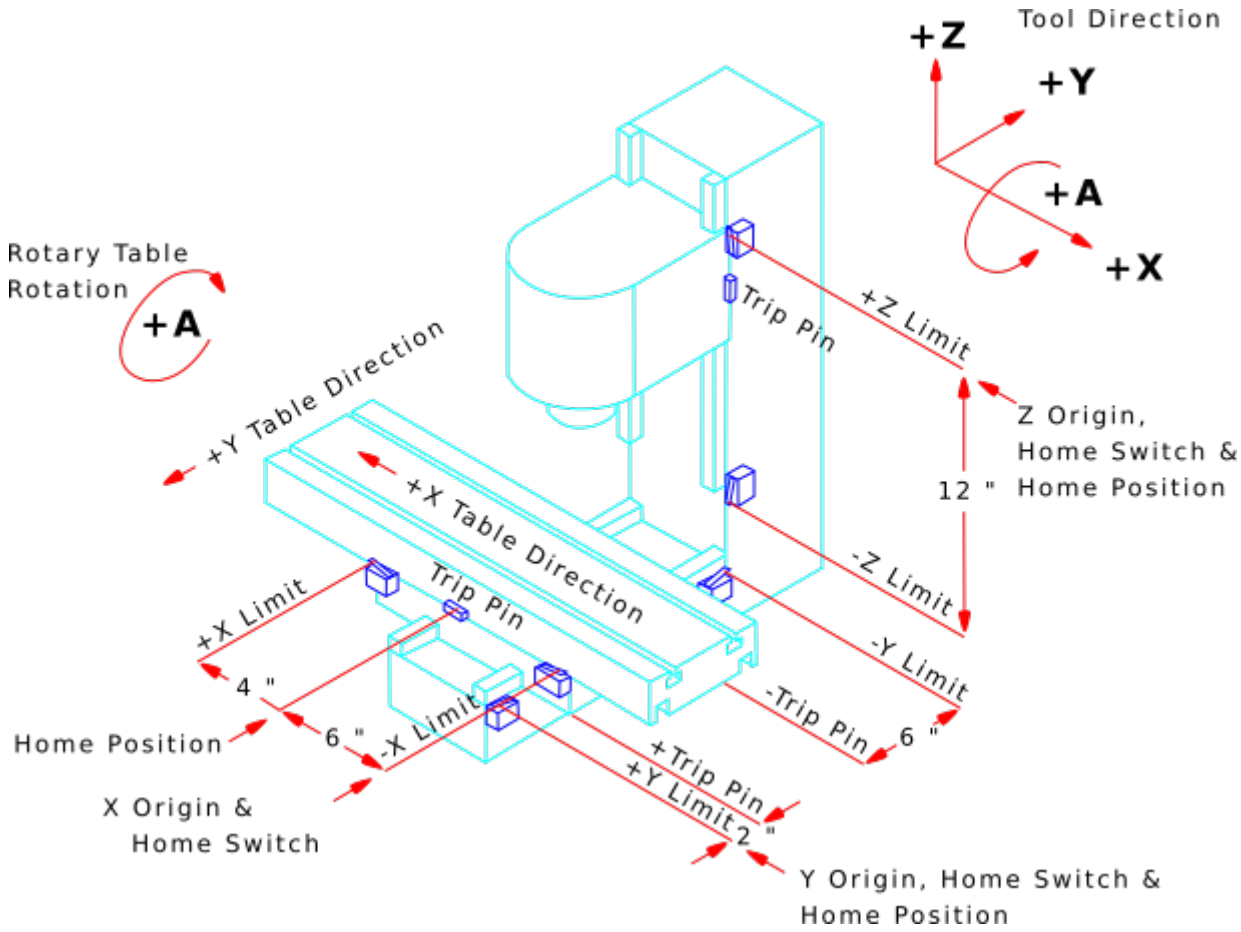
Si vous avez des difficultés pour obtenir 0,0,0 sur la visu alors que vous pensez que vous devriez l'avoir, c'est peut être provoqué par plusieurs décalages programmés et qu'il conviendrait de supprimer. Pour cela:

- Placez vous sur l'origine machine avec G53 G0 X0 Y0 Z0
- Supprimez tous les décalages G92 avec G92.1
- Utilisez les coordonnées utilisateur avec G54
- Rendez les coordonnées utilisateur G54, identiques aux coordonnées machine avec G10 L2 P1 X0 Y0 Z0 R0
- Annulez les offsets d'outil avec G49
- Activez l'affichage des coordonnées relatives depuis le menu.

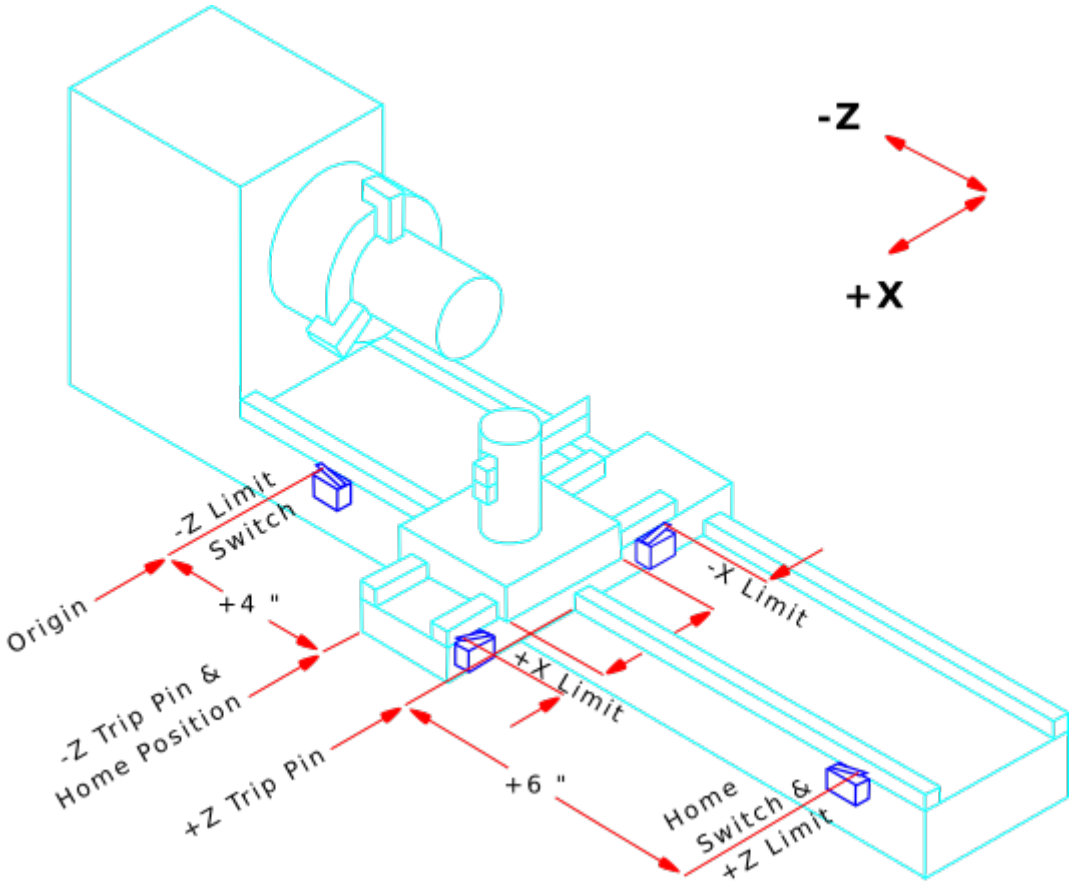
Maintenant vous devriez être, à l'origine machine X0 Y0 Z0 et le système de coordonnées relatives devrait être le même que le système de coordonnées machine.

3.3.9 Machine Configurations

The following diagram shows a typical mill showing direction of travel of the tool and the mill table and limit switches. Notice how the mill table moves in the opposite direction of the Cartesian coordinate system arrows shown by the Tool Direction image. This makes the tool move in the correct direction in relation to the material.



The following diagram shows a typical lathe showing direction of travel of the tool and limit switches.



3.4 Aperçu global d'une machine CNC

Cette section donne une description des différents organes constituant une machine à commande numérique (CNC).

3.4.1 Composants mécaniques

Une machine à commande numérique 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 autres composants mécaniques, comme les boutons de jog, ne seront pas décrits ici, même si ils affectent le contrôle.

3.4.1.1 Axes

Toute machine à commande numérique 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 XYUV.[2] [3]

[2] 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.

[3] Avec LinuxCNC, 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.

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

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.

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.

3.4.1.2 Broche

Une machine à commande numérique est équipée d'une broche qui maintient un outil coupant, un palpeur ou 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.

3.4.1.3 Arrosages

Une machine à commande numérique peut être équipée d'un système fournissant l'arrosage fluide ou un arrosage par gouttelettes.

3.4.1.4 Correcteurs de vitesse d'avance et de broche

Une machine à commande numérique est équipée 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.

3.4.1.5 Bouton d'effacement de bloc

Une machine à commande numérique peut être équipée d'un bouton d'effacement de bloc. Voir la section effacement de bloc.

3.4.1.6 Bouton d'arrêt optionnel du programme

Une machine à commande numérique peut être équipée d'un bouton d'arrêt du programme. Voir la section arrêts optionnels.

3.4.2 Composants de contrôle et de données

3.4.2.1 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.

3.4.2.2 Axes linéaires secondaires

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.

3.4.2.3 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. [4]

3.4.2.4 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 à 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.

3.4.2.5 Mouvement linéaire coordonné

Pour mener un outil sur une trajectoire spécifiée, une machine à commande numérique 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 physiques de l'axe rendent le déplacement impossible, tous les axes seront ralentis pour maintenir le parcours prévu.

3.4.2.6 Vitesse d'avance

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 sur les modes de vitesse, 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.

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

3.4.2.7 Arrosage

Arrosage fluide ou par gouttelettes peuvent être activés séparément. Le langage RS274/NGC les arrête ensemble, voir la section des contrôles d'arrosage.

3.4.2.8 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 la machine s'arrêtera exactement à la fin du dernier mouvement avant la temporisation.

3.4.2.9 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 des vitesses.

3.4.2.10 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.

3.4.2.11 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.

3.4.2.12 carrousel d'outils

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

3.4.2.13 Changeur d'outil

Une machine à commande numérique peut commander un changeur d'outils.

3.4.2.14 Chargeur de pièce

Les deux porte-pièces peuvent être intervertis par commande.

3.4.2.15 Chargeur de pièces

Une machine à commande numérique peut être équipée 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.

3.4.2.16 Boutons des correcteurs de vitesses

Les boutons des correcteurs de vitesses peuvent être activés (ils fonctionnent normalement) ou rendus inopérants (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 l'inhibition et l'activation des correcteurs de vitesse. Voir également ici pour d'autres détails.

3.4.2.17 Modes de contrôle de trajectoire

La machine peut être placée dans un de ces trois modes de contrôle de trajectoire:

- mode arrêt exact:: En mode arrêt exact, le mobile s'arrête brièvement à la fin de chaque mouvement programmé.
- mode trajectoire exacte:: 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.
- mode trajectoire continue avec tolérance optionnelle:: 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 également les G-codes G61/G61.1 et G64 des contrôles de trajectoire.

== 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.

3.4.2.18 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.

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

3.4.2.19 Bouton d'effacement de bloc

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

3.4.2.20 Bouton d'arrêt optionnel du programme

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

3.4.3 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 nom de la table d'outils est défini sous cette forme dans le fichier ini:

```
[EMCIO]

# tool table file
TOOL_TABLE = tooltable.tbl
```

Il est également possible de donner à la table d'outils le même nom que le fichier ini, mais avec une extension tbl, par exemple:

```
TOOL_TABLE = acme_300.tbl
```

ou encore:

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

D'autres informations sont disponibles sur les spécificités du format de la table d'outils.

3.4.4 Paramètres

Dans le langage RS274/NGC, la machine 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. LinuxCNC 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.

Tous les paramètres sont disponibles pour une utilisation dans les programmes de G-code.

Un fichier de paramètres 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 le tableau ci-dessous, 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 décrit ci-dessous, 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 compris dans une fourchette de 1 à 5400. Les numéros de paramètre doivent être disposés dans l'ordre croissant. 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.

Table 3.1: Format d'un fichier de paramètres

Numéro d'index	**Valeur**	**Commentaire**
5161	0.0	G28 pom X
5162	0.0	G28 pom Y

3.5 Lancer LinuxCNC

LinuxCNC se lance comme un autre programme Linux: depuis un terminal en passant la commande linuxcnc, ou depuis le menu Applications → CNC.

3.5.1 Sélecteur de configuration

Le Sélecteur de configuration s'affichera à chaque fois que vous lancerez LinuxCNC depuis le menu Applications → CNC → LinuxCNC. Vos propres configurations personnalisées s'affichent dans le haut de la liste, suivies par les différentes configurations fournies en standard. Étant 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 fonctionneront toutes, même sans matériel raccordé, ce sont des simulations de machines.

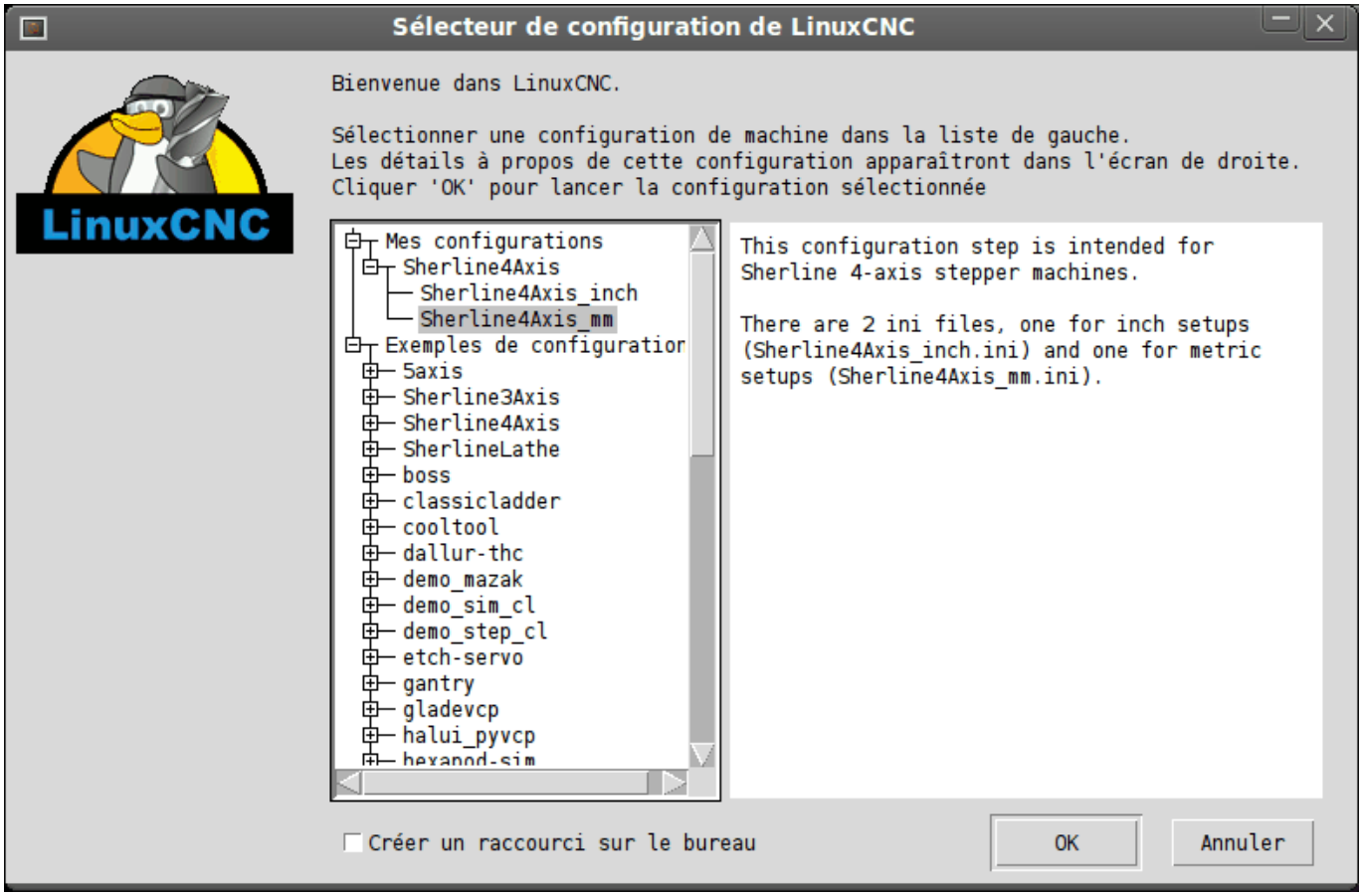


Figure 3.9: Sélecteur de configuration pour LinuxCNC

Cliquez dans la liste, sur les différentes configurations pour afficher les informations les concernant. Double-cliquez sur une configuration ou cliquez OK pour démarrer LinuxCNC 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 LinuxCNC 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 linuxcnc 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, LinuxCNC 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.

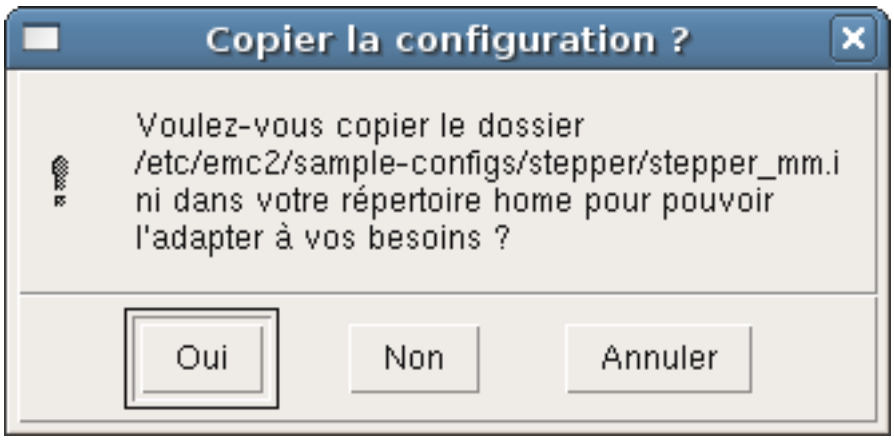


Figure 3.10: Dialogue de copie de la configuration

3.5.2 L'interface utilisateur graphique Axis

L'interface AXIS est une des interfaces parmi lesquelles vous avez à choisir. Elle peut être configurée pour lui ajouter un panneau de commandes virtuel personnalisé en fonction des besoins. AXIS est l'interface utilisateur par défaut et est activement développée. C'est aussi la plus populaire.

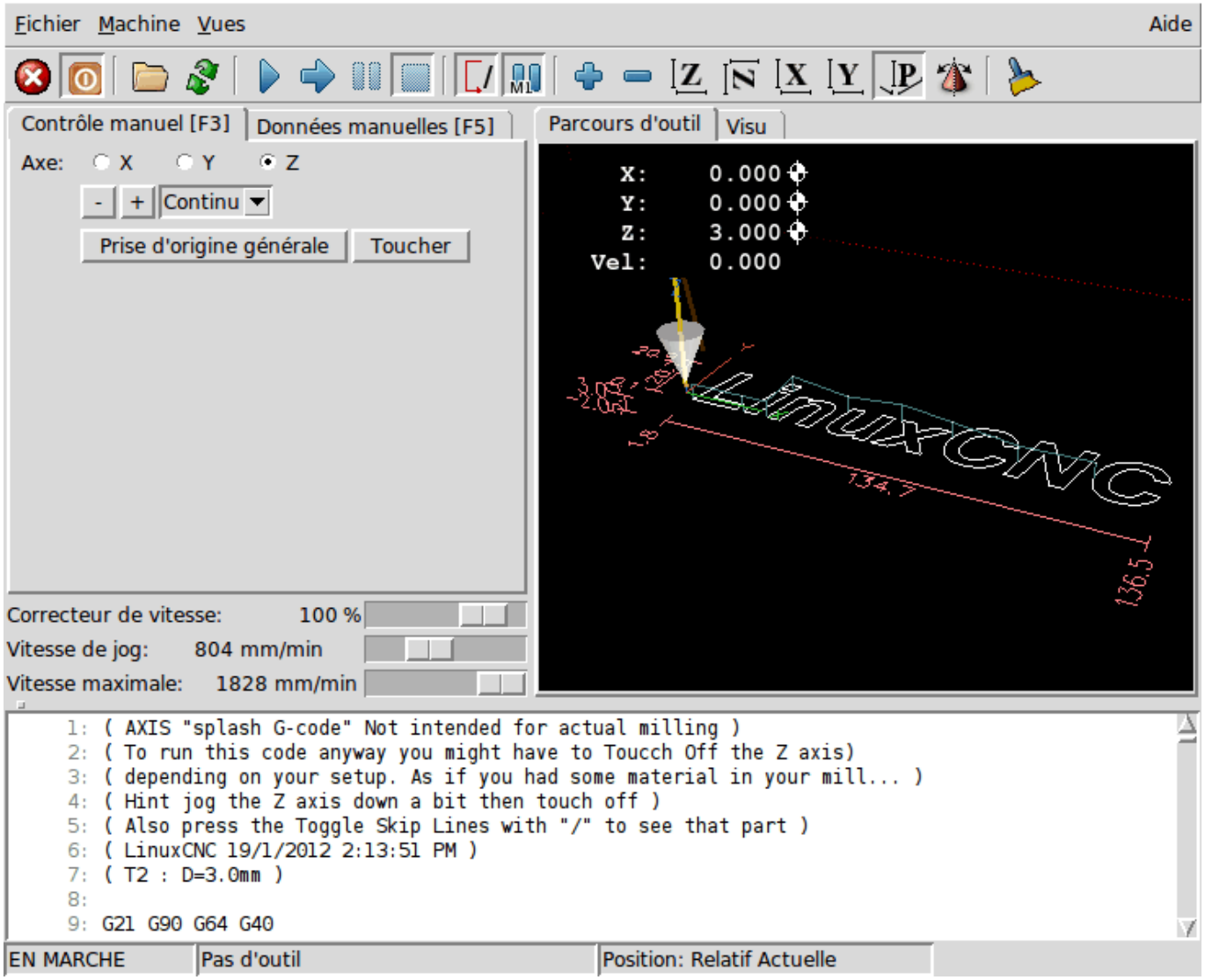


Figure 3.11: Interface Axis

3.5.3 Les étapes suivantes de la configuration

Après avoir trouvé l'exemple de configuration qui utilise le même matériel que votre machine, et en avoir enregistré une copie dans votre répertoire personnel, vous pouvez la personnaliser en fonction des besoins spécifiques à votre machine. Consultez le Manuel de l'intégrateur pour tous les détails de configuration.

Si vous souhaitez créer une configuration personnalisée, vous pouvez utiliser pour cela, un des assistants graphiques de configuration, StepConf ou PncConf selon votre type de machine.

= Assistant graphique de configuration StepConf

3.5.4 Introduction

LinuxCNC est capable de contrôler un large éventail de machines utilisant de nombreuses interfaces matérielles différentes.

Stepconf est un programme qui génère des fichiers de configuration LinuxCNC pour une classe spécifique de machine CNC: celles qui sont pilotées via un, ou plusieurs ports parallèles standards et contrôlées par des signaux de type pas/direction (step/dir).

Stepconf est installé en même temps que LinuxCNC et un lanceur se trouve dans le menu Application → CNC → LinuxCNC StepConf.

Stepconf place les fichiers qu'il crée dans le répertoire ~/linuxcnc/config pour y stocker les paramètres de chaque configuration. Lorsque quelque chose doit être modifié, il faut choisir le fichier correspondant au nom de la configuration et portant l'extension .stepconf.

L'Assistant Stepconf a besoin, au minimum, d'une résolution de 800 x 600 pour que les boutons sur le bas des pages soient apparents.

3.5.5 Page d'accueil

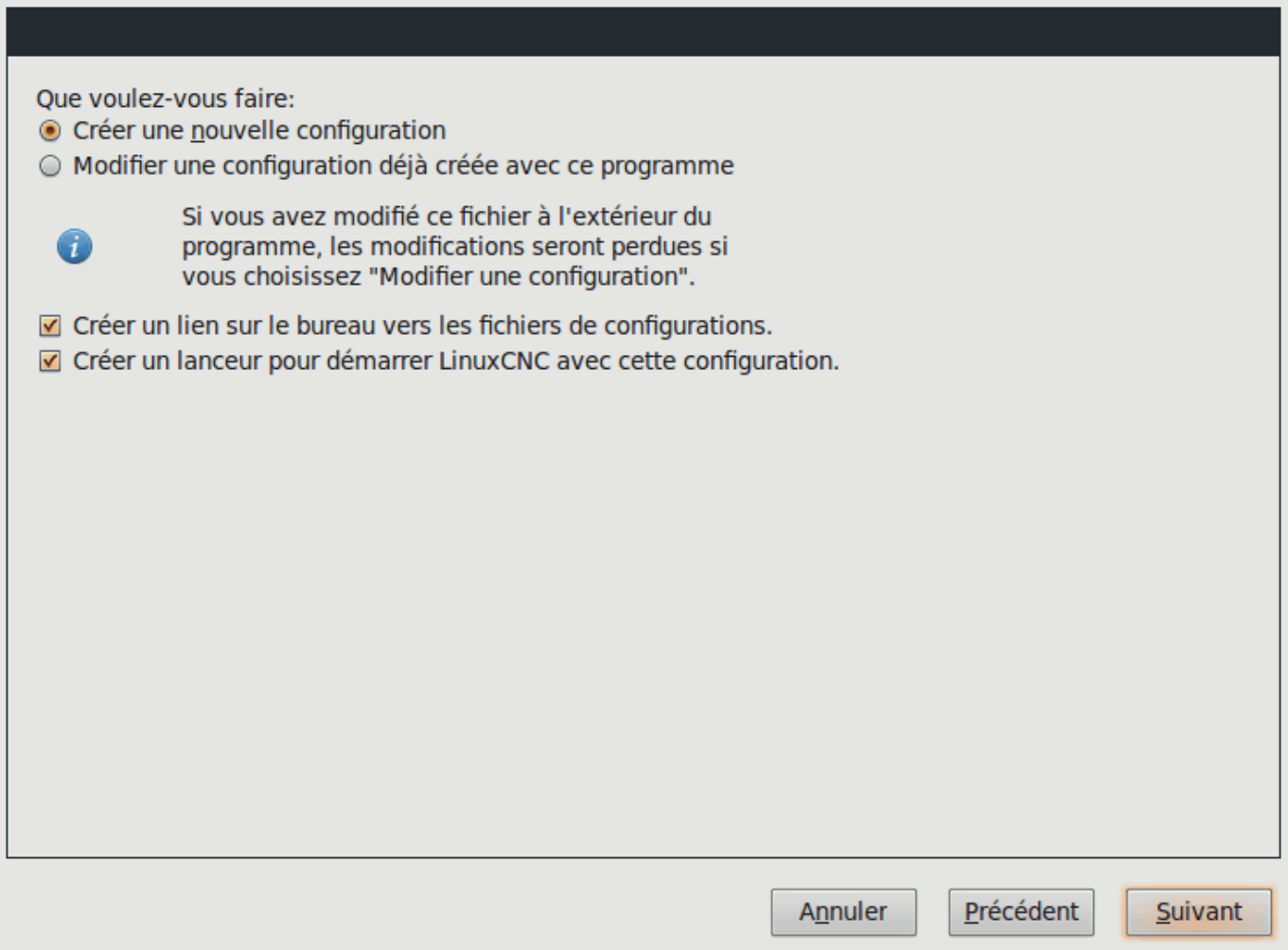


Figure 3.12: La page d'accueil de stepconf

- Créer une nouvelle configuration - Créera une configuration nouvelle.
- Modifier une configuration déjà créée - Modifiera une configuration existante, déjà créée avec Stepconf. Après la sélection de celle-ci un sélecteur de fichier s'ouvre pour y choisir le fichier .stepconf à modifier. Si des modifications sont faites aux fichiers .hal et .ini avec un autre éditeur, ils ne seront

plus utilisables par Stepconf. Les modifications de custom.hal et de custom_postgui.hal, par contre, sont conservées par Stepconf.

- Créer un lien - Placera un lien sur le bureau, pointant sur le dossier des fichiers de configuration.
- Créer un lanceur - Placera un lanceur sur le bureau pour démarrer l'application avec sa configuration.

3.5.6 Informations machine

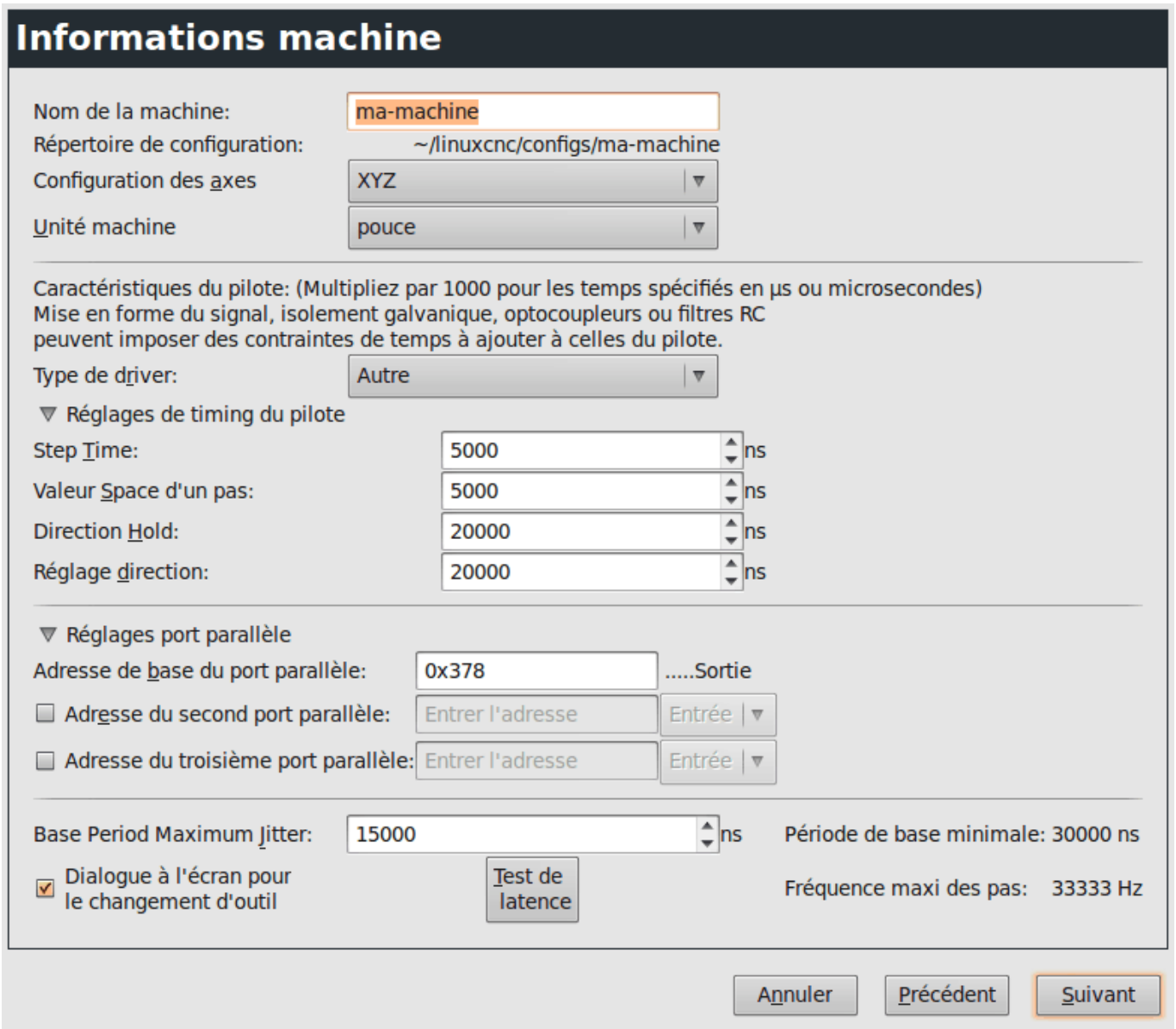


Figure 3.13: Page d'informations sur la machine

- Nom de la machine - Choisir un nom pour la machine. Utiliser uniquement des lettres majuscules, minuscules, des chiffres ou "-" et "_".
- Configuration des axes - Choisir les axes correspondants à la machine: XYZ (fraiseuse 3 axes), XYZA (fraiseuse 4 axes) ou XZ (tour).

- Unité machine - Choisir entre le pouce et le millimètre. Toutes les questions suivantes (telles que la longueur des courses, le pas de la vis, etc) devront obtenir des réponses dans l'unité choisie ici.
- Caractéristiques du pilote - Si un des pilotes énumérés dans la liste déroulante peut être utilisé, cliquer sur son nom. Sinon, trouver les 4 valeurs de timing dans la fiche de caractéristiques fournie par le fabricant du pilote et les saisir. Si la fiche donne des valeurs en microsecondes, les multiplier par 1000. pour les convertir en nanosecondes. Par exemple, pour 4.5µs saisir 4500ns.

Une liste de certains des pilotes pas à pas les plus populaires, avec leurs valeurs caractéristiques de timing, se trouve sur le wiki de LinuxCNC.org à la page Timming des pilotes.

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. La Configuration LinuxCNC pour Sherline est déjà réglée.

- Step Time - Durée de la largeur de l'impulsion de pas à l'état on, en nanosecondes.
- Valeur Space d'un pas - Temps entre deux impulsions de pas, en nanosecondes.
- Direction Hold - Durée de maintien du signal après un changement de direction, en nanosecondes.
- Réglage direction - Délai avant le changement de direction après la dernière impulsion de pas, en nanosecondes.
- Adresse de base du premier port parallèle - Généralement l'adresse 0x378 est correcte pour le premier port. Le premier port a toujours ses broches 2 à 9 configurées en sortie.
- Adresse du second port parallèle - Si un second port parallèle est nécessaire, entrer son adresse ici. Si ce port est intégré à la carte mère il est possible de vérifier leur ordre dans le BIOS, habituellement 0x378 0x278 0x3bc. Attention les cartes additionnelles ont d'autres adresses. Dans ce cas, la commande lspci -v dans un terminal peux aider, si le nom du chipset de la carte est connu. Plus de détails à ce sujet sont disponibles dans le manuel de l'intégrateur. Le bouton situé à droite du champs d'adresse du port, permet de choisir le sens des broches 2 à 9, soit comme étant des entrée, soit comme étant des sorties.
- Adresse du troisième port parallèle - Si un troisième port parallèle est nécessaire, entrer son adresse ici. Si ce port est intégré à la carte mère il est possible de vérifier leur ordre dans le BIOS, habituellement 0x378 0x278 0x3bc. Attention les cartes additionnelles ont d'autres adresses. Dans ce cas, la commande lspci -v dans un terminal peux aider, si le nom du chipset de la carte est connu. Plus de détails à ce sujet sont disponibles dans le manuel de l'intégrateur. Le bouton situé à droite du champs d'adresse du port, permet de choisir le sens des broches 2 à 9, soit comme étant des entrée, soit comme étant des sorties.
- Période de base minimale - En se basant sur les caractéristiques du pilote et sur le résultat du test de latence, Stepconf détermine automatiquement la période de base (BASE_PERIOD) la plus petite utilisable et l'affiche ici.
- 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.
- Base Period Maximum Jitter - Après un test de latence, entrer ici la valeur retournée dans la colonne "Max Jitter" et à la ligne "Base thread". Cette valeur correspond à la latence maximale du PC testé. Pour exécuter directement un test de latence cliquer sur le bouton Test de latence. Lire le chapitre sur le test de latence pour tous les détails concernant ce test.
- Dialogue à l'écran pour le changement d'outil - Si cette case est cochée, LinuxCNC va faire une pause et ouvrir un dialogue pour charger l'outil <n> lorsque qu'un G-Code M6 T<n> sera rencontré. Laisser cette case cochée sauf si le support d'un changeur d'outils automatique est prévu dans un fichier personnalisé HAL.

3.5.7 Options de configuration avancée

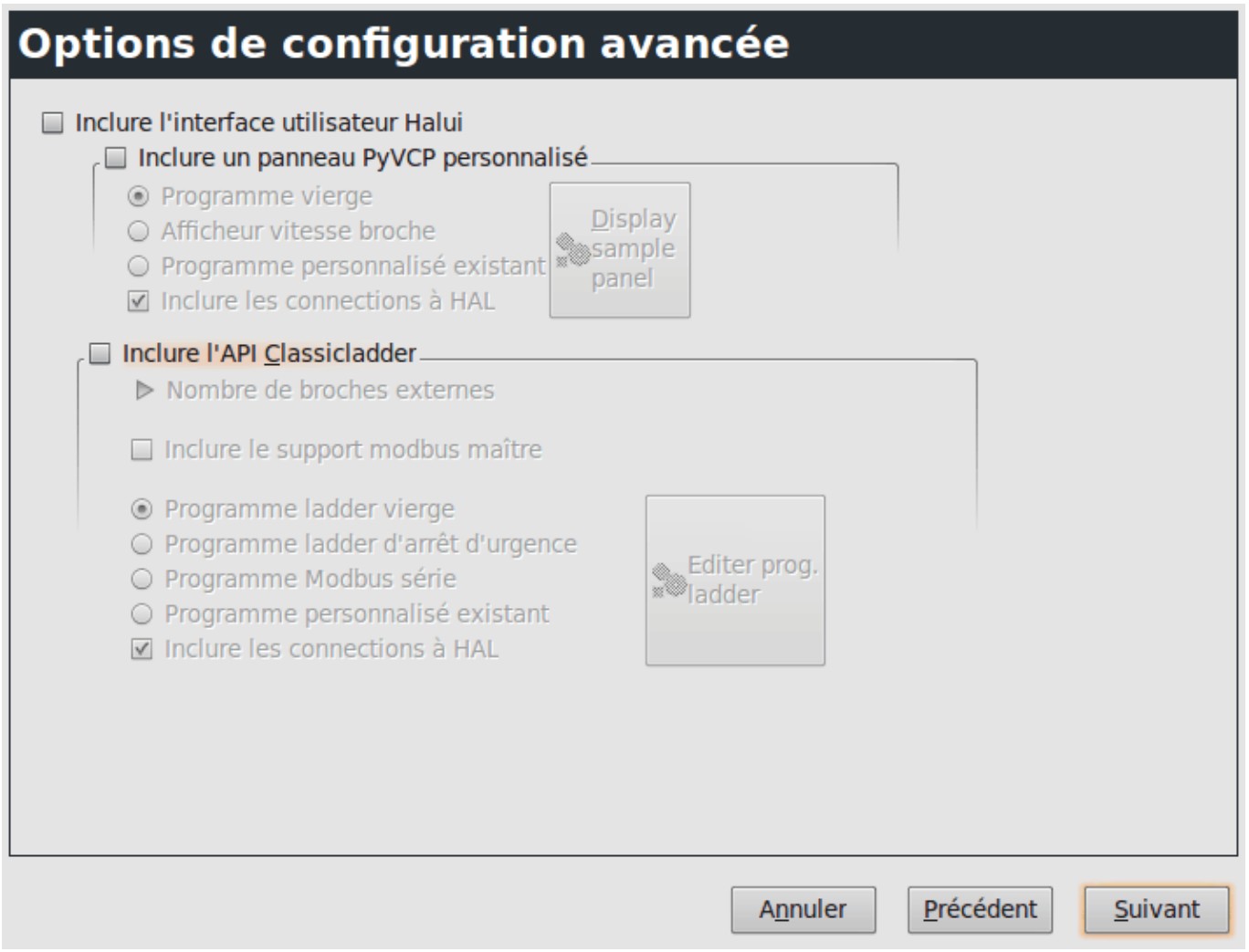


Figure 3.14: Configuration avancée

- Inclure l'interface Halui - Ajoutera l'interface utilisateur Halui. Voir le manuel de l'intégrateur pour plus d'informations sur Halui.
- Inclure un panneau pyVCP - Ceci ajoutera un panneau pyVCP de base, avec son fichier de configuration sur lequel il sera possible de travailler. Quelques options sont disponibles pour enrichir le panneau grâce à des cases à cocher. Voir le manuel de l'intégrateur pour plus d'information sur pyVCP.
- Inclure l'API ClassicLadder - Cette option ajoutera l'automate programmable en logique à contacts ClassicLadder. Un certain nombre d'options sont disponibles pour enrichir l'API grâce à des cases à cocher. L'éditeur de programme ladder est accessible par le bouton Editer prog. ladder Voir le manuel de l'intégrateur pour plus d'information sur ClassicLadder.

3.5.8 Réglage du port parallèle

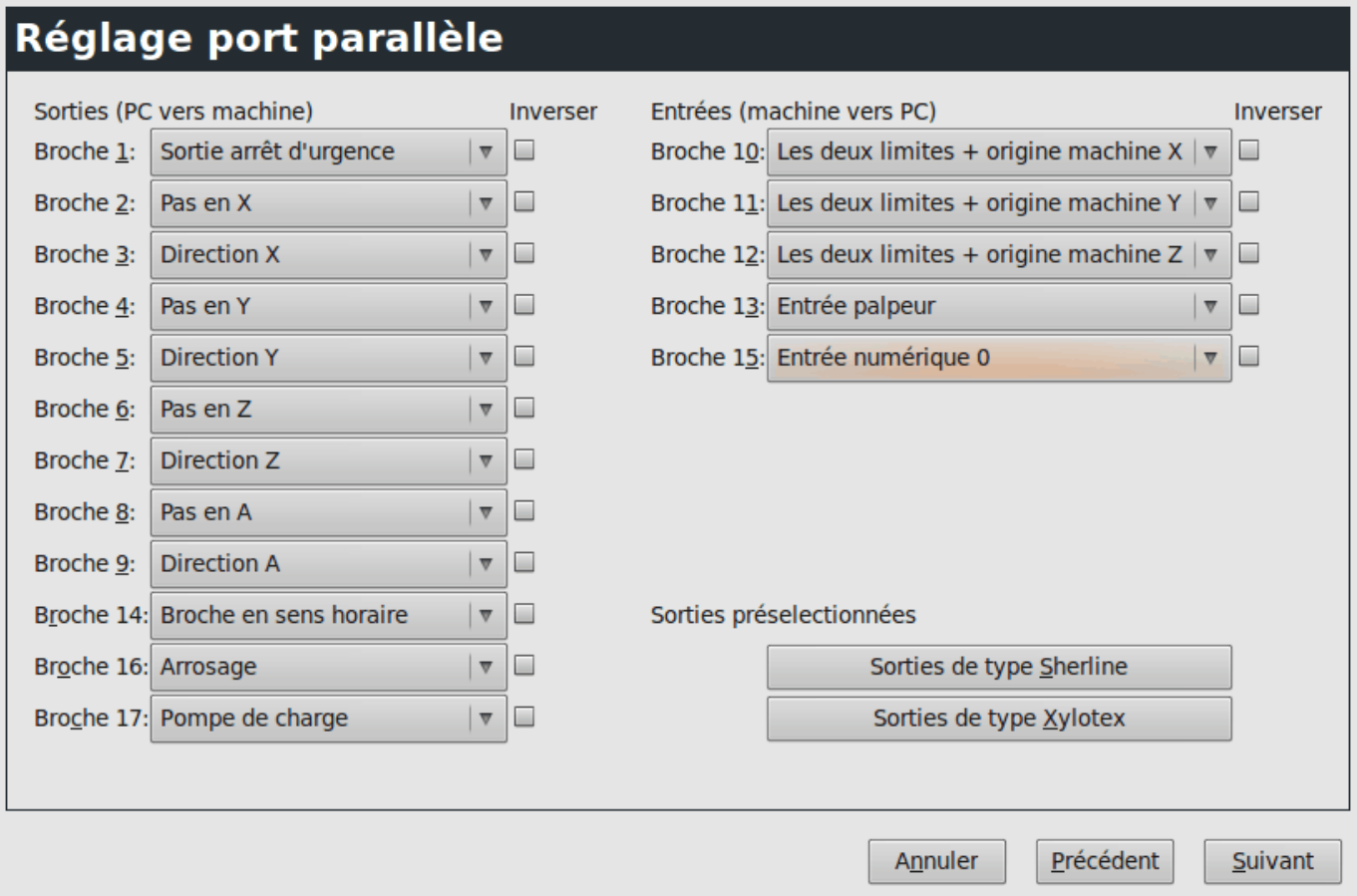


Figure 3.15: Page de réglage du port parallèle

- Sorties (PC vers machine) - Pour chacune des broches, choisir le signal correspondant au brochage entre le port parallèle et l'interface matérielle. Cocher la case inverser si le signal est inversé (0V pour vrai/actif, 5V pour faux/inactif).
- Sorties présélectionnées - Réglage automatique des pins 2 à 9 Direction sur les pins 2, 4, 6, 8, selon le type Sherline Direction sur les pins 3, 5, 7, 9, selon le type Xylotex
- Entrées et sorties - Les entrées ou les sorties non utilisées doivent être placées sur Inutilisé.
- Sortie arrêt d'urgence - Sélectionnable dans la liste déroulante des sorties. La sortie d'arrêt d'urgence est utilisée pour actionner l'organe de coupure du circuit de puissance de la machine. Le contact de cet organe est câblé en série avec les contacts des boutons d'arrêt d'urgence extérieurs ainsi qu'avec tous les contacts compris dans la boucle d'arrêt d'urgence.
- Entrées (machine vers PC) - Ces choix se font dans la liste déroulante des entrées.
- Pompe de charge - Si la carte de contrôle accepte un signal pompe de charge, dans la liste déroulante des sorties, sélectionner Pompe de charge sur la sortie correspondant à l'entrée Pompe de charge de la carte de contrôle. La sortie pompe de charge sera connectée en interne par Stepconf. Le signal de pompe de charge sera d'environ la moitié de la fréquence maxi des pas affichée sur la page des informations machine.

3.5.9 Configuration des axes

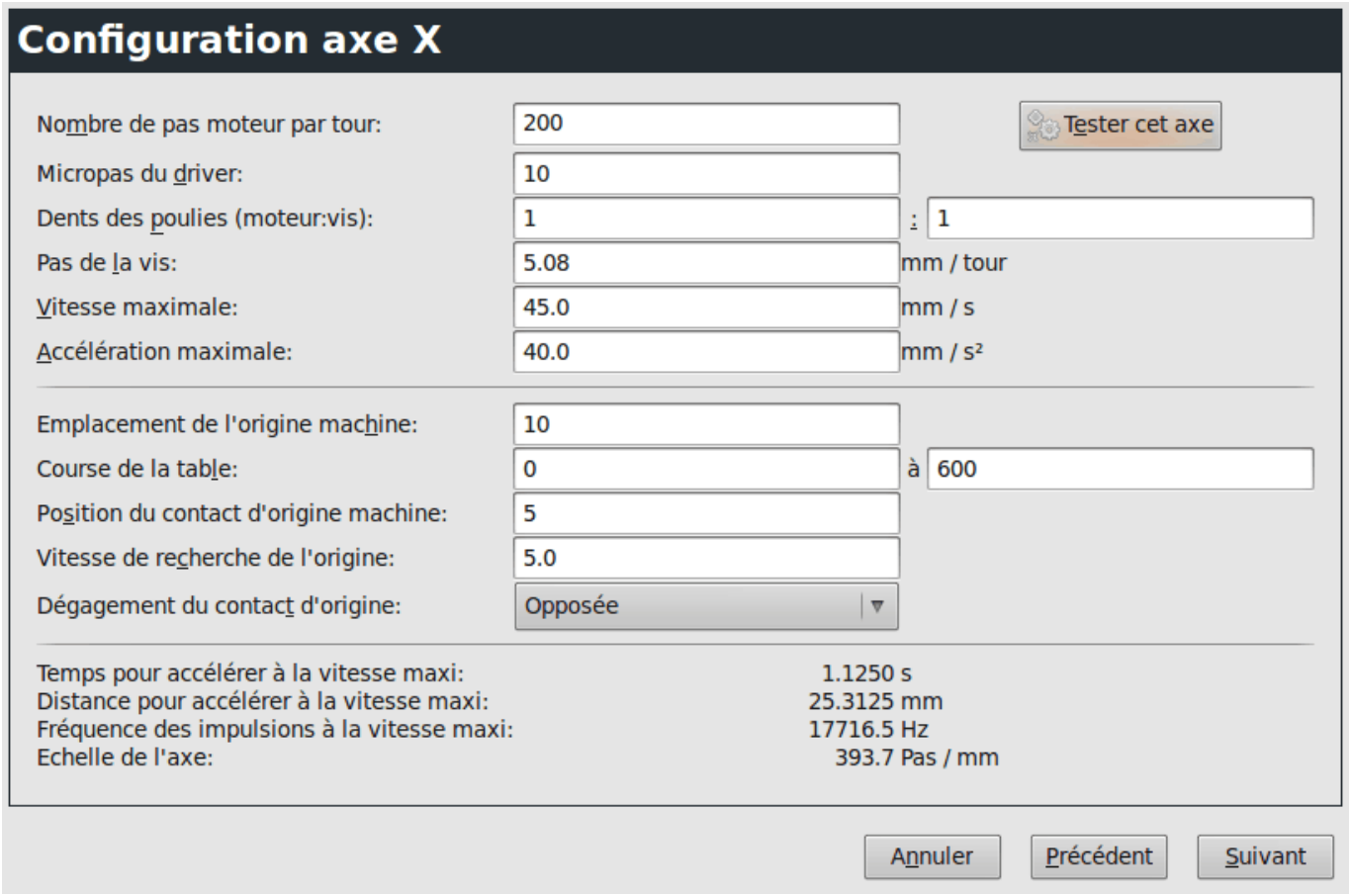


Figure 3.16: Page de configuration des axes

- Nombre de pas moteur par tour - Nombre de pas entiers par tour de moteur. Si l'angle d'un pas en degrés est connu (par exemple, 1.8 degrés), diviser 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. Entrer par exemple 2 pour le demi pas ou une des valeurs permise par le pilote du moteur.
- Dents des poulies - Si entre le moteur et la vis un réducteur poulie/courroie est présent, entrer ici le nombre de dents de chacune des poulies. Pour un entrainement direct, entrer 1:1.
- Pas de la vis - Entrer ici le pas de la vis. Si le pouce a été choisi comme unité, entrer ici le nombre de filets par pouce. Si le mm a été choisi, entrer ici le pas du filet en millimètres. Si la vis est à plusieurs filets, déterminer de combien se déplace le mobile par tour de vis et entrer cette valeur ici. Si la machine se déplace dans la mauvaise direction, entrer une valeur négative au lieu d'une positive, et vice-versa.
- Vitesse maximale - Entrer ici la vitesse de déplacement maximale de l'axe, en unités par seconde.
- Accélération maximale - Les valeurs correctes pour ces deux entrées ne peuvent être déterminées que par l'expérimentation. Consulter le calcul de la vitesse pour trouver la vitesse et le calcul de l'accélération pour trouver l'accélération maximale.
- Emplacement de l'origine machine - 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. Si des capteurs de fin de course sont utilisés pour la prise d’origine, le point d’origine ne doit pas se trouver au même coordonnées que le capteur. Une erreur de limite simultanée à l’origine surviendrait.

- Course de la table - Étendue de la course que le programme en G-code 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 cette 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 physique 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 et cela, malgré l’inertie du mobile. 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, rendre négative la valeur de Vitesse de recherche de l’origine.
- Dégagement du contact d’origine - Choisir Identique pour que la machine reparte d’abord 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. Choisir 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 en secondes, calculé en fonction des paramètres renseignés précédemment.
- Distance pour accélérer à la vitesse maxi - Distance en mm, calculée en fonction des paramètres renseignés précédemment.
- 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 blocage (La plus grande fréquence utilisable variera d’un ordinateur à un autre)
- Échelle de l’axe - Le nombre qui sera utilisé dans le fichier ini [SCALE]. C’est le nombre de pas moteur par unité utilisateur.
- Test de cet axe - Ouvre une fenêtre permettant de tester les paramètres pour chaque axe. Il est possible de modifier par expérimentation certaines données et de les reporter dans la configuration.
- Adresse du second port parallèle - Si un second port parallèle est nécessaire, entrer son adresse ici. Si les ports sont intégrés à la carte mère il est possible de vérifier dans le BIOS, habituellement 0x378 0x278 0x3bc. Attention les cartes additionnelles ont d’autres adresses. Dans ce cas, la commande lspci -v dans un terminal peux aider, si le nom du chipset de la carte est connu. Plus de détails à ce sujet sont disponibles dans le manuel de l’intégrateur.

3.5.10 Tester cet axe

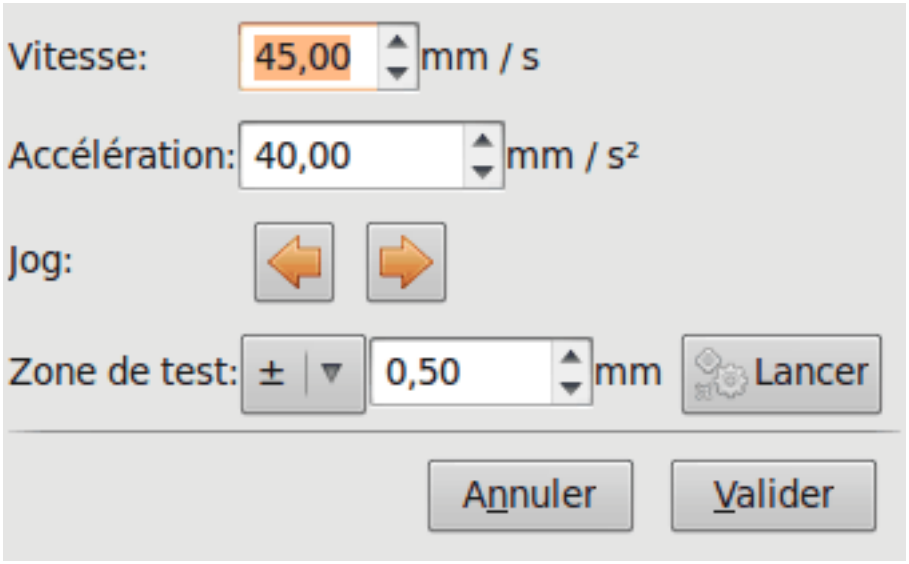


Figure 3.17: Tester cet axe

Tester cet axe et un test simple pour définir les signaux de directions et de pas, ainsi que les valeurs d'accélération et de vitesse.

Important
Pour pouvoir utiliser ce test d'axe, il sera peut-être nécessaire de valider manuellement l'axe à tester. Si le driver utilise une pompe de charge, il faudra la bi-passer pour essayer les différentes valeurs de vitesse et d'accélération.

3.5.11 Trouver la vitesse maximale

Commencer avec une faible valeur d'accélération (par exemple, **2 pouces/s²** ou **50 mm/s²**) et la vitesse que espérée. En utilisant les boutons de jog, positionner l'axe vers son centre. Il faut être prudent, car avec peu d'accélération, la distance d'arrêt peut être très surprenante. Après avoir évalué le déplacement possible dans chaque direction en toute sécurité, entrer une distance dans le champs Zone de test garder à l'esprit qu'après un décrochage, le moteur peut repartir dans la direction inattendue. Puis cliquer sur Lancer. La machine commencera à aller et venir le long de 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 au moins, sur une courte distance. La formule **d = 0.5 * v * v/a**, donne la distance minimale requise pour atteindre la vitesse de croisière. Si la sécurité est garantie, pousser sur la table dans la direction inverse du mouvement pour simuler les efforts de coupe. Si la table décroche, réduire la vitesse et recommencer le test. Si la machine ne présente aucun décrochage, cliquer sur le bouton Lancer. 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, vibre ou perd des pas, même à faible vitesse, vérifier les éléments suivants:

- Corriger les paramètres de temps des impulsions de commande.
- Le brochage du port et la polarité des impulsions. Les cases Inverser.

- La qualité des connexions et le blindage des câbles.
- Les problèmes mécaniques avec le moteur, l'accouplement moteur, vis, raideurs etc.

Quand la vitesse à laquelle l'axe ne perd plus de pas et à laquelle les mesures sont exactes pendant le test a été déterminée, réduire cette vitesse de 10% et l'utiliser comme vitesse maximale pour cet axe.

3.5.12 Trouver l'accélération maximale

Avec la vitesse maximale déterminée à l'étape précédente, entrer une valeur d'accélération approximative. Procéder 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 de test permette à la machine d'atteindre la vitesse sélectionnée. Une fois que la valeur à laquelle l'axe ne perd plus de pas pendant le test a été déterminée, la réduire de 10% et l'utiliser comme accélération maximale pour cet axe.

3.5.13 Configuration de la broche

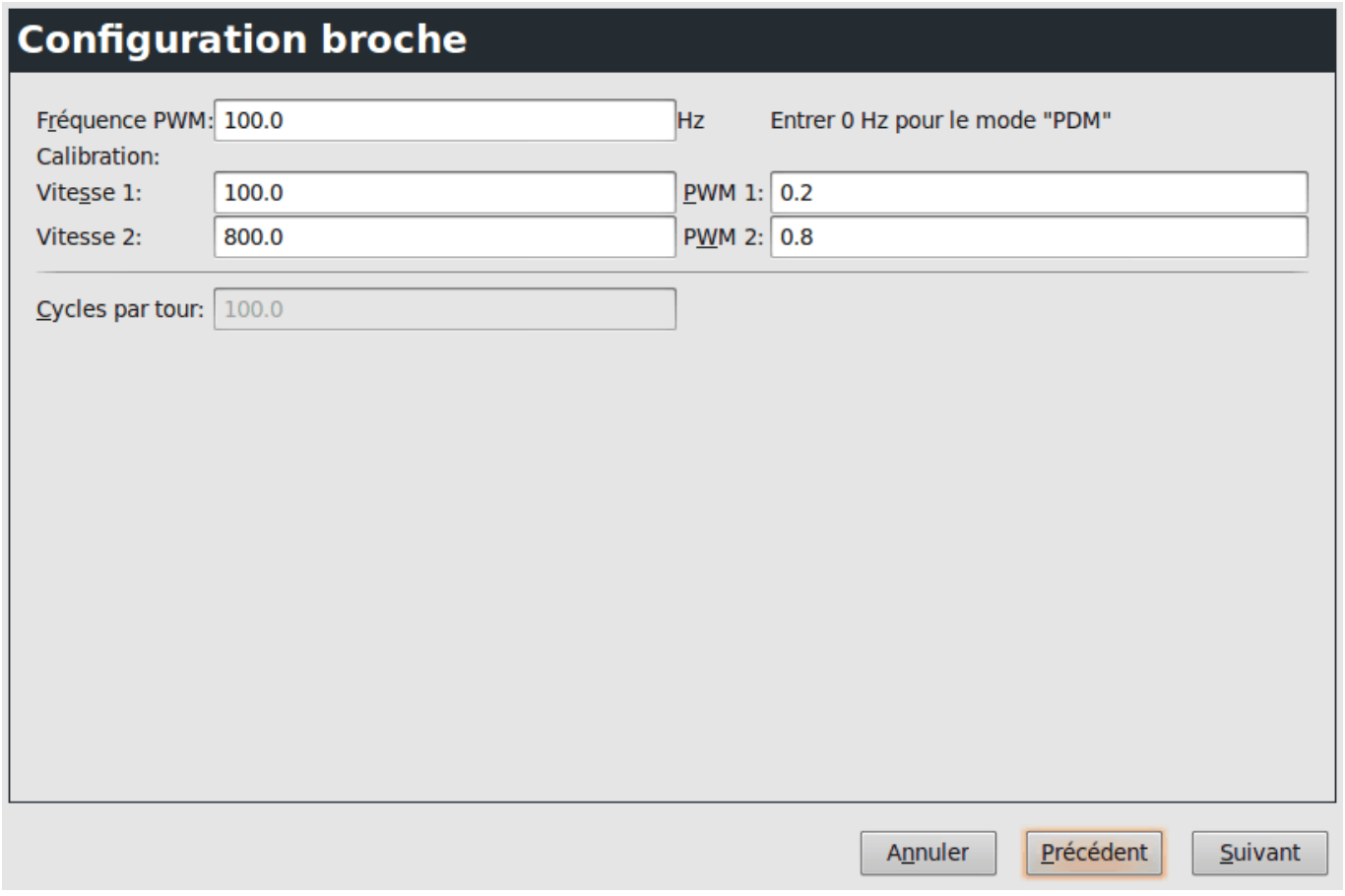


Figure 3.18: Page configuration de la broche

Ces options ne sont accessibles que quand PWM broche, Phase A codeur broche ou index broche sont configurés dans le réglage du port parallèle.

3.5.14 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. Entrer 0 pour le mode PDM (modulation de densité d'impulsions), qui est très utile pour générer une tension de consigne analogique. Se reporter à la documentation du variateur de broche pour connaître la valeur appropriée.
- Vitesse 1 et 2, PWM 1 et 2 - Le fichier de configuration généré utilise une simple relation linéaire pour déterminer la valeur PWM correspondant à une vitesse de rotation. Si les valeurs ne sont pas connues, elles peuvent être déterminées. Voir la section sur la calibration de la broche.

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

Lorsque les signaux appropriés, provenant d'un codeur de broche, sont connectés au port parallèle, LinuxCNC peut être utilisé pour les usinages avec broche synchronisée comme le filetage ou le taraudage rigide. Ces signaux son:

- Index broche - Également appelé PPR broche, c'est une impulsion produite à chaque tour de broche.
- Phase A broche - C'est une suite d'impulsions carrées générées sur la voie A du codeur pendant la rotation de la broche. Le nombre d'impulsions pour un tour correspond à la résolution du codeur.
- Phase B broche (optionnelle) - 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'accroitre l'immunité au bruit et la résolution d'un facteur 4.

Si Phase A broche et Index broche apparaissent dans le réglage des broches du port, l'information suivante doit être renseignée sur la page de configuration broche:

- Cycles par tour - Le nombre d'impulsions par tour sur la broche Phase A broche.
- La vitesse maximale en filetage - La vitesse de broche maximale utilisée en filetage. Pour exploiter un moteur de broche rapide ou un codeur ayant une résolution élevée, une valeur basse de BASE_PERIOD est requise.

3.5.16 Calibrer la broche

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

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

Finir les étapes suivantes de la configuration, puis lancer LinuxCNC avec cette configuration. Mettre la machine en marche et aller dans l'onglet Données manuelles, démarrer le moteur de broche en entrant: M3 S100. Modifier la vitesse de broche avec différentes valeurs comme: S800. Les valeurs permises vont de 1 à 1000.

Pour deux différentes valeurs de Sxxx, mesurer la vitesse de rotation réelle de la broche en tours/mn. Enregistrer ces vitesses réelles de la broche. Relancer Stepconf. Pour les Vitesses, entrer les valeurs réelles mesurées et pour les PWM, entrer la valeur Sxxx divisée par 1000.

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

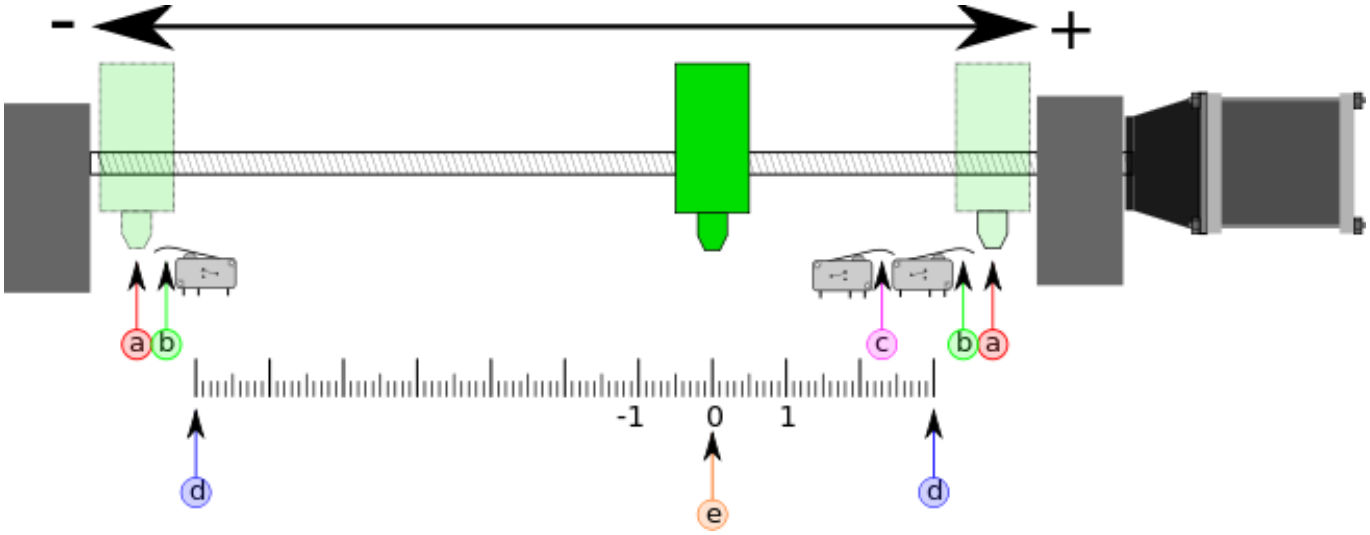
- S’assurer que les deux points de mesure des vitesses en tr/mn ne soient pas trop rapprochés
- S’assurer que les deux vitesses utilisées sont dans la gamme des vitesses utilisées généralement par la machine.

Par exemple, si la broche tourne entre 0tr/mn et 8000tr/mn, mais qu’elle est utilisée généralement entre 400tr/mn et 4000tr/mn, prendre alors des valeurs qui donneront 1600tr/mn et 2800tr/mn.

3.5.17 Terminer la configuration

Cliquer Appliquer pour enregistrer les fichiers de configuration. Ensuite, il sera possible de relancer ce programme et ajuster les réglages entrés précédemment.

3.5.18 Position des fins de course sur les axes



La course de chaque axe est bien délimitée. Les extrémités physiques d’une course sont appelées les butées mécaniques, position **(a)**.

Warning
Si une butée mécanique venait à être dépassée, la vis ou le bâti machine seraient détériorés!

Avant la butée mécanique se trouve un contact de fin de course **(b)**. Si ce contact est rencontré pendant les opérations normales, LinuxCNC 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. Ces fins de course doivent détecter le mobile sur toutes la distance d’arrêt et ne pas se réactiver à cause d’un dépassement dû à l’inertie.

Avant le contact de fin de course se trouve une limite logicielle **(d)**. Cette limite logicielle est introduite après la prise d’origine machine. Si une commande manuelle ou un programme G-code dépasse cette limite, ils ne seront pas exécutés. Si un mouvement en jog ou en manuel cherche à dépasser la limite logicielle, il sera interrompu sur cette limite.

Le contact d’origine machine **(c)** 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 **(e)** correspond au 0 de l'axe dans le système de coordonnées pièce, après que la prise d'origine pièce de cette axe ait été faite. La position zéro doit se trouver entre les deux limites logicielles pour que l'usinage soit possible. Sur les tours, le mode vitesse à 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 logicielles. En particulier, la position de l'origine ne doit jamais être égale à une limite logicielle. On place habituellement cette position au point le plus facile pour réaliser le changement d'outil.

3.5.19 Exploitation sans fin de course

Une machine peut être utilisée sans contact de fin de course. Dans ce cas, seules les limites logicielles empêcheront la machine d'atteindre les butées mécaniques. Les limites logicielles n'opèrent qu'après que la POM (prise d'origine machine) soit faite sur la machine. Puisqu'il n'y a pas de contact, la machine doit être déplacée à la main et à l'œil, à sa position d'origine avant de presser le bouton POM des axes ou le sous-menu Machine → Prises d'origines machine → POM de l'axe. L'opérateur devra cocher chacun des axes individuellement pour faire la POM de chacun d'eux.

3.5.20 Exploitation sans contact d'origine

Une machine peut être utilisée sans contact d'origine machine. Si la machine dispose de contacts de fin de course, mais pas de contact d'origine machine, il est préférable d'utiliser le contact de fin de course comme contact d'origine machine (exemple, choisir 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 toujours 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 logicielles d'être utilisables.

3.5.21 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 ensembles, selon diverse combinaisons, 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, LinuxCNC 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. Une résistance de rappel est nécessaire dans le circuit pour tirer l'entrée au nivaux haut. La valeur typique pour un port parallèle est de 47K. Une bonne sécurité utilise des contacts normalement fermés sans pièce de commande souple.

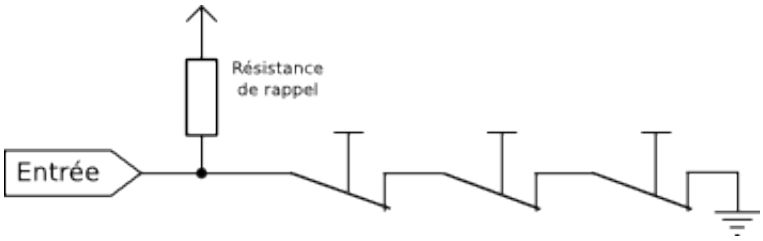


Figure 3.19: Contacts normalement fermés

Câblage de contacts NC en série (schéma simplifié)

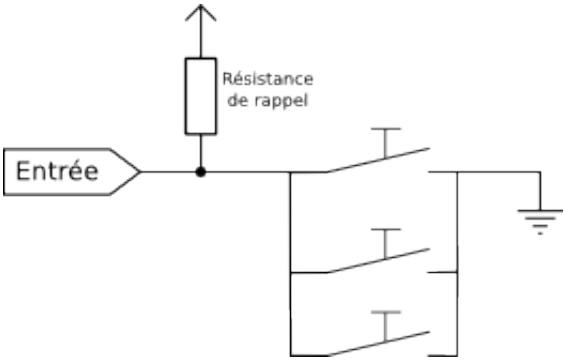


Figure 3.20: Contacts normalement ouverts

Câblage de contacts NO en parallèle (schéma simplifié)

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.
- 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.

3.6 Assistant graphique de configuration Mesa

L'assistant de configuration PNCconf couvre toute la gamme des cartes d'entrées/ sorties Mesa et jusqu'à trois ports parallèles. Il est destiné à la configuration de systèmes à servomoteurs en boucle fermée ou de systèmes à moteurs pas à pas avec pilotage matériel externe. Il utilise une approche d'assistance similaire à celle de StepConf qui est lui, utilisé pour configurer les systèmes avec pilotage logiciel des moteurs pas à pas, au travers de ports parallèles. Pour une machine n'utilisant qu'un à trois ports parallèles standards, le logiciel StepConf est un assistant mieux adapté.

L'assistant PNCconf permet de produire des configurations avancées sans connaitre quoi que ce soit de HAL.

PNCconf en est encore au stade du développement (Beta 1) il peut exister quelques bogues et manques de fonctionnalités. Merci de rapporter les bogues et les suggestions à la page du forum ou par courriel à la liste de diffusion.

Il y a deux manières d'utiliser PNCconf:

1. La première consiste à l'utiliser pour configurer le système et si, par la suite, certaines options doivent être modifiées, il suffira alors de recharger PNCconf et d'apporter les modifications aux réglages. Cela fonctionne bien si la machine est assez standard, pour les machines particulières il est possible d'ajouter à la main les nouvelles fonctionnalités. PCNConf est bien adapté pour cette utilisation.
2. La seconde consiste à l'utiliser pour construire une configuration la plus proche possible de ce qui est souhaité, puis à la modifier à la main pour l'adapter aux besoins. C'est le bon choix si les besoins de modifications vont au-delà des possibilités de PNCconf ou pour expérimenter et en apprendre plus sur LinuxCNC.

Il est possible de naviguer dans l'assistant, revenir sur des pages, annuler des choix, obtenir de l'aide et des diagrammes puis enfin, de valider la configuration par la page de sortie du programme.

NDT: Certaines divergences entre cette traduction et l'aspect réel de l'interface peuvent apparaitre pendant la phase de développement de PNCconf. Elles disparaitront quand le logiciel sera finalisé.

3.6.1 Instructions pas à pas

Démarrer le programme depuis le menu Applications → CNC → PNCconf ou depuis un terminal avec la commande:

```
pncconf
```

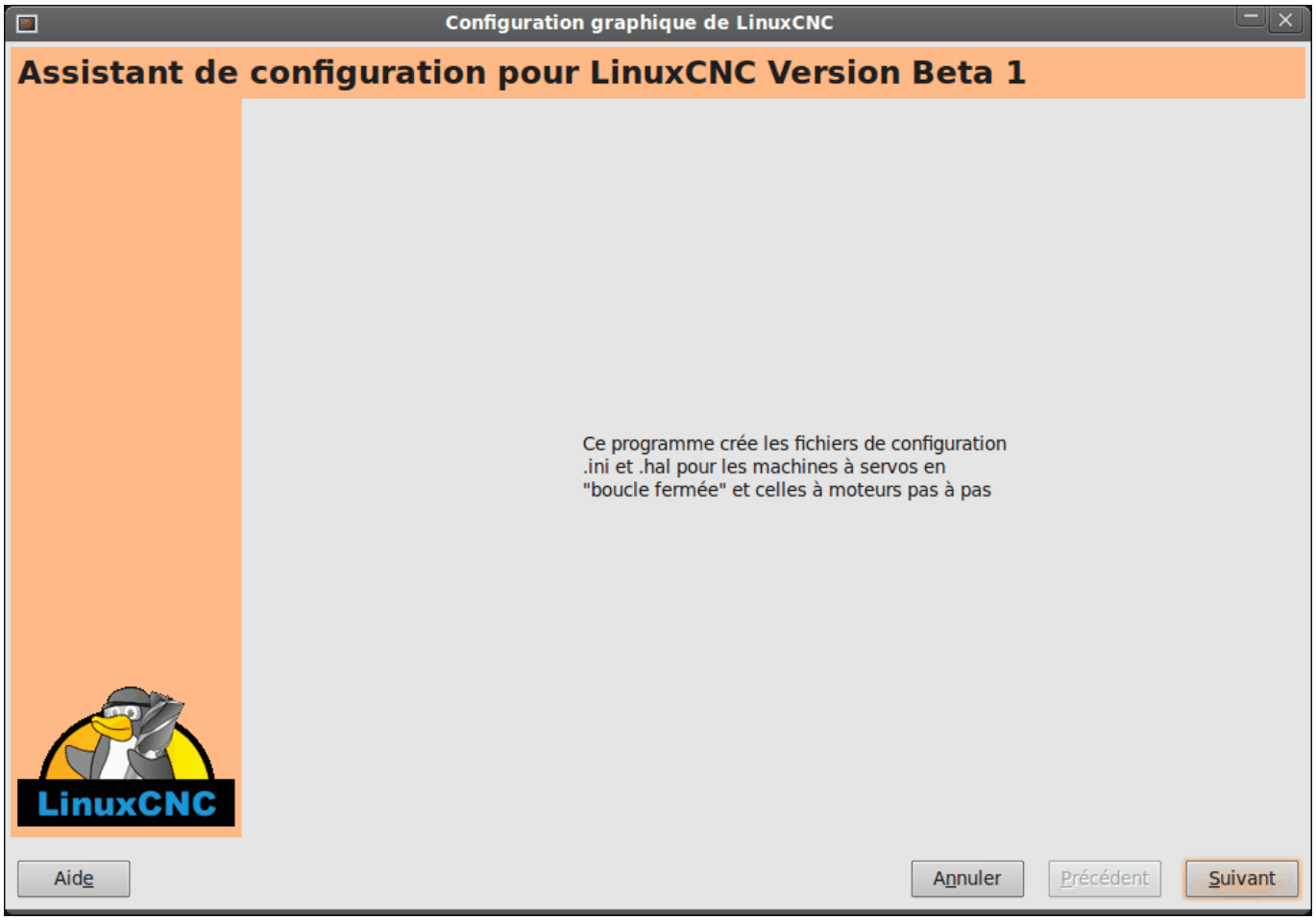


Figure 3.21: Écran d’accueil de PnCConf

3.6.2 Créer ou éditer une configuration

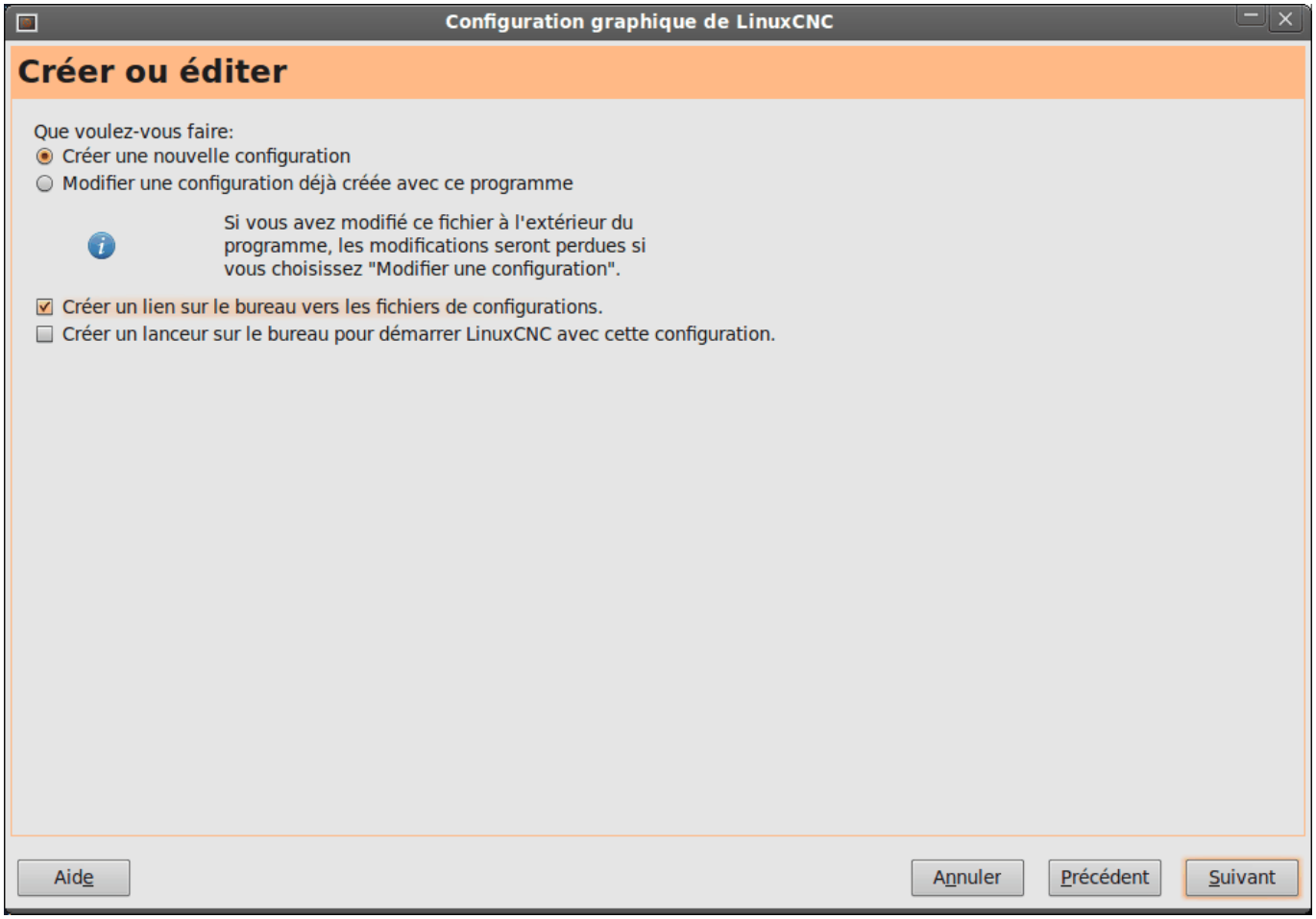


Figure 3.22: Créer ou éditer

Il est possible de créer une nouvelle configuration ou d'en modifier une existante. Si Modifier une configuration déjà créée est choisi, suivi d'un clic sur Suivant, un sélecteur de fichier apparait pour choisir la configuration existante à modifier. Par défaut, Pncconf présélectionne le dernier fichier enregistré. Il est possible de cocher les options Créer un lien sur le bureau qui créera un lien sur le bureau pointant sur ce nouveau fichier de configuration, Créer un lanceur qui créera un lanceur sur le bureau qui démarrera LinuxCNC dans cette configuration. Si ces options ne sont pas utilisées, le nouveau fichier de configuration se trouvera dans le dossier ~/linuxcnc/configs. Il est toujours possible de lancer LinuxCNC normalement et de sélectionner la configuration souhaitée dans la liste.

3.6.3 Informations machine

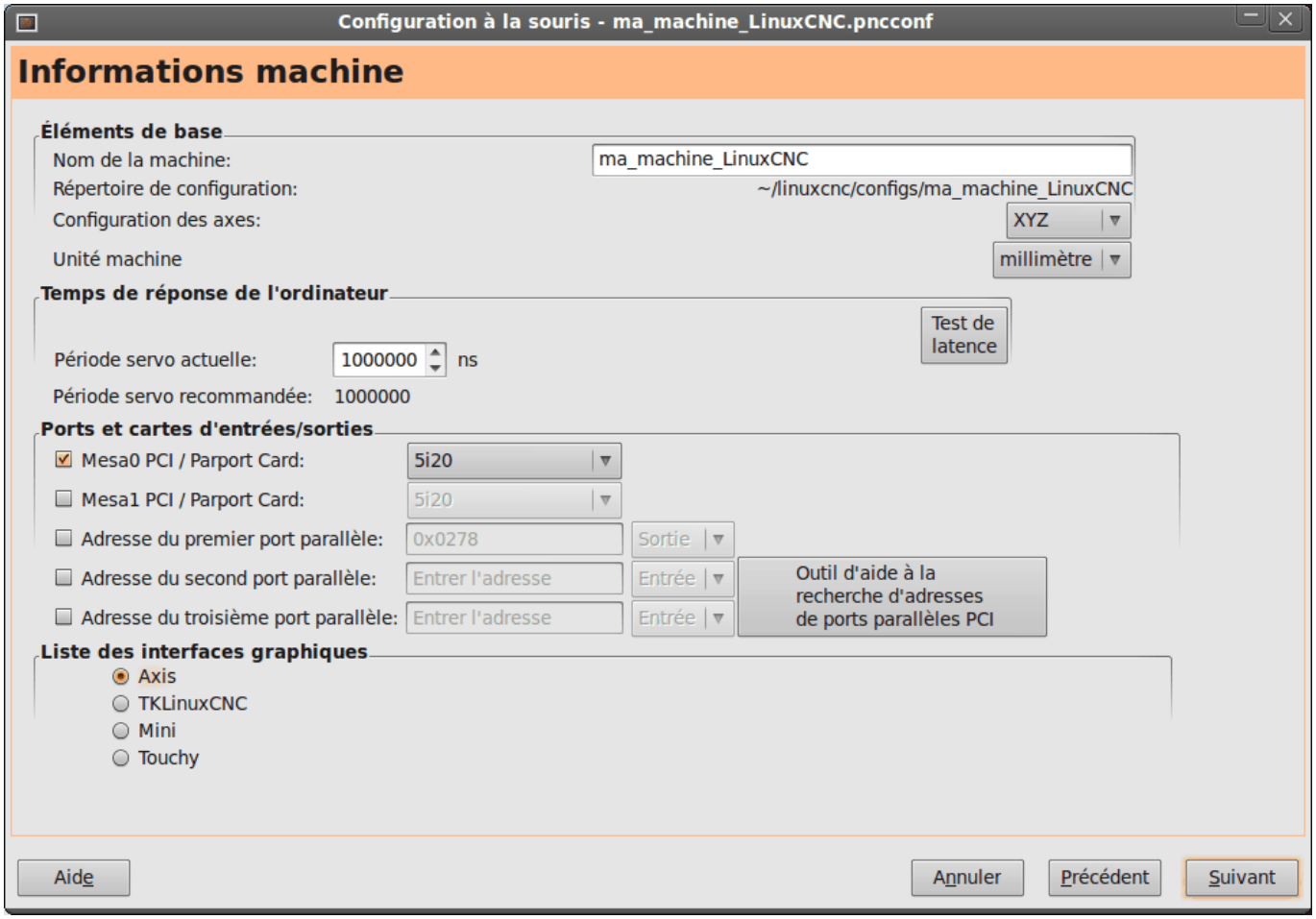


Figure 3.23: Informations machine

Éléments de base

Nom de la machine
Préciser ici le nom de la machine à configurer, les espaces dans les noms seront remplacés par des _ (en règle générale, Linux n'aime pas les espaces dans les noms de fichiers).

Configuration des axes
Cette liste déroulante précise le nombre d'axes de la machine, sélectionner selon la machine XYZ (fraiseuse 3 axes), XYZA (fraiseuse 4 axes) ou XZ (tour).

Unité machine
Définit l'unité de mesure utilisée par la machine, pouce ou millimètre, toutes les données introduites par la suite devront être données dans l'unité choisie ici.

`Les valeurs introduites par défaut dans cet assistant ne sont pas converties automatiquement dans l'unité choisie ici, bien vérifier toutes ces valeurs.`

Temps de réponse de l'ordinateur

Période servo actuelle
La période d'asservissement. C'est l'horloge du système. La latence donne la variation de cette horloge. LinuxCNC demande une chronologie serrée et cohérente, sinon des problèmes surviendront.

Quelques explications: LinuxCNC requiert et utilise un système d'exploitation temps réel, ce qui signifie qu'il a une latence très faible et un temps de réponse très court. Les évènements arrivent avec précision dans le temps quand LinuxCNC nécessite pour ses calculs, de ne pas être interrompu par des demandes de priorité inférieure (interruptions) comme des saisies au clavier ou des demandes d'affichage.

Le test de latence est très important, il est un élément clef qui doit être effectué au plus tôt. Heureusement les cartes Mesa se chargent des tâches critiques en temps de réponse, comme le comptage d'impulsions, la génération de PWM et cela leur permet de supporter une latence supérieure à celle d'un système utilisant les ports parallèles de la carte mère.

Le test standard dans LinuxCNC, consiste à vérifier la latence de base du PC. Un appui sur le bouton Test de latence lancera le test de latence, il est également possible de le lancer depuis le menu application → cnc → latency test. Une fenêtre s'ouvre dans laquelle s'affichent les temps mesurés. Ce test doit fonctionner plusieurs minutes, en fait, le plus longtemps possible. 15 minutes est un minimum. Pendant le test, essayer d'utiliser le plus possible l'ordinateur, le réseau, le port USB, les disques durs, l'affichage. Observer et noter si une action particulière dégrade le temps de latence. A la fin, il sera possible de connaitre la base period jitter, la latence de base. Une valeur en dessous de 20000 est excellente et permet une génération rapide des impulsions de pas avec cette machine.+ 20000 à 50000 est assez bon pour la génération de pas.
50000 à 100000 ce n'est pas très bon mais la machine peu encore servir pour la génération de pas avec une carte ayant des temps de réponse courts.
Plus grand que 100000, la machine n'est pas utilisable pour cette fonction

Si la latence est médiocre ou si des problèmes intermittents surviennent régulièrement il sera toujours possible de l'améliorer.

Tip
Il y a une liste d'équipements et de leurs temps de latence sur le wiki de LinuxCNC
SVP, pensez à ajouter vos infos à la liste. Sur cette page il y a des liens vers des informations pour résoudre certains problèmes de latence.

Maintenant que nous avons un temps de latence acceptable nous devons choisir une période d'asservissemen
(Période servo actuelle). Dans la plupart des cas une période d'asservissement de 1000000ns est bonne, cela donne un taux de calcul de 1 kHz soit 1000 calculs par seconde. Si le système d'asservissement est construit en boucle fermée avec contrôle de couple (courant) plutôt que de vitesse (tension) le taux sera meilleur, quelque chose comme 5000 calculs par seconde (5 kHz). Le problème avec l'abaissement de la période, c'est qu'elle laisse moins de temps disponible à l'ordinateur pour faire d'autres choses. Typiquement la réponse de l'affichage (GUI) est moins bonne. Il faut choisir un équilibre. Garder à l'esprit que sur un mécanisme en boucle fermée, une modification de la période d'asservissement nécessitera de réajuster l'ensemble des paramètres de la boucle.

Ports et cartes d'entrées/sorties

PNCconf est capable de configurer une machines avec deux cartes Mesa et trois ports parallèles. Les ports parallèles ne sont utilisables que pour des actions simples et peu rapide.

Mesa
Au moins une carte Mesa doit être choisie. PNCconf ne peut pas configurer les ports parallèles pour des codeurs, des signaux de pas ou pour la génération de signaux PWM. La liste de sélection des cartes Mesa présentes dans la liste de sélection est construite selon les micros logiciels des cartes trouvées sur le système. Il existe des options permettant d'ajouter des micros logiciels

personnalisés ou pour ignorer (blacklister) certaines versions de micros logiciels ou certaines cartes, en utilisant un fichier de préférences. Si aucune carte n'est détectée PNCconf affichera un avertissement et utilisera des valeurs par défaut mais aucun test ne sera possible. Il faut noter que, si plusieurs cartes Mesa sont utilisées, il n'existe aucun moyen de déterminer laquelle sera la carte N°0 ou N°1 et il sera indispensable de le tester. Déplacer les cartes dans les ports PCI, peut changer leur ordre. Si la configuration est créée pour deux cartes, elles doivent être installées pour que les tests fonctionnent.

Ports parallèles

Jusqu'à 3 ports parallèles, appelés parports par Mesa, peuvent être utilisés comme de simples entrées sorties. L'adresse du port parallèle doit être définie. Il est possible soit d'entrer le N° du port parallèle selon le système de numérotation de Linux 0, 1 ou 2 ou, d'entrer l'adresse réelle en hexadécimal. Les adresses des ports parallèles intégrés à la carte mère son le plus souvent aux adresses 0x0378 et 0x0278, elles peuvent être trouvées dans la configuration du BIOS. Le Bios s'ouvre en enfonçant une touche du clavier au tout début du cycle de démarrage de l'ordinateur, souvent (Del ou F2) se reporter au document de la carte mère. Sur une des pages du BIOS, il est possible de choisir l'adresse des ports parallèles et de définir leurs modes de fonctionnement comme SPP, EPP, etc, sur certains ordinateurs cette information est affichée pendant quelques secondes lors du démarrage du PC. Pour les ports parallèles sur carte PCI les adresses sont trouvées en cliquant sur le bouton Outil d'aide à la recherche d'adresse de ports parallèles qui affichera la liste des périphériques PCI découverts. Dans cette liste, se trouvera une référence aux ports parallèles avec une liste d'adresses. Une de ces adresses doit fonctionner. Noter que tous les ports parallèles PCI ne fonctionnent pas correctement en EPP. Chaque port peut être sélectionné comme Entrée pour augmenter le nombre d'entrées sur ce port ou Sortie pour un maximum de sorties. Par défaut, les ports parallèles sont configurés avec leurs broches 2 à 9 en Sortie.

Liste des interfaces graphiques

Spécifie les interfaces utilisateur graphiques que LinuxCNC peut utiliser. Chacune dispose d'options particulières.

AXIS

- Supporte les tours.
- C'est l'interface la plus utilisée et la plus développée.
- Elle est conçue pour être utilisée à la souris est avec un clavier.
- Elle est basée sur tkinter et intègre donc PYVCP (contrôle visuel python).
- Elle dispose d'un affichage graphique en 3D.
- Elle est intégrable sur les barres de tâches ou sur le bureau.

TOUCHY

- Touchy est une interface conçue pour les écrans tactiles.
- Elle ne nécessite que quelques interrupteurs physiques et une manivelle de jog.
- Elle nécessite les boutons Départ cycle, Abandon, Marche par pas.
- Elle nécessite également un bouton sélecteur d'axe sur le jog.
- Elle est basée sur GTK et intègre naturellement GladeVCP (création de panneaux de contrôle).
- Elle permet d'intégrer les panneaux de contrôle virtuels (VCP).
- Elle n'a pas de fenêtre de suivi du parcours d'outil.

- L'aspect peut être modifié avec des thèmes personnalisés.

*MINI_

- Est fourni en standard sur les machines Sherline.
- N'utilise pas d'arrêt d'urgence (ESTOP).
- Pas de possibilité d'intégrer un panneau de contrôle.

TkLinuxCNC

- Contraste élevé grâce à un fond bleu.
- Fenêtre graphique séparée.
- Pas d'intégration de panneau de contrôle possible.

3.6.4 Contrôles externes

Cette page permet de sélectionner des contrôles externes pour la commande manuelle de déplacement des axes (jog) ou des curseurs des correcteurs de vitesse.

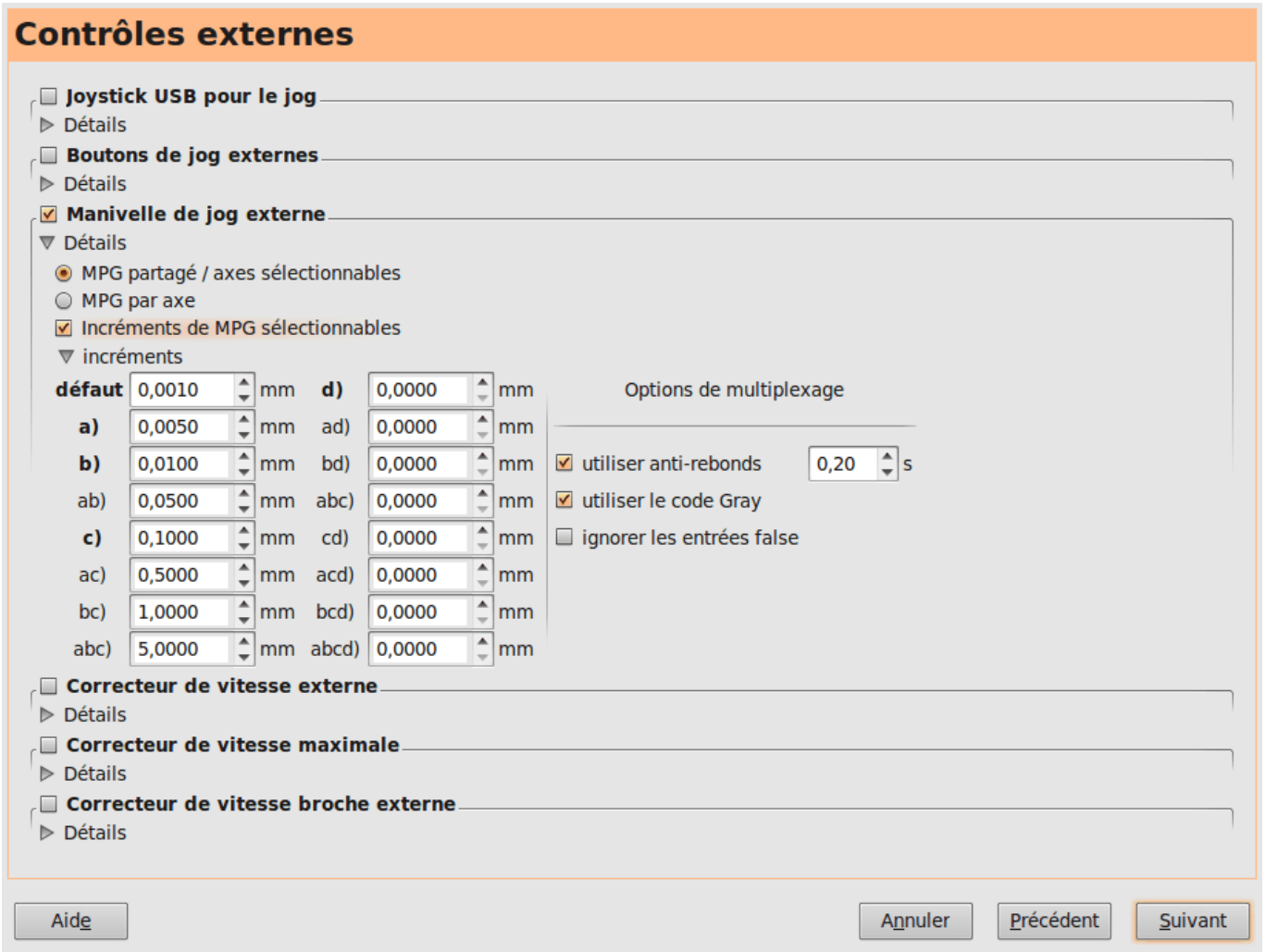


Figure 3.24: Contrôles externes

Si une manette de jeu externe est sélectionnée pour le jog, il faudra toujours la connecter à LinuxCNC avant de démarrer celui-ci. Si la manette est analogique il faudra probablement ajouter du code personnalisé à HAL. Les manivelles de jog à vernier et micro impulsion nécessitent d'être connectées à une carte Mesa sur un compteur de codeur. Pour les correcteurs de vitesses externe il est possible d'utiliser un mécanisme à générateur d'impulsions ou à commutation comme un commutateur rotatif. Les boutons externes peuvent être ceux d'une manette de jeu.

Joystick USB pour le jog
Demande des règlages spécifiques personnalisés pour être installé dans le système. Il s'agi d'un fichier qui est utilisé par LinuxCNC pour se connecter à la liste des périphériques Linux. PNCconf aidera à la construction de ce fichier.

- Ajouter règle dispositif: s'utilise pour configurer un nouveau périphérique en suivant les instructions. Le périphérique doit être branché et disponible.
- test dispositif: permet de charger un périphérique, d'afficher les noms de ses broches et de visualiser ses fonctions avec l'outil halmeter.
- Rechercher règles pour le dispositif: va rechercher les règles dans le système, utilisable pour trouver le nom des périphériques déjà construits avec PNCconf.

Les manettes de jeu utilisées en jog utilisent HALUI et le composant hal_input.

Boutons de jog externes
Permet le jog de l'axe avec de simples boutons à une vitesse spécifiée. Probablement mieux adapté pour le jog en vitesse rapide.

Manivelle de jog externe
Permet d'utiliser un générateur d'impulsions manuel pour faire du jog sur les axes de la machine. Les manivelles à impulsions (MPG) sont souvent présentes sur les machines de bonne qualité. Elles délivrent en sortie des impulsions en quadrature qui peuvent être comptées avec un compteur de codeur MESA. PNCconf gère une manivelle par axe ou une manivelle partagée entre les axes. Il permet la sélection des vitesses de jog en utilisant des commutateurs rotatifs. L'option de sélection des incréments de jog utilise le composant mux16. Ce composant dispose d'options telles que l'anti-rebond et l'utilisation du code Gray pour filtrer l'entrée physique du commutateur.

Correcteurs de vitesses
PNCconf permet de modifier les vitesses d'avances ou de broche en utilisant une manivelle à micros impulsions ou un commutateur rotatif. Les incréments sont configurables.

3.6.5 Configuration des GUI

Ici il est possible de configurer l'interface graphique utilisateur (GUI), lui ajouter des panneaux de commande virtuels (VCP) et définir certaines options d'LinuxCNC.

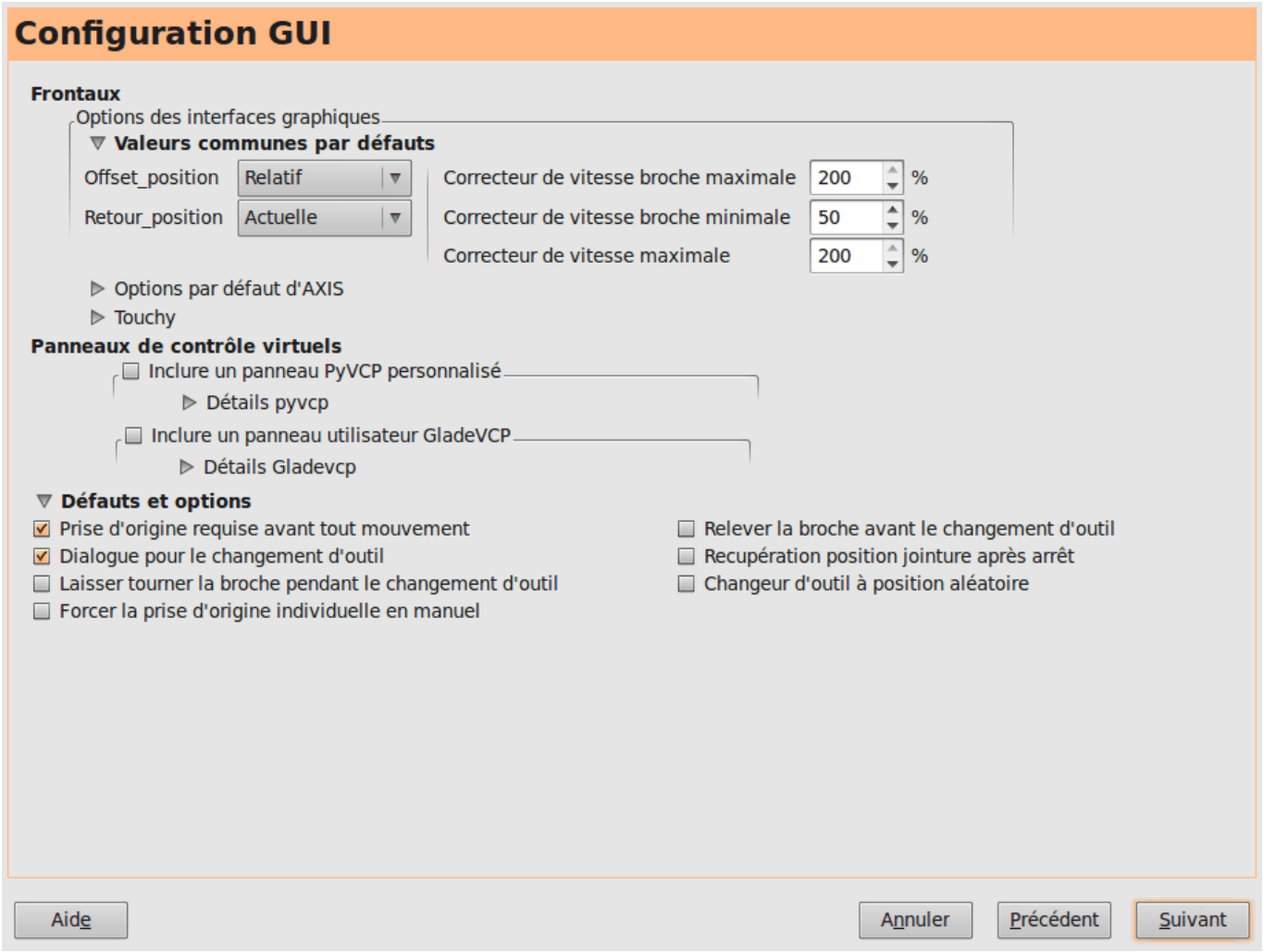


Figure 3.25: Configuration des GUI

Options des interfaces graphiques

Valeurs communes par défaut
Permet de fixer des valeurs générales par défaut, communes à toutes les interfaces graphiques.

Options par défaut d'AXIS
Ici se trouve les options spécifiques à AXIS. Si une des options Taille, Position ou Forcer à maximiser et choisie, il sera possible de modifier les valeurs de vitesse minimale ou maximale, le choix de l'éditeur de fichiers, la géométrie de la machine affichée. Ensuite, PNCconf demandera si il peut écraser le fichier de préférences (.Axisrc). Ce qui écrasera les données qui aurait été ajoutées extérieurement dans ce fichier.

Touchy
Ici se trouve les options spécifiques à Touchy. La plupart des options de Touchy peuvent être modifiées dans la page des préférences de l'application même quand elle est en marche. Touchy utilise GTK pour dessiner son écran, et supporte les thèmes GTK. Les thèmes modifient l'apparence et l'ergonomie du programme. il est possible de télécharger des thèmes depuis le net ou de les modifier soit-même. Il y a déjà une liste des thèmes utilisables sur le système. PNCconf permet de modifier facilement le thème par défaut.

Panneaux de contrôle virtuels

Les panneaux de contrôle virtuels permettent d'ajouter des contrôles et des afficheurs personnalisés. AXIS et Touchy peuvent intégrer ces contrôles dans une zone déterminée de leur écran. Il y a deux sortes de panneaux de contrôle (VCP), pyVCP qui utilise Tkinter pour dessiner l'écran ou GLADE VCP qui utilise GTK.

Panneau PyVCP
: PyVCP est un écran construit par un fichier XML. Il ne peut pas être construit à la main. Les PyVCP s'intègrent naturellement avec AXIS car ils utilisent tous les deux Tkinter. Des HAL pins sont créées pour que l'utilisateur puisse les connecter dans son fichier HAL personnalisé. Il existe par exemple, un tachymètre pour la vitesse de broche ou un panneau de boutons XYZ pour le jog, l'utilisateur peut les utiliser tel quel ou les reconstruire à son gout. Sélectionner un fichier vide où les contrôles (widgets) personnels seront enregistrés ou sélectionner un des modèles d'affichage prêts à l'emploi, PCCcong établira alors lui-même les bonnes connexions avec HAL. Si AXIS est utilisé, le panneau sera intégré sur le côté droit. Si AXIS n'est pas utilisé, le panneau sera distinct de l'écran frontal. Il est possible d'utiliser les options de géométrie et de dimensions et de déplacer le panneau, par exemple si le système le permet vers un second écran. Si le bouton Ouvrir un panneau simple est pressé, les données de géométrie et de dimensions seront utilisées et le panneau affiché.

Panneau GladeVCP
: GladeVCP s'intègre naturellement à l'intérieur de l'écran TOUCHY car ils utilisent tous les deux GTK pour leurs interfaces, mais en modifiant le thème de GladeVCP il se fond très bien dans AXIS. Il utilise un éditeur graphique pour créer ses fichiers XML. Des HAL pins sont créées, que l'utilisateur pourra connecter dans son fichier HAL personnalisé. GladeVCP permet aussi une interaction de programmation beaucoup plus sophistiquée et compliquée, ce qui n'est actuellement pas possible par PNCconf. Voir le chapitre sur GladeVCP et la création d'interfaces graphiques

PNCconf propose des exemples de panneaux à utiliser tel quel ou à reconstruire. Avec PNCconf, GladeVCP permettra de sélectionner différentes options d'affichage sur le modèle. Sous Echantillon d'options sélectionner les options souhaitées. Les boutons de zéro utilisent des commandes HALUI qui pourront être modifiées ultérieurement dans la section HALUI. Le bouton Toucher Z automatique nécessite le programme Touch-off de classicladder et que l'entrée de sonde soit sélectionnée. Il faut aussi un palpeur qui peut être réalisé avec une plaque conductrice reliée à la masse. Pour avoir une idée sur la façon dont cela fonctionne, voir:

Sous Options d'affichage, les options de géométrie et de dimensions permettent de déplacer le panneau, par exemple vers un second écran, si le système le permet. Sélectionner un thème GTK pour définir l'aspect du panneaux. En général, on le souhaite identique à l'aspect de l'écran frontal. Le panneau créé et ses options seront visibles en appuyant sur le bouton Ouvrir un panneau simple. GladeVCP placé sur l'écran frontal permet de sélectionner la position du panneau sur celui-ci. Il peut fonctionner de manière autonome ou avec AXIS, il peut être au centre ou sur le côté droit, avec Touchy il peut être au centre.

Défauts et options

Prise d'origine requise avant tout mouvement
: Pour pouvoir déplacer la machine sans passer par une recherche du point d'origine machine décocher la case. Dans ce cas la plus grande vigilance est nécessaire pour ne pas percuter une limite.

Dialogue pour le changement d'outil
: Permet le choix entre l'utilisation d'un dialogue de changement d'outil et l'exportation d'un signal standard pour utiliser un changeur d'outils automatique externe et la table d'outils.

Laisser tourner la broche pendant le changement d'outil
: Laisse tourner la broche pendant le changement d'outil. Utile pour les tours.

Forcer la prise d'origine individuelle en manuel
: Oblige à effectuer la prise d'origine individuelle de chaque axe en manuel.

Relever la broche avant le changement d'outil
Met la broche en position haute avant le changement d'outil.

Récupérer position jointure après arrêt
Mémorise la position des articulations lors de l'arrêt. Utilisé pour les machines a cinématique complexe.

Changeur d'outil à position aléatoire
Utilisé pour les changeurs d'outils qui ne reçoivent pas toujours les outils au mêmes emplacements. Des codes HAL doivent être ajoutés pour le support de ces changeurs d'outils.

3.6.6 Configuration Mesa

Les pages de configuration Mesa permettent d'utiliser les différents micros logiciels. Sur la page de configuration, si une carte Mesa a été sélectionnée, ici s'effectue le choix du micro logiciel parmi ceux disponibles, puis le choix et le paramétrage des composants nécessaires à la machine.

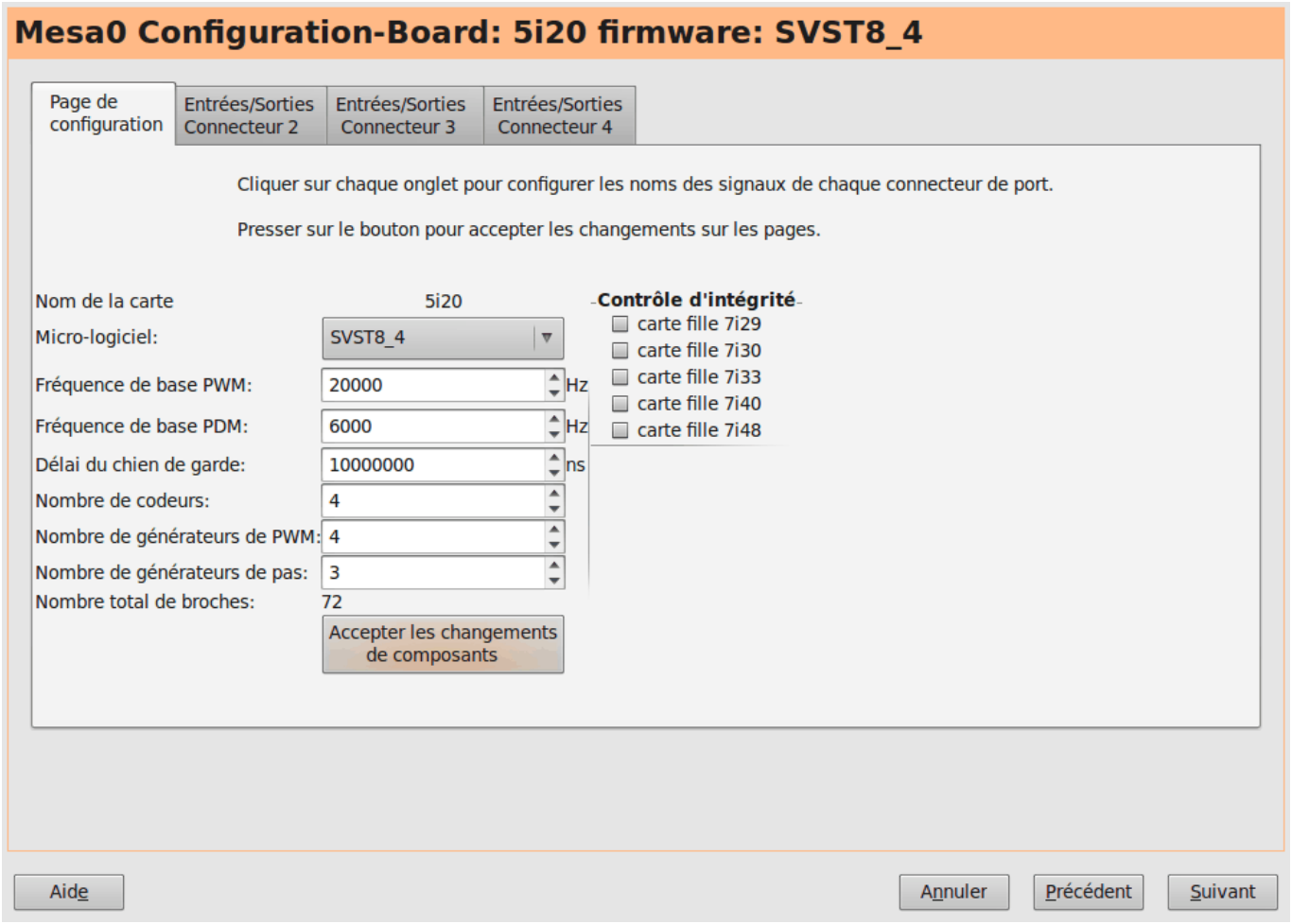


Figure 3.26: Configuration Mesa

Adresse du port parallèle MESA
Un port parallèle est utilisé seulement avec la carte Mesa 7i43. Les ports parallèles sur la carte mère ont généralement les adresses 0x378 et 0x278 il est possible de trouver l'adresse sur la

page du BIOS. Le 7i43 nécessite de programmer le port parallèle dans le mode EPP, encore une fois cela se configure dans la page du BIOS. Si un port parallèle sur carte PCI est utilisé, les adresses peuvent être recherchées en utilisant le bouton de recherche sur la page de base de PNCConf.

Important
Noter que beaucoup de cartes PCI ne prennent pas en charge le protocole EPP correctement.

Fréquence de base PWM, PDM et 3PWM
[5] Règle l'équilibrage entre entrainement et linéarité. Si des cartes filles Mesa sont utilisées, les documents de celles-ci devraient donner des recommandations. Il est important de les suivre pour éviter des dommages et obtenir les meilleures performances.

Par exemple....

- La carte 7i33 demande un PDM et une fréquence de base de 6 mHz.
- La carte 7i29 demande un PWM et une fréquence de base de 20 Khz.
- La carte 7i30 demande un PWM et une fréquence de base de 20 Khz.
- La carte 7i40 demande un PWM et une fréquence de base de 50 Khz.
- La carte 7i48 demande un PWM et une fréquence de base de 24 Khz.

Délai du chien de garde
Définit le délai durant lequel la carte Mesa va attendre avant de déconnecter les sorties si la communication est interrompue avec l'ordinateur. Les carte Mesa utilisent sur ce contact un niveau actif bas ce qui signifie que lorsque la sortie est activée son niveau logique est à 0 et si la sortie est inactive son niveau logique est à 1 soit environ 5 volts. S'assurer que l'équipement est en sécurité quand le chien de garde est déclenché.

Nombre de codeurs , Nombre de générateur de PWM , Nombre de générateur de PAS
Il est possible de choisir les composants en dé-sélectionnant ceux qui sont inutilisés. Les types de composants disponibles varient selon le micro logiciel et les cartes installées. Si des composants ne sont pas sélectionnés, des broches GPIO seront gagnées. Si des cartes filles sont utilisées, garder à l'esprit que les pins que les cartes utilisent ne doivent pas être dé-sélectionnées. Par exemple, certain micros logiciels supportent deux cartes 7i33, si une seule est installée, il est possible de dé-sélectionner assez de composants non nécessaires pour utiliser le connecteur qui était prévus pour la seconde 7i33. Les composants sont dé-sélectionnés numériquement en commençant par le plus grand nombre d'abord, puis en descendant sans en sauter. Si en faisant cela, les composants ne sont pas là où il devraient, alors il faut utiliser un micro logiciel différent. Le micro logiciel dicte où, quoi et les nombre maximum de composants. Un micro logiciel personnalisé est possible en le demandant gentiment aux développeurs LinuxCNC et Mesa. Les micros logiciels dans PNCconf nécessitent des procédures spéciales et ce n'est pas toujours possible. Bien que nous essayons de rendre PNCconf aussi souple que possible. Après avoir choisi toutes les options, appuyer sur le bouton Accepter le changement de composants et PNCconf mettra à jour les pages de configuration des E / S. Seuls les onglets nécessaires seront affichés pour les connexions disponibles, selon les documents de Mesa.

3.6.7 Réglages des E/S Mesa

Les onglets sont utilisés pour configurer les broches d'entrée et de sortie des cartes Mesa. PNCconf permet de créer des noms de signaux personnalisés à utiliser dans les fichiers de HAL personnalisés.

[5] PDM: acronyme de Modulation de Densité d'Impulsions, PWM: acronyme de Modulation de Largeur d'Impulsions

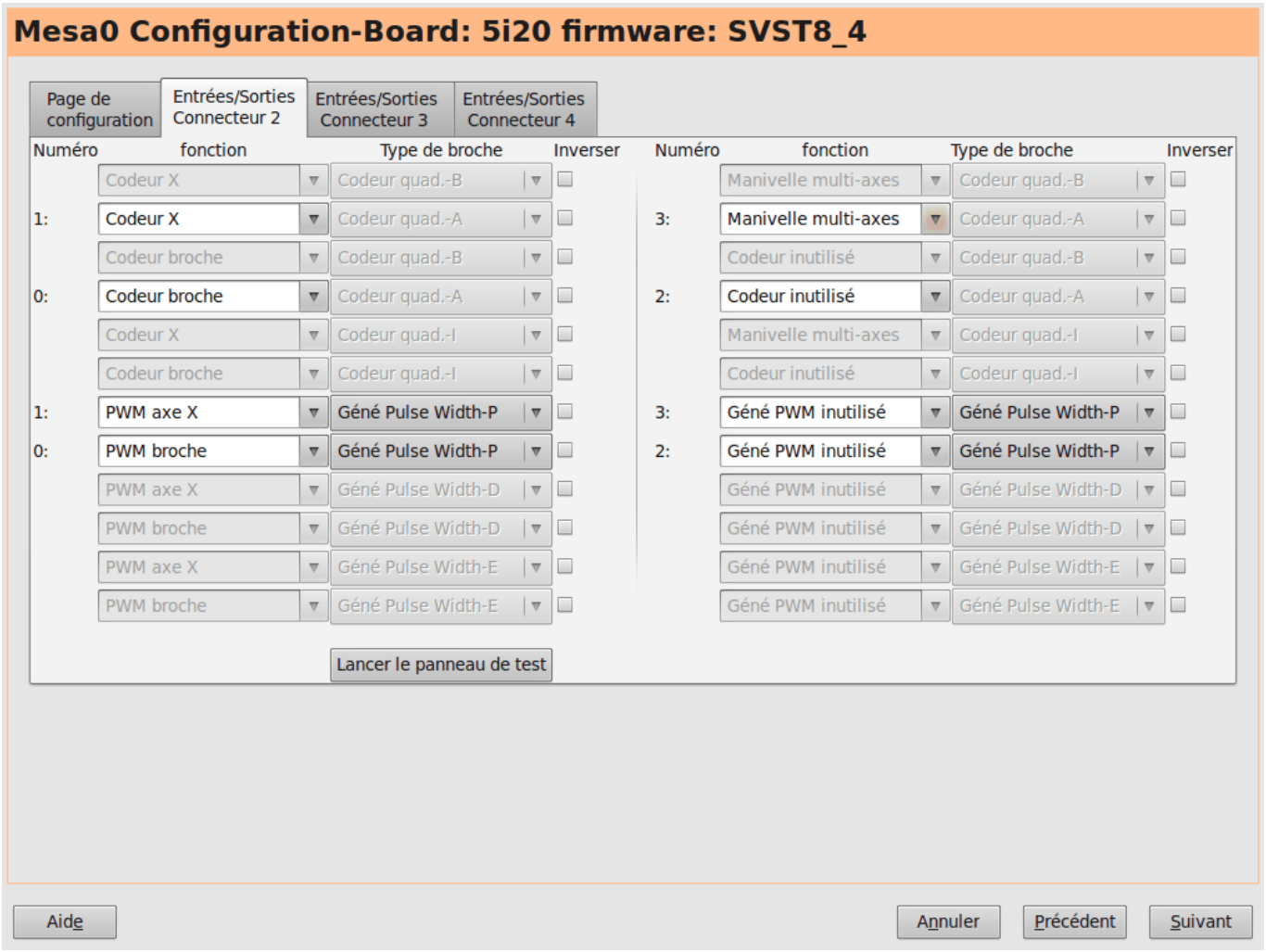


Figure 3.27: Réglages des E/S Mesa C2

Sur cet onglet, avec ce micro logiciel, les composants sont liés à l'installation d'une carte fille 7i33, généralement utilisée avec des servomoteurs en boucle fermée. Noter que les numéros de composant des codeurs, des compteurs et des pilotes PWM ne sont pas dans l'ordre numérique. Cela fait suite aux exigences de l'architecture des cartes filles.

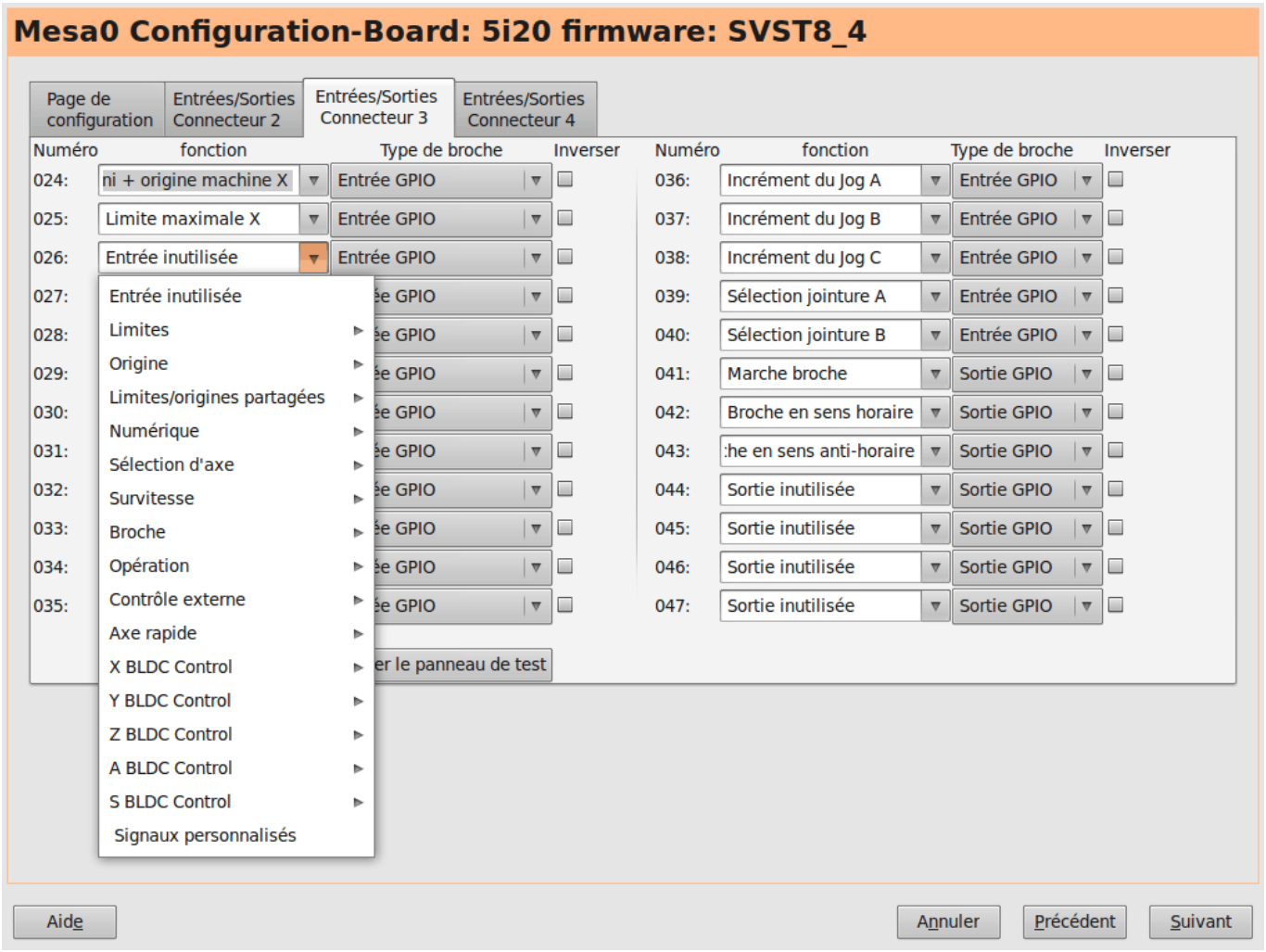


Figure 3.28: Réglages des E/S Mesa C3

Sur cet onglet, il n'y a que des broches GPIO. Noter les numéros à trois chiffres, ils correspondent au numéros des HAL pins. Les broches GPIO peuvent être sélectionnées comme des entrées ou des sorties et elles peuvent être inversées.

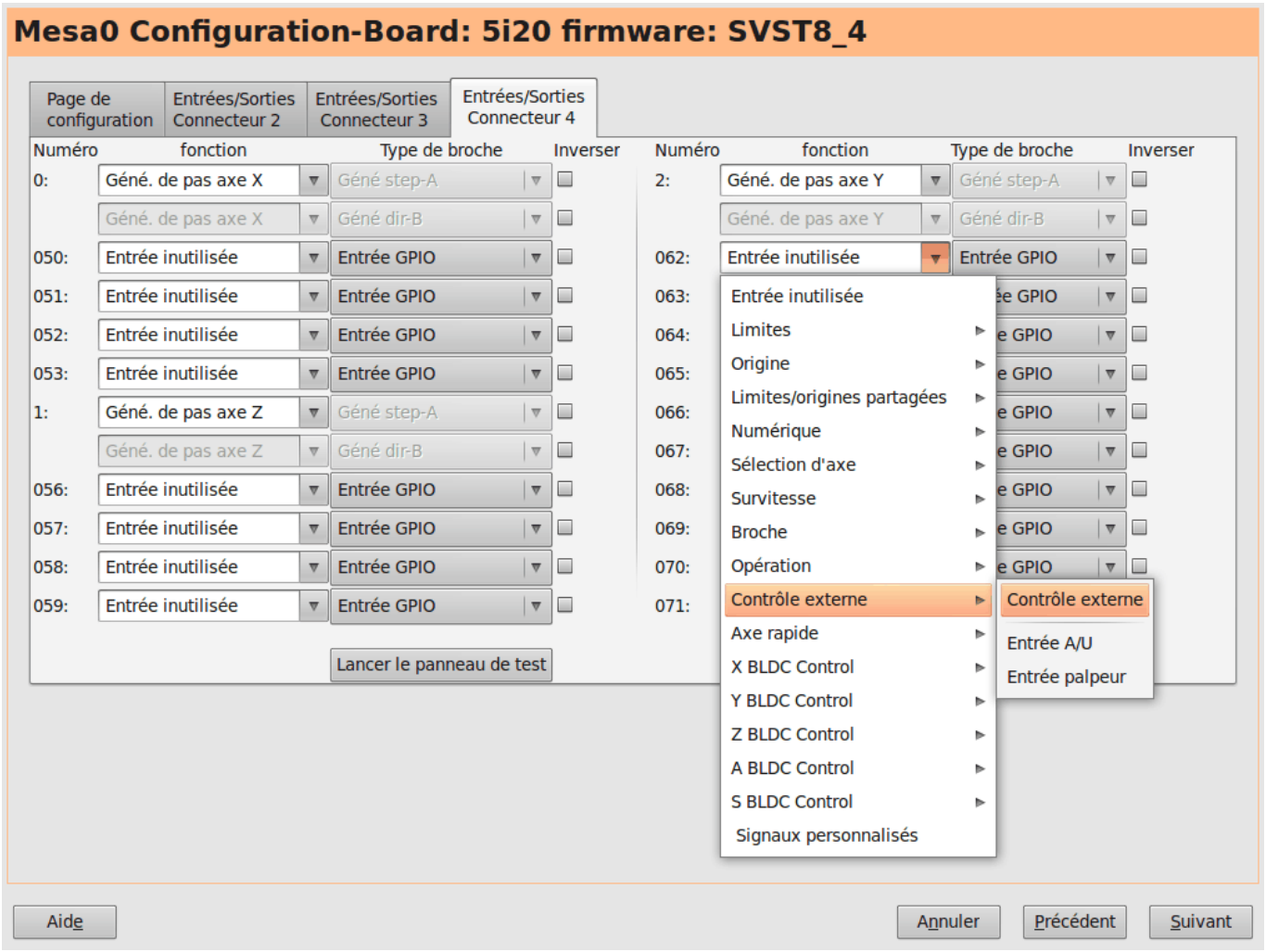
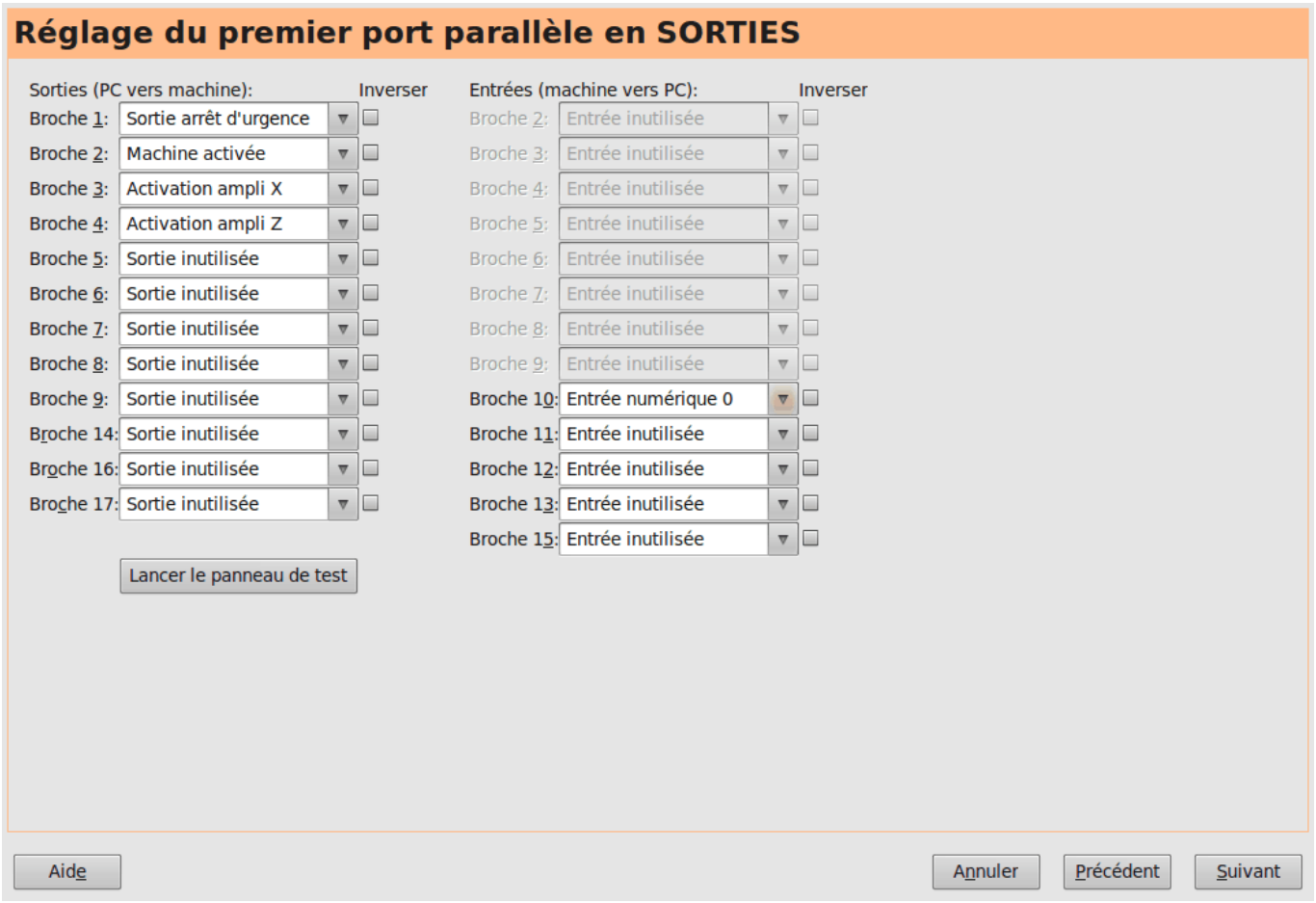


Figure 3.29: Réglages des E/S Mesa C4

Sur cet onglet, il y a un mélange entre des broches GPIO et des générateurs de pas. Les sorties générateur de pas et de direction peuvent être inversées. Noter que l'inversion d'un signal Step Gen modifie les délais de pas, il doivent correspondre à ce que le contrôleur attend.

Configuration des ports parallèles



Les ports parallèles peuvent être utilisés pour de simples E/S similaires aux broches GPIO Mesa.

3.6.8 Configuration des axes

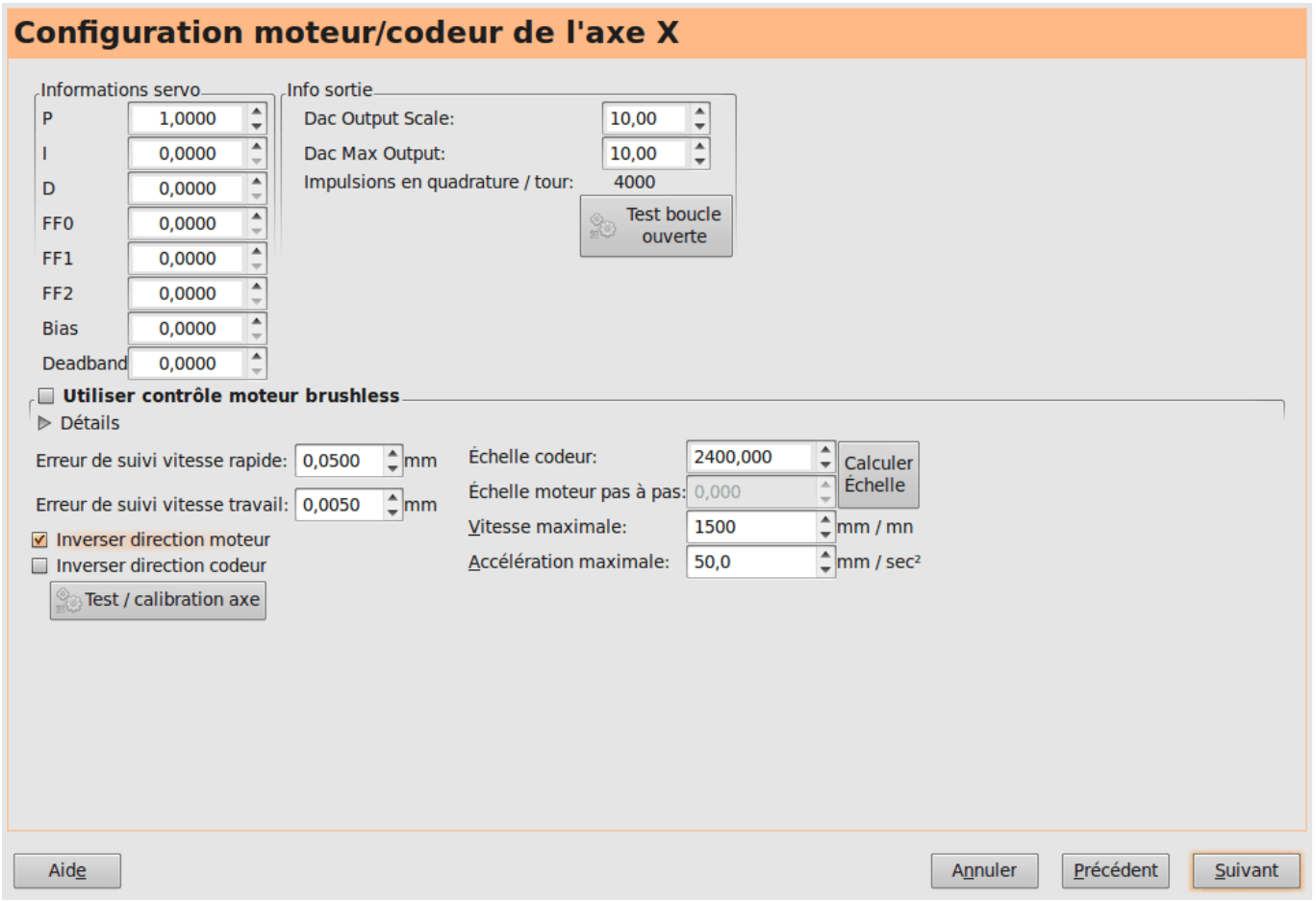


Figure 3.30: Configuration des axes

Cette page permet de configurer et tester un moteur combiné ou non à un codeur. Si un servomoteur est utilisé, un test en boucle ouverte est disponible. si un moteur pas à pas est utilisé, un test de réglage est disponible.

Test en boucle ouverte

Le test en boucle ouverte est important car il confirme la bonne direction du moteur et du codeur. Le moteur doit se déplacer dans le sens positif sur l'axe lorsque le bouton est pressé dans le sens positifs et aussi le codeur doit compter dans le même sens. Le mouvement de l'axe doit suivre les normes conventionnelles des machine-outil, sinon l'affichage graphique de l'axe n'aura pas de sens. Espérons que la page d'aide et le diagramme vous aideront à comprendre cela. Noter que les directions des axes sont celles du mouvement de l'outil et non celle du mouvement de la table. Il n'y a pas de rampe d'accélération lors du test en boucle ouverte, il convient donc de commencer avec une valeur faible du DAC. Déplacer l'axe sur une distance connue, confirmera la bonne mise à l'échelle du codeur. Le codeur doit compter dans le même sens, même sans la puissance sur le moteur, mais cela dépend de la manière dont le codeur est alimenté.

AVERTISSEMENT: Si le moteur et le codeur ne comptent pas dans le même sens, le servomoteur sera incontrôlable et s'emballera lors de l'utilisation en boucle fermée sous régulation PID.[6]

[6] PID: acronyme de Proportionnelle, Intégrale, Dérivée. Ce sont les 3 composantes de la régulation en boucle fermée de type PID.

Pour le moment les paramètres PID ne peuvent pas être testés dans PNCconf, ces réglages sont vraiment, pour quand vous rééditerez une configuration pour y mettre vos paramètres PID testés…

Echelle du DAC
[7] Deux valeurs de mise à l'échelle, Max Output et Offset sont utilisées pour linéariser le DAC.

Théorie
Ces deux valeurs sont les facteurs d'échelle et d'offset de la sortie vers l'amplificateur moteur, de l'axe. La deuxième valeur, l'offset, est soustraite de la sortie calculée (en Volts) et divisée par la première valeur (le facteur d'échelle), avant d'être écrite dans le DAC. La valeur d'échelle (Scale) s'exprime en Volts/Volts de sortie du DAC. Le décalage (offset) s'exprime en Volts. Elles peuvent être utilisées pour linéariser le DAC. Plus précisément, lors de l'écriture des sorties, LinuxCNC convertit d'abord la valeur effective de la sortie concernée, qui est en quasi-unités SI, en valeurs brute d'actionneur. Par exemple, des Volts pour un amplificateur DAC. La valeur de l'échelle peut être obtenue en analysant l'unité c'est-à-dire en déterminant le rapport [sortie unités SI]/[unités actionneur]. Par exemple, sur une machine avec un amplificateur en mode vitesse, qui fourni 1 Volt pour une vitesse résultante de 250 mm/s. Noter que les unités de l'offset sont en unités machine, ici des mm/s et qu'elles sont pré-soustraites des lectures capteur. La valeur de cet offset est obtenue en trouvant la valeur de sortie qui donne 0,0 sur la sortie de l'actionneur. Si le DAC est linéarisé, cet offset est normalement de 0,0. L'échelle et l'offset peuvent être utilisés pour linéariser le DAC, il en résultera des valeurs qui reflèteront les effets combinés du gain de l'amplificateur, de la non-linéarité du DAC, des unités du DAC, etc. Pour le faire, suivre cette procédure:

Construire une table de calibration pour la sortie. Piloter le DAC avec la tension souhaitée et mesurer le résultat:

Table 3.2: Mesure des tensions de sortie

Sortie brute	**Mesure**
-10	**-9.93**
-9	**-8.83**
0	**-0.96**
1	**-0.03**
9	**9.87**
10	**10.07**

- Par la méthode des moindres carrés, déterminer les coefficients **a**, **b** tels que **Mesure=a*Sortiebrute+b**
- Noter que nous voulons une sortie effective telle que la valeur mesurée soit identique à la consigne. Cela signifie

  - cmd=a*Sortiebrute+b

  - Sortiebrute=(cmd-b)/a

- Par conséquent, les coefficients **a** et **b** de l'ajustement linéaire peuvent être utilisés directement comme échelle et offset pour le contrôleur.

Valeur maximale de sortie
La valeur maximale pour la sortie de compensation PID qui est écrite sur l'ampli moteur, exprimée en volts. La valeur de sortie calculée est alignée sur cette limite. La limite est appliquée avant la mise à l'échelle des unités de sortie effective. La valeur est appliquée de manière symétrique aux deux limites, positive et négative.

[7] DAC, acronyme pour Convertisseur Analogique Digital

Test de réglage
Le test de réglage ne fonctionne, malheureusement, qu'avec les systèmes à base moteur pas à pas. Encore une fois vérifier que les directions de déplacements sur l'axe sont correctes. Puis tester le système en déplaçant l'axe d'avant en arrière, si l'accélération ou la vitesse maximum sont trop élevées, des pas seront perdus. Attention: Au cours de ce déplacement manuel garder à l'esprit que la distance d'arrêt est inversement proportionnelle à l'accélération et qu'avec une accélération faible il faut du temps et de la distance pour arrêter l'axe. Les fins de course ne sont pas fonctionnels pendant ce test. Un temps de pause peut être défini entre chaque mouvement d'essai. Cela permet de vérifier la position de l'axe et de voir si des pas sont perdus.

Timing des moteur pas à pas
La séquence de signaux des sorties pas a pas, doit être adaptée aux exigences du pilote des moteurs. Pncconf propose par défaut, certaines de ces séquences et il est possible de les personnaliser. Voir http://wiki.linuxcnc.org/cgi-bin/wiki.pl?Stepper_Drive_Timing pour y trouver des séquences pour le matériel le plus commun (n'hésitez pas à ajouter celles que vous avez expérimenté). En cas de doute utiliser une valeur élevée comme 5000, cela ne fera que limiter la vitesse maximale.

Contrôle de moteur Brushless
Ces options sont utilisées pour permettre le contrôle bas niveau des moteurs brushless avec un micro logiciel spécial et des cartes filles. Elles permettent également la conversion des capteurs à effet Hall d'un fabricant à l'autre. Ce n'est que partiellement pris en charge et aura besoin d'une intervention pour terminer les connexions de HAL. Contacter la mail-liste ou un forum pour avoir de l'aide.

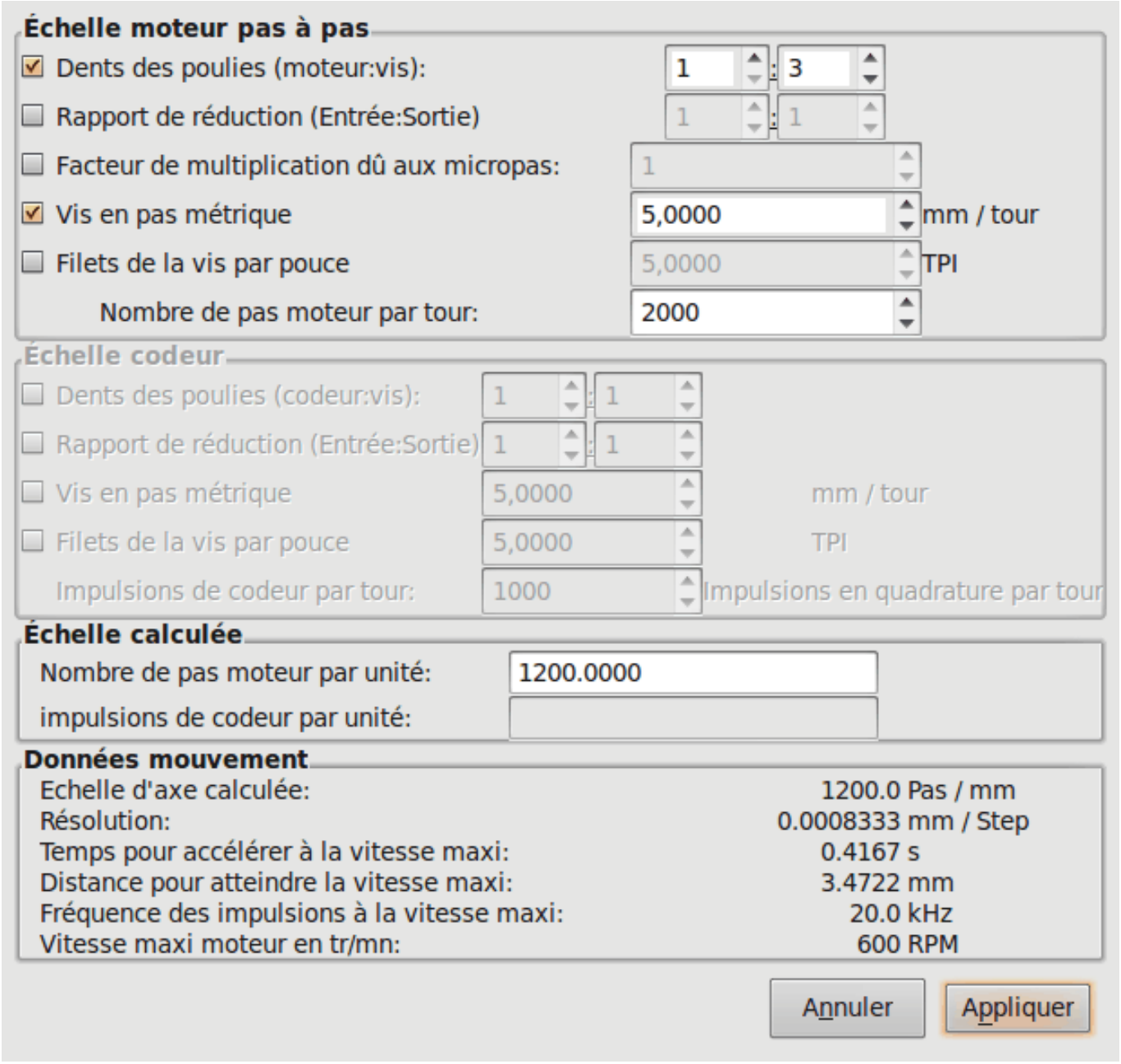


Figure 3.31: Calcul de l’échelle d’axe

Les paramètres d’échelle peuvent être saisis directement ou, on peut utiliser le bouton calculer échelle pour être assisté. Utiliser alors les cases à cocher pour sélectionner les calculs appropriés. Noter que Dents des poulies exige le nombre de dents et non le rapport de réduction. Rapport de réduction, le rapport de réduction est exactement le contraire, il exige le rapport entre poulie menante et poulie menée (Entrée/Sortie). Si l’échelle à déjà été calculée manuellement, il est possible de la saisir directement sans passer par l’assistant.

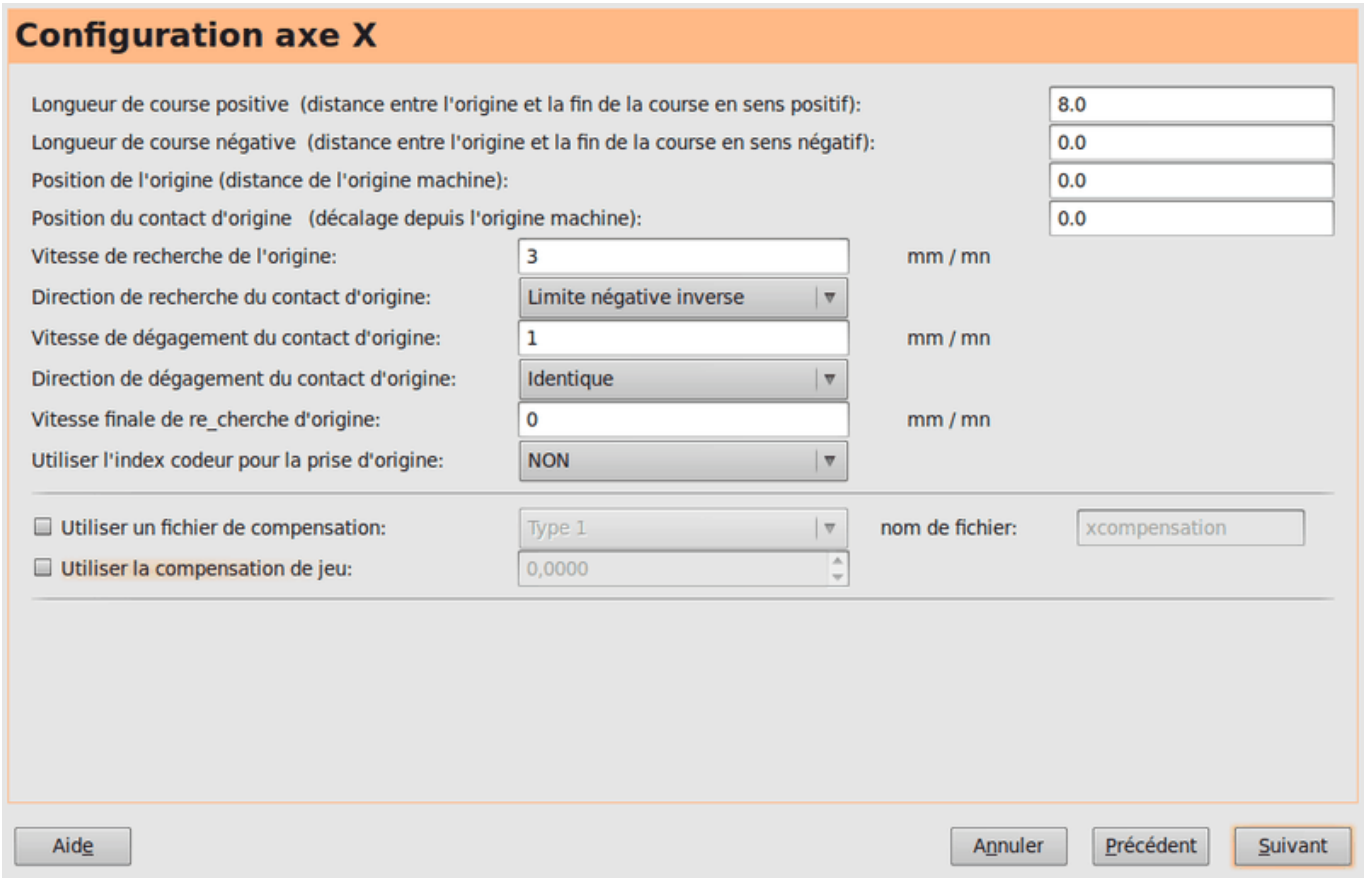


Figure 3.32: Configuration des axes

Se référer également à l'onglet diagramme pour deux exemples de disposition des contacts de fin de course d'origine machine et de limites. Ce sont deux exemples parmi les nombreuses façons différentes de placer ces contacts.

Important
Il est très important de commencer avec l'axe se déplaçant dans la bonne direction sinon l'acquisition du point d'origine est impossible !

Se souvenir que les directions positives et négatives se réfèrent toujours à l'outil et jamais à la table.

Sur une fraiseuse classique

- Lorsque la table se déplace vers l'opérateur, c'est la direction positive de l'axe Y.
- Lorsque la table se déplace à gauche, c'est la direction positive de l'axe X.
- Lorsque la table se déplace vers le bas, c'est la direction positive de l'axe Z.
- Lorsque la tête se déplace vers le haut, c'est aussi la direction positive de l'axe Z.

Sur un tour classique

- Lorsque l'outil se déplace à droite, en s'éloignant du mandrin, c'est le sens positif de l'axe Z.
- Lorsque l'outil se déplace vers l'opérateur, c'est le sens positif de l'axe X.

- Certains tours ont un axe X opposé, dans ce cas l'outil est à l'arrière, cela fonctionne bien, mais l'affichage graphique d'AXIS ne peut pas refléter cette configuration.

Lorsque des contacts d'origine machine et des contacts de fin de course sont utilisés, LinuxCNC attend des signaux de HAL au niveaux haut lorsque le contact est actionné. Si le signal d'un fin de course est inversé, LinuxCNC détectera en permanence que la machine est en bout de course. Si la logique de recherche du contact d'origine machine est mauvaise (fichier ini), LinuxCNC lancera la séquence de recherche d'origine machine de l'axe dans la mauvaise direction.

Décider de l'emplacement des fins de courses
Les fins de course de limite d'axe sont au delà des limites logicielles, ils protègent la machine en cas de problème électrique, par exemple, l'emballement d'un servomoteur. Les fins de course doivent être placés de manière à ce que l'axe ne puisse pas percuter une butée mécanique. Attention: si la distance d'activation du contact de fin de course est trop faible, avec l'inertie du mobile il pourra le dépasser. Les fins de course des limites d'axes, doivent être actifs à l'état bas et ils doivent aussi couper la puissance sur l'axe concerné. Le contact doit s'ouvrir à l'activation du fin de course. Utiliser un autre câblage est possible mais il est moins sécurisé. Il peut être nécessaire d'inverser le signal de HAL dans LinuxCNC pour avoir un état actif haut, TRUE signifie que le contact a été activé. Lorsqu'au démarrage de LinuxCNC un avertissement de limite et affiché même si l'axe n'est pas sur un des fins de course, le signal est probablement inversé. Utiliser HALMETER pour vérifier l'état du signal de HAL correspondant, par exemple, axis.0.pos-lim-sw-in, fin de course positif de l'axe X.

Décider de l'emplacement des contacts d'origine machine
Si des fins de course de limite d'axe sont utilisés, il est possible de les utiliser également comme contacts d'origine machine. Un contact d'origine machine séparé est utile si les axes sont longs et que le déplacement vers un fin de course dure trop longtemps pour un usage normale ou que le déplacement vers une extrémité présente des problèmes d'interférences avec le porte-pièce ou la pièce. Par exemple sur un tour, le déplacement en bout de banc n'est pas efficace pour un point d'origine machine et un contact placé vers le centre est certainement meilleur. Si codeur avec un index est utilisé, le contact agit comme point de référence et l'index suivant sera le point d'origine machine effectif.

Décider de la position de l'origine machine
L'origine machine dans LinuxCNC sert de référence à tous les systèmes de coordonnées utilisateur. Il n'y a pas d'emplacement particulier pour ce point. Seuls quelques G-codes accèdent au système de coordonnées machine (G53, G30 et G28). Si l'option de changement d'outil sur G30 est utilisée, placer l'origine machine à cet endroit peut être commode. Par convention, il est plus simple d'avoir l'origine machine sur le contact d'origine.

Décider de la position finale de l'origine
Ça consiste simplement à placer le chariot ou la broche à la position la plus commode après que LinuxCNC soit initialisé et que les points d'origines machine de chacun des axes lui soit connus.

Définition des côtés positifs/négatifs et des longueurs de courses maximales
Placer l'axe à l'origine. Faire un repère sur le mobile et un autre sur la partie fixe. Déplacer la machine jusqu'au contact de limite d'axe. Mesurer la distance entre les deux repères pour obtenir la longueur de déplacement maximale dans ce sens. Déplacer dans l'autre sens, sur le contact de limite de l'autre côté. Mesurer de nouveau les repères pour obtenir la longueur de déplacement maximale dans l'autre sens. Si l'origine machine est située sur une des limites d'axe, alors cette distance de déplacement sera évidemment de zéro.

Point d'origine machine
Ce point est le point de référence de la machine. (Ne pas confondre avec le point zéro de l'outil ou de la pièce). LinuxCNC référence tout à partir de ce point. Il doit être à l'intérieur des limites logicielles sinon la machine ne pourrait jamais l'atteindre. LinuxCNC utilise la position du contact d'origine machine pour calculer la position d'origine. Si la machine ne dispose pas de contact il faudra la positionner manuellement sur les points d'origine, cocher les axes l'un après

l'autre et pour chacun, presser le bouton POM des axes. Dans Axis, le symbole indiquant que l'origine machine de l'axe est connue s'affichera alors à droite de la visu de l'axe concerné.

Course de la table
C'est la distance maximale que l'axe peut parcourir dans chaque direction. Ceci peut ou ne peut pas être mesuré directement de l'origine aux contacts de fin de course. Le cumul des courses positives et négatives sera égal à la longueur de course totale.

Course positive
C'est la distance depuis l'origine de l'axe, jusqu'au fin de course de limite du côté positif. Si l'origine de l'axe est placée sur le fin de course de limite positive, cette valeur est égale à zéro. Les valeurs possibles sont positives ou égales à zéro.

Course négative
C'est la distance depuis l'origine de l'axe, jusqu'au fin de course de limite du coté négatif. Ou la course totale moins la course positive. Si l'origine de l'axe est placée sur le fin de course de limite négative, cette valeur est de zéro. Les valeurs possibles sont négatives égales à zéro. Si la valeur entrée dans PNCconf n'est pas négative, elle sera déduite des autres valeurs.

Position de l'origine
C'est la position ou se termine la séquence de prise d'origine machine. Elle est référencée par rapport à l'origine et peut être positive, si cette position finale est du coté positif ou négative, si cette position finale est du coté négatif.

Position du contact d'origine machine
C'est la distance depuis le contact d'origine jusqu'à la position de l'origine. Il peut être négatif ou positif selon de quel côté de l'origine il est placé. Depuis ce point, si l'axe doit être déplacé dans la direction positive pour arriver à l'origine, alors la valeur sera négative, sinon elle sera positive. Si il est mis à zéro, l'origine sera à l'emplacement du contact (plus la distance éventuelle pour attendre l'index suivant, si une règle de mesure, ou un codeur de position avec index sont utilisés).

Vitesse de recherche du contact d'origine machine
Vitesse utilisée pendant le déplacement vers le contact d'origine machine en unités par minute.

Direction de recherche du contact d'origine machine
Direction de la recherche de l'origine machine. Négatif ou Positif selon le coté de l'axe où se trouve le contact d'origine machine.

Vitesse d'acquisition du contact d'origine machine
Vitesse lente de détection du contact d'origine machine, en unités par minute.

Vitesse vers la position de l'origine
Vitesse utilisée pour déplacer le mobile de la position d'acquisition du contact d'origine machine, vers la position finale de l'origine, en unités par minute. Si réglée à 0 c'est la vitesse de déplacement rapide qui sera utilisée.

Direction d'acquisition du contact d'origine machine
Direction d'acquisition de l'origine machine, peut être dans la même direction que la recherche, ou à l'opposé.

Origine machine sur l'index du codeur
LinuxCNC attendra l'impulsion d'index du codeur après l'acquisition du contact d'origine machine.

Utiliser un fichier de compensation de jeu
Permet de spécifier le nom et le type d'un fichier de compensation de jeu. Permet une compensation sophistiquée. Voir le manuel.

Utiliser la compensation de jeu
Permet de régler la compensation du jeu de la vis, ne peut pas être utilisé en même temps qu'un fichier de compensation. Voir le manuel.

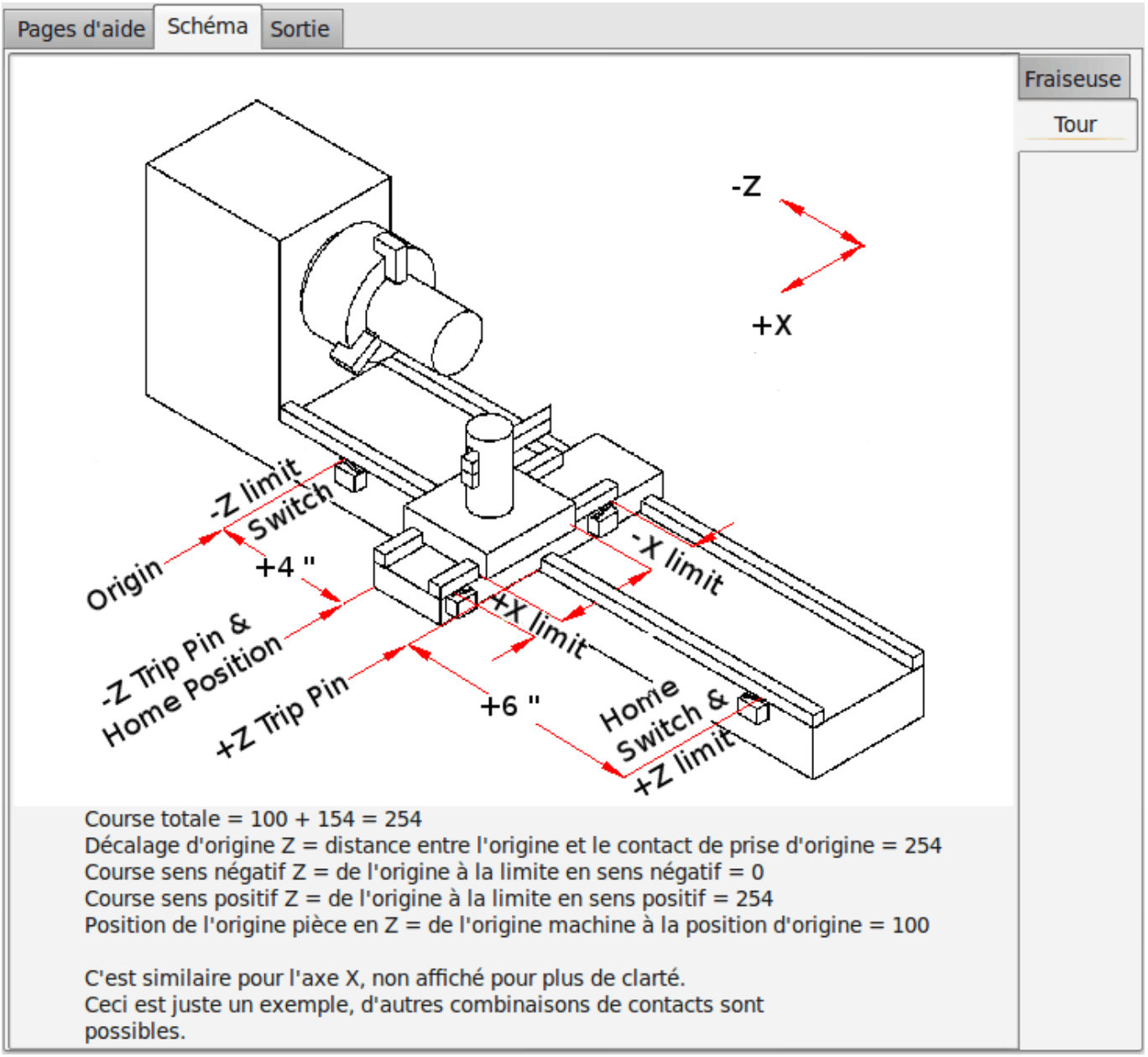


Figure 3.33: Dessin d’aide à l’identification des axes et fins de course

Ce dessin devrait aider à comprendre un exemple de positionnement des contacts de fin de course et les directions standards sur un tour. Sur ce tour, l’axe Z a deux contacts de fin de course, le contact positif est utilisé également comme contact de prise d’origine machine. La position du zéro machine (origine machine de l’axe) est placée à la limite négative. Le bord gauche du chariot est la came qui active le fin de course de la limite négative et le côté droit, la came qui active le fin de course de la limite positive. Nous voyons que la position finale de l’origine se trouve à 4 pouces de distance de l’origine de l’axe, du côté positif. Si le chariot était déplacé jusqu’à la limite positive, nous mesurerions 10 pouces entre la limite négative et la came du côté négatif du chariot (fin de course bord gauche du chariot).

Configuration de la broche

Si un signal de contrôle de la broche est présent, cette page permet de le configurer.

Tip
Beaucoup d'options de cette page ne sont visibles que si les sélections appropriées ont été choisies dans les pages précédentes. Si des signaux de broche ont été sélectionnés, alors cette page est disponible pour les configurer.

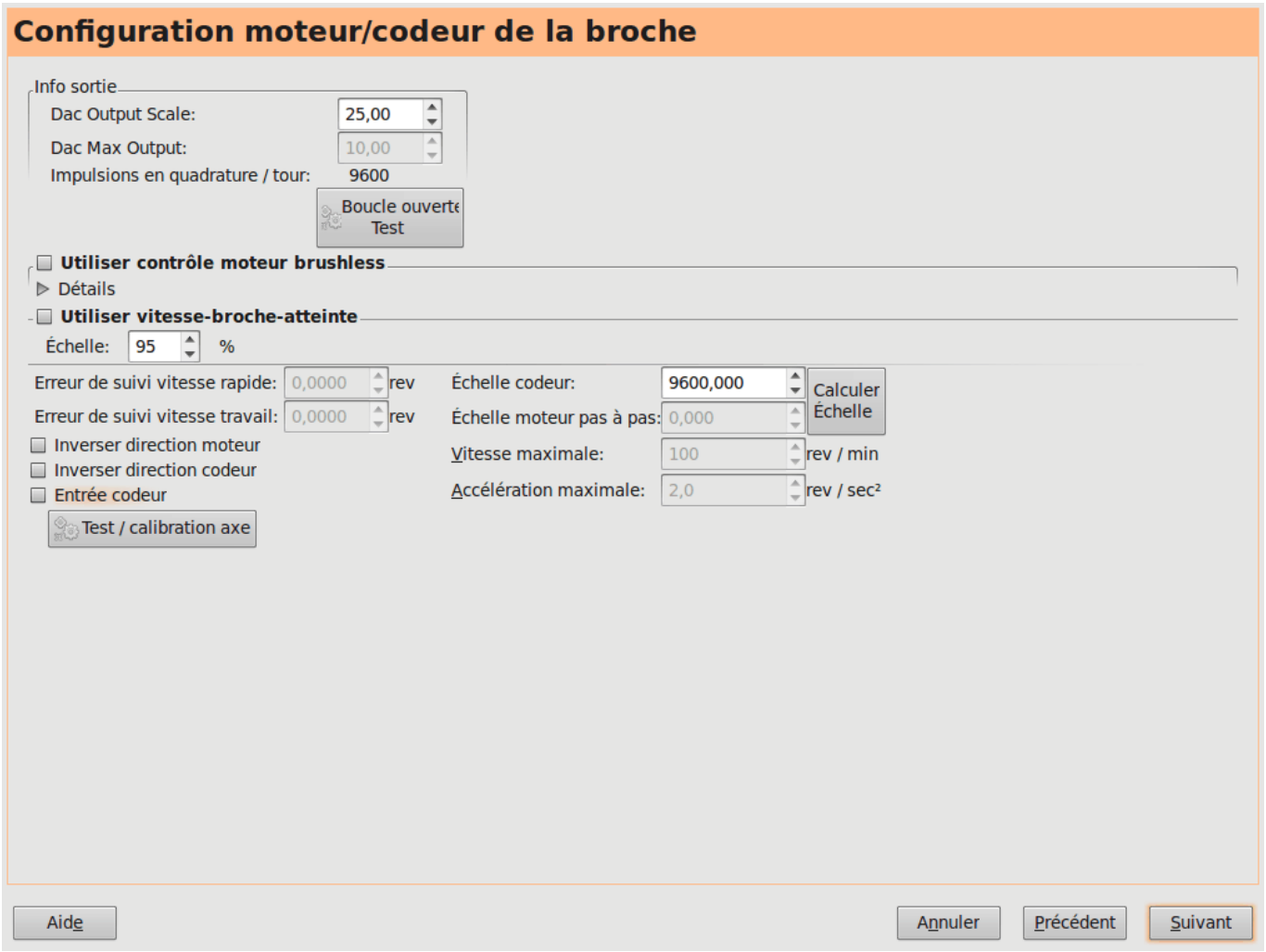


Figure 3.34: Configuration de la broche

Cette page est semblable à la page de configuration des moteurs d'axe mais il y a quelques différences: À moins que l'on ait choisi un moteur pas à pas pour la conduite de la broche il n'y a pas d'accélération ni de limitation de vitesse. Il n'y a pas de support pour les changements de vitesse ni pour les gammes de vitesses. Si une option VCP d'affichage de vitesse broche est choisie, alors la Vitesse broche atteinte, l'échelle, la vitesse et les réglages des filtres seront visibles. L'information sur la vitesse de broche permet à LinuxCNC d'attendre que celle-ci ait atteint la vitesse de consigne, avant de déplacer les axes. C'est particulièrement pratique sur les tours, lors de l'utilisation d'une vitesse de coupe constante avec de grands changements de diamètre. Il exige un retour d'information par codeur ou par un signal de vitesse broche numérique, typiquement connecté à un variateur de vitesse (VFD).

En utilisant le retour d'information d'un codeur, il est possible de choisir une plage de vitesse broche atteinte comme tolérance de vitesse, au delà de laquelle, la vitesse broche sera admise comme étant la vitesse de consigne.

En utilisant le retour d'information d'un codeur, l'affichage de vitesse VCP peut être irrégulier, des

filtres peuvent dans ce cas, être utilisés pour corriger l'affichage. L'échelle du codeur doit être réglée à la valeur comptage codeur/rapport de réduction utilisé. Si une seule entrée est utilisée pour le codeur de broche, la ligne suivante doit être ajoutée:

```
setp hm2_7i43.0.encoder.00.counter-mode 1
```

(Changer le nom de la carte et le numéro de codeur selon besoins) dans le fichier HAL personnalisé. Lire la section codeurs dans Hostmot2 pour plus d'information sur les modes de comptage.

3.6.9 Options avancées

Cette page permet de régler les commandes HALUI, de charger classicladder. Elle propose des exemples de programmes en Ladder. Si l'option GladeVCP a été choisie, comme pour la mise à zéro de l'axe sur l'origine pièce. Les commandes nécessaires s'afficheront. Voir le manuel de HALUI pour utiliser des commandes personnalisées halcmds. Parmi les exemples de programmes ladder: Le programme Estop permet de gérer un contact externe d'arrêt d'urgence ou permet à l'interface graphique de déclencher l'arrêt d'urgence. La commande périodique de la pompe du graissage centralisé est disponible.
Le contact de mise au zéro pièce de l'axe Z (longueur d'outil) s'utilise avec une plaque de référence, le contact (touch-off) de GladeVCP et les commandes spéciales HALUI sont là pour permettre rapidement, une recherche de l'origine pièce.

Le programme série modbus est un squelette de programme, vierge, préréglé pour l'utilisation de classicladder avec le protocole série modbus. Voir la section classicladder dans le manuel.

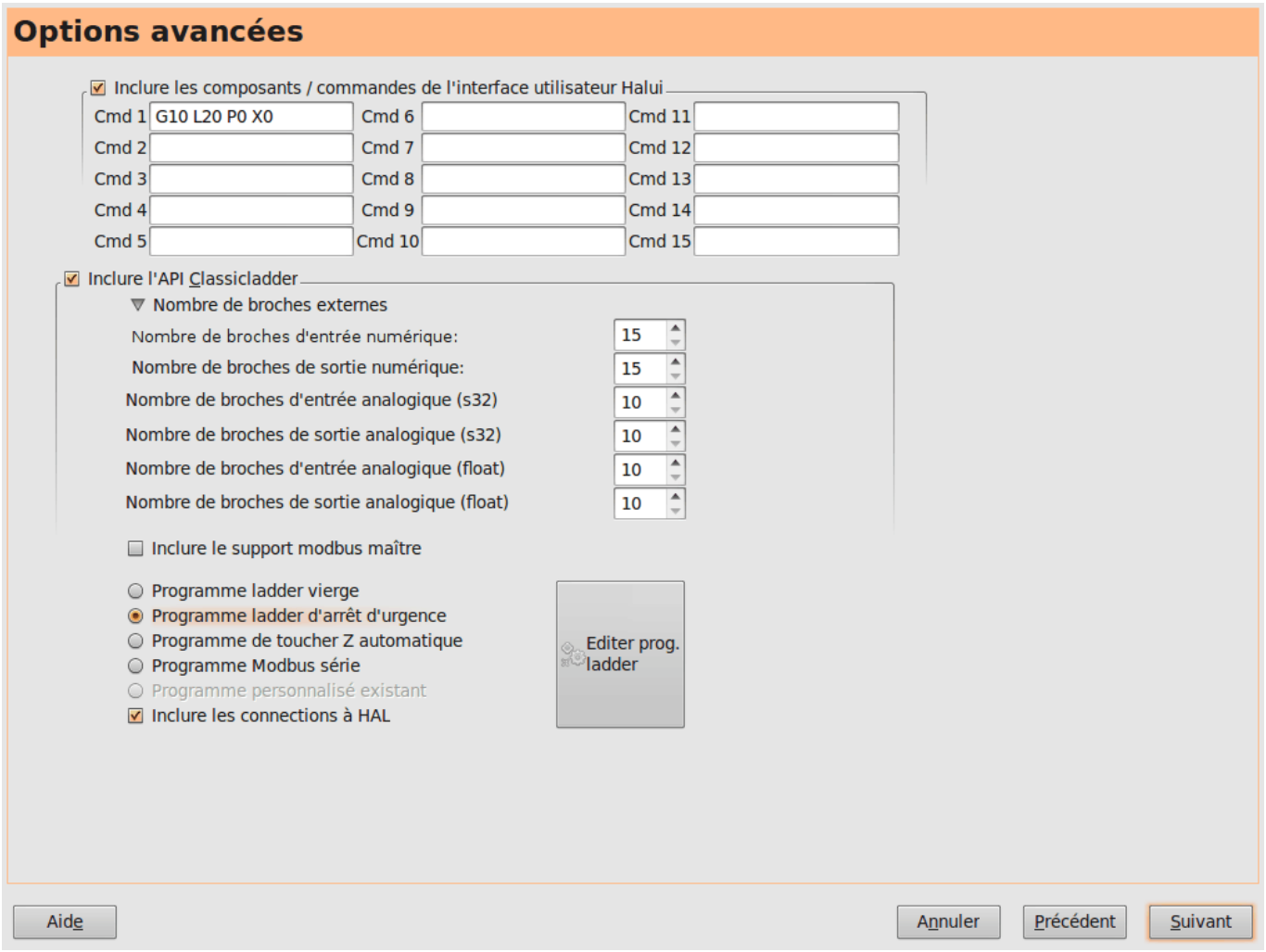


Figure 3.35: Options avancées

3.6.10 Composants de HAL

Cette page permet d'ajouter des composants de HAL supplémentaires qui seront utilisés dans les fichiers HAL personnalisés. De cette manière il n'est pas nécessaire d'éditer le fichier HAL principal en permettant malgré tout à l'utilisateur de définir ses propres composants.

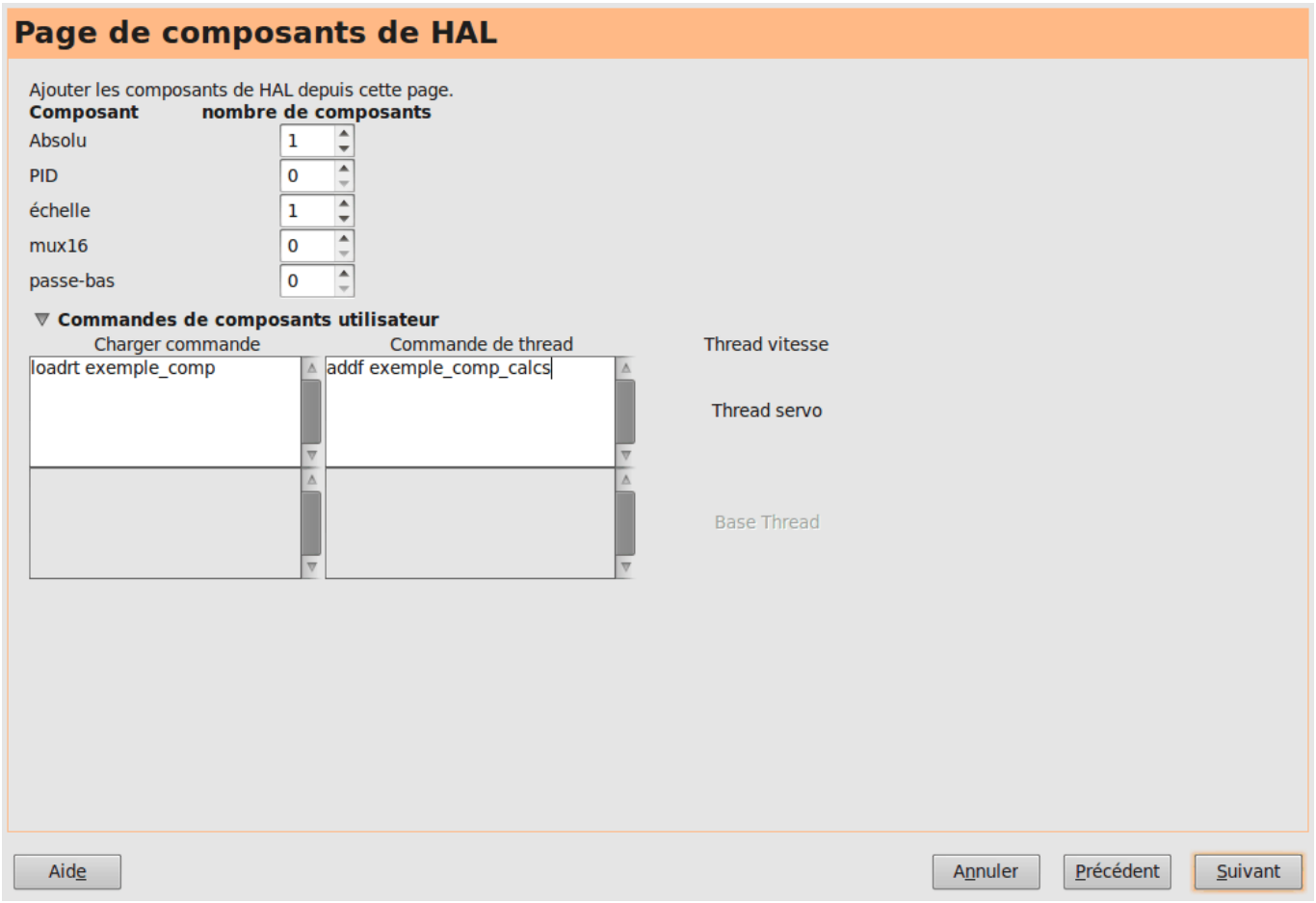


Figure 3.36: Composants de HAL

La première sélection est prévue pour les composants que pncconf utilise en interne. Il est possible de configurer pncconf pour qu'il charge les instances additionnelles pour votre fichier HAL personnalisé. Sélectionner le nombres d'instances dont a besoin le fichier de personnalisation et pncconf ajoutera ce qui est nécessaire. Si 2 composants sont nécessaires et que pncconf à besoin d'un composant interne, il chargera 3 composants et utilisera le dernier.

Composants de commande personnalisés
: Cette sélection permettra de charger des composants de HAL que pncconf n'utilise pas. Ajoute les commandes loadrt ou loadusr dans l'entête loading command. Ajoute la commande addf dans l'entête Thread command. Les composants seront ajoutés au thread entre la lecture des entrées et l'écriture des sorties, dans l'ordre ou ils sont écrits dans thread command.

3.6.11 Utilisation avancée de PNCConf

PNCconf fait de son mieux pour permettre un personnalisation souple à l'utilisateur, PNCconf supporte les noms de signaux particuliers, le chargement de composants personnalisés comme la personnalisation des fichiers de HAL et des microprogrammes. Il y a aussi les noms de signaux que PNCconf fournit, indépendamment des options choisies, pour les fichiers HAL personnalisés.

Avec une conception réfléchie, la plupart des personnalisations devraient fonctionner, même si des options doivent être modifiées par la suite dans PNCCONF. Finalement, si les personnalisations vont au-delà du périmètre de travail de PNCCONF, il sera possible d'utiliser PNCCONF pour construire

une configuration de base, ou d'utiliser une des configurations fournies en standards par LinuxCNC et de l'éditer pour obtenir ce que est souhaité.

Nom de signaux personnalisés

Si un composant doit être connecté à quelque chose dans un fichier HAL personnalisé, écrire un nom de signal unique dans la boîte de dialogue. Certains composants ajouteront des suffixes au nom du signal personnalisé.

```
Les codeurs ajoutent  < Nom personnalisé >:
-position
-count
-velocity
-index-enable
-reset

Les contrôles de moteurs pas à pas ajoutent:
-enable
-counts
-position-cmd
-position-fb
-velocity-fb

Les PWM ajoutent:
-enable
-value

Les broches GPIO auront juste le nom du signal d'entrée qui leur est connecté.
```

De cette façon on peut établir des connexions à ces signaux dans les fichiers personnalisés de HAL et avoir toujours la possibilité de les déplacer plus tard.

Charger un microprogramme personnalisé

PNCconf cherche le microprogramme sur le système et cherche ensuite le fichier XML qu'il peut convertir et qu'il comprend. Ces fichiers XML sont seulement fournis pour les microprogrammes officiellement délivrés par l'équipe LinuxCNC. Pour utiliser un microprogramme personnalisé, il faut le convertir en tableau que PNCconf comprend et ajouter son chemin dans le fichier de préférences de PNCCONF. Par défaut le chemin recherché est sur le bureau, dans un dossier nommé custom_firmware contenant un fichier nommé firmware.py.

Le fichier caché des préférence est dans le dossier home de l'utilisateur et se nomme .pncconf-preferences, pour l'éditer il faut sélectionner Afficher les fichiers cachés. On peut voir le contenu de ce fichier au premier démarrage de PNCCONF. Presser le bouton d'aide et regarder la page de sortie. Demander sur la liste de diffusion LinuxCNC ou sur le forum pour des renseignements pour convertir un microprogramme personnalisé. Tous les microprogrammes ne peuvent pas être utilisés avec PNCCONF.

Fichiers HAL Personnalisés

Il y a quatre fichiers personnalisés utilisables pour ajouter des commandes a HAL:

- custom.hal est prévu pour les commandes HAL utilisées avant le chargement de l'interface graphique. Il est exécuté après le fichier HAL de configuration nommé : non-de-la-configuration.hal
- custom_postgui.hal est prévu pour les commandes qui doivent être exécutées après le chargement de l'interface graphique Axis ou PYVCP autonomes.
- custom_gvcp.hal est prévu pour les commandes qui doivent être exécutées après le chargement de GLADE VCP.
- shutdown.hal est prévu pour des commandes exécutées quand LinuxCNC se ferme de façon contrôlée.

3.7 Petite FAQ Linux

Voici quelques commandes et techniques de base pour l'utilisateur débutant sous Linux. Beaucoup d'autres informations peuvent être trouvées sur le site web de LinuxCNC ou dans les man pages.

3.7.1 Login automatique

Quand vous installez LinuxCNC avec le CD-Live Ubuntu, par défaut vous devez passer par la fenêtre de connexion à chaque démarrage du PC. Pour activer le login automatique ouvrez le menu Système → Administration → Fenêtre de connexion. Si l'installation est récente la fenêtre de connexion peut prendre quelques secondes pour s'ouvrir. Vous devez entrer le mot de passe utilisé pour l'installation pour accéder à la fenêtre des préférences. Ouvrez alors l'onglet Sécurité, cochez la case Activer les connexions automatiques et saisissez votre nom d'utilisateur ou choisissez en un dans la liste déroulante. Vous êtes maintenant dispensé de la fenêtre de connexion.

3.7.2 Les Man Pages

Les Man pages sont des pages de manuel générées automatiquement le plus souvent. Les Man pages existent pour quasiment tous les programmes et les commandes de Linux.

Pour visualiser une man page ouvrez un terminal depuis Applications → Accessoires → Terminal. Par exemple si vous voulez trouver quelques choses concernant la commande find, tapez alors dans le terminal:

```
man find
```

Utilisez les touches Vers le haut et Vers le bas pour faire défiler le texte et la touche **Q** pour quitter.

3.7.3 Lister les modules du noyau

En cas de problème il est parfois utile de connaître la liste des modules du noyau qui sont chargés. Ouvrez une console et tapez:

```
lsmod
```

Si vous voulez, pour le consulter tranquillement, envoyer le résultat de la commande dans un fichier, tapez la sous cette forme:

```
lsmod > mes_modules.txt
```

Le fichier mes_modules.txt résultant, se trouvera alors dans votre répertoire home si c'est de là que vous avez ouvert la console.

3.7.4 Éditer un fichier en root

Éditer certains fichiers du système en root peut donner des résultats inattendus! Soyez très vigilant quand vous éditez en root, une erreur peut compromettre tout le système et l'empêcher de redémarrer. Vous pouvez ouvrir et lire de nombreux fichiers systèmes appartenant au root qui sont en mode lecture seule.

3.7.4.1 A la ligne de commande

Ouvrir un terminal depuis Applications → Accessoires → Terminal.
Dans ce terminal, tapez:

```
sudo gedit
```

Ouvrez un fichier depuis Fichiers → Ouvrir puis éditez le.

3.7.4.2 En mode graphique

1. Faites un clic droit sur le bureau et choisissez Créer un lanceur...
2. Tapez un nom tel que éditeur, dans la zone Nom.
3. Entrez gksudo ”gnome-open %u” dans la zone Commande et validez.
4. Glissez un fichier et déposez le sur votre lanceur, il s’ouvrira alors directement dans l’éditeur.

3.7.5 Commandes du terminal

3.7.5.1 Répertoire de travail

Pour afficher le chemin du répertoire courant dans le terminal tapez:

```
pwd
```

3.7.5.2 Changer de répertoire

Pour remonter dans le répertoire précédent, tapez dans le terminal:

```
cd ..
```

Pour remonter de deux niveaux de répertoire, tapez dans le terminal:

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

Pour aller directement dans le sous-répertoire linuxcnc/configs tapez:

```
cd linuxcnc/configs
```

3.7.5.3 Lister les fichiers du répertoire courant

Pour voir le contenu du répertoire courant tapez:

```
ls
```

3.7.5.4 Trouver un fichier

La commande find peut être un peu déroutante pour le nouvel utilisateur de Linux. La syntaxe de base est:

```
find répertoire_de_départ <paramètres> <actions>
```

Par exemple, pour trouver tous les fichiers .ini dans votre répertoire linuxcnc utilisez d'abord la commande pwd pour trouver le répertoire courant. Ouvrez un nouveau terminal et tapez:

```
pwd
```

il vous est retourné par exemple le résultat suivant:

```
/home/robert
```

Avec cette information vous pouvez taper, par exemple, la commande:

```
find /home/robert/linuxcnc -name *.ini -print
```

Le -name est le nom du fichier que vous recherchez et le -print indique à find d'afficher le résultat dans le terminal. Le *.ini indique à find de retourner tous les fichiers portant l'extension .ini

3.7.5.5 Rechercher un texte

Tapez dans un terminal:

```
grep -lir "texte à rechercher" *
```

Pour trouver tous les fichiers contenant le texte "texte à rechercher" dans le répertoire courant, tous ses sous-répertoires et en ignorant la casse. Le paramètre **-l** demande de ne pas afficher les résultats normalement mais à la place, d'indiquer le nom des fichiers pour lesquels des résultats auraient été affichés. Le paramètre **-i** demande d'ignorer la casse. Le paramètre **-r** demande une recherche récursive (qui inclus tous les sous-répertoires dans la recherche). Le caractère * est un jocker indiquant tous les fichiers.

3.7.5.6 Messages du boot

Pour visualiser les messages du boot utilisez la commande dmesg depuis un terminal. Pour enregistrer ces messages dans un fichier, redirigez les avec:

```
dmesg > dmesg.txt
```

Le contenu de ce fichier pourra alors être copié et collé à destination des personnes en ligne qui vous aideront à diagnostiquer votre problème.

Pour nettoyer le tampon des messages tapez cette commande:

```
sudo dmesg -c
```

C'est utile avant de lancer LinuxCNC, pour que ne soit enregistrés que les messages relatifs au fonctionnement courant de LinuxCNC.

Pour trouver les adresses des ports parallèles de la machine, tapez cette commande grep pour filtrer les informations contenues dans dmesg.

```
dmesg|grep parport
```

3.7.6 Problèmes matériels

3.7.6.1 Informations sur le matériel

Pour voir la liste du matériel installé sur les ports PCI de votre carte mère, tapez la commande suivante dans un terminal:

```
lspci -v
```

Pour voir la liste du matériel installé sur les ports USB de votre carte mère, tapez la commande suivante dans un terminal:

```
lsusb -v
```

3.7.6.2 Résolution du moniteur

Lors de l'installation d'Ubuntu les réglages du moniteur sont automatiquement détectés. Il peut arriver que la détection fonctionne mal et que la résolution ne soit que celle d'un moniteur générique en 800x600.

Pour résoudre ce cas, suivez les instructions données ici:

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

3.8 Particularités des tours

Ce chapitre va regrouper les informations spécifiques aux tours, il est encore en cours de rédaction.

3.8.1 Mode tour

Si l'interface graphique Axis est utilisée et que la machine est un tour, pour qu'Axis représente les axes et la position de l'outil correctement il conviendra d'éditer le fichier ini et de modifier la section [DISPLAY] comme ceci:

```
[DISPLAY]

# Pour indiquer à Axis que la machine est un tour (lathe).
LATHE = TRUE
```

Le mode tour dans Axis ne fixe pas le plan par défaut comme étant G18 (XZ). Il est nécessaire de l'ajouter dans le préambule de tout programmer G-code ou, encore mieux, de l'ajouter directement dans le fichier ini comme cela:

```
[RS274NGC]

# G-code modaux pour initialiser l'interpréteur
RS274NGC_STARTUP_CODE = G18 G20 G90
```

3.8.2 Fichier d'outils

La table d'outils est un fichier texte qui contient les informations de chaque outil. Ce fichier se trouve dans le même répertoire que le fichier ini, il est appelé tool.tbl par défaut. Les outils peuvent être dans un changeur d'outils ou simplement changés manuellement. Le fichier peut être édité avec un éditeur de texte ou être mis à jour en utilisant G10 L1, L10, L11. Axis peut aussi lancer l'éditeur de texte du système en y chargeant la table, l'opérateur peut ainsi intervenir directement sur les valeurs des outils. Le nombre maximum d'outils dans la table d'outils est de 56. Les numéros d'outil et de poches peuvent aller jusqu'à 99999.

Les versions antérieures de LinuxCNC avaient deux différents formats de table d'outils pour les fraiseuses et les tours, mais depuis la version 2.4.x, un format de table d'outil unique est utilisé. Il faut juste ignorer les parties de la table d'outils qui ne concernent pas la machine. Plus d'informations ici sur le format de la table d'outils.

3.8.3 Orientations des outils de tournage

La figure suivante montre les angles d'orientations des outils de tour ainsi que les informations sur l'angle frontal de l'arête de coupe (FRONTANGLE) et l'angle arrière de l'arête de coupe (BACKANGLE). Les positions vont croissantes dans le sens horaire par rapport à une ligne parallèle à l'axe Z, le zéro étant côté Z+.

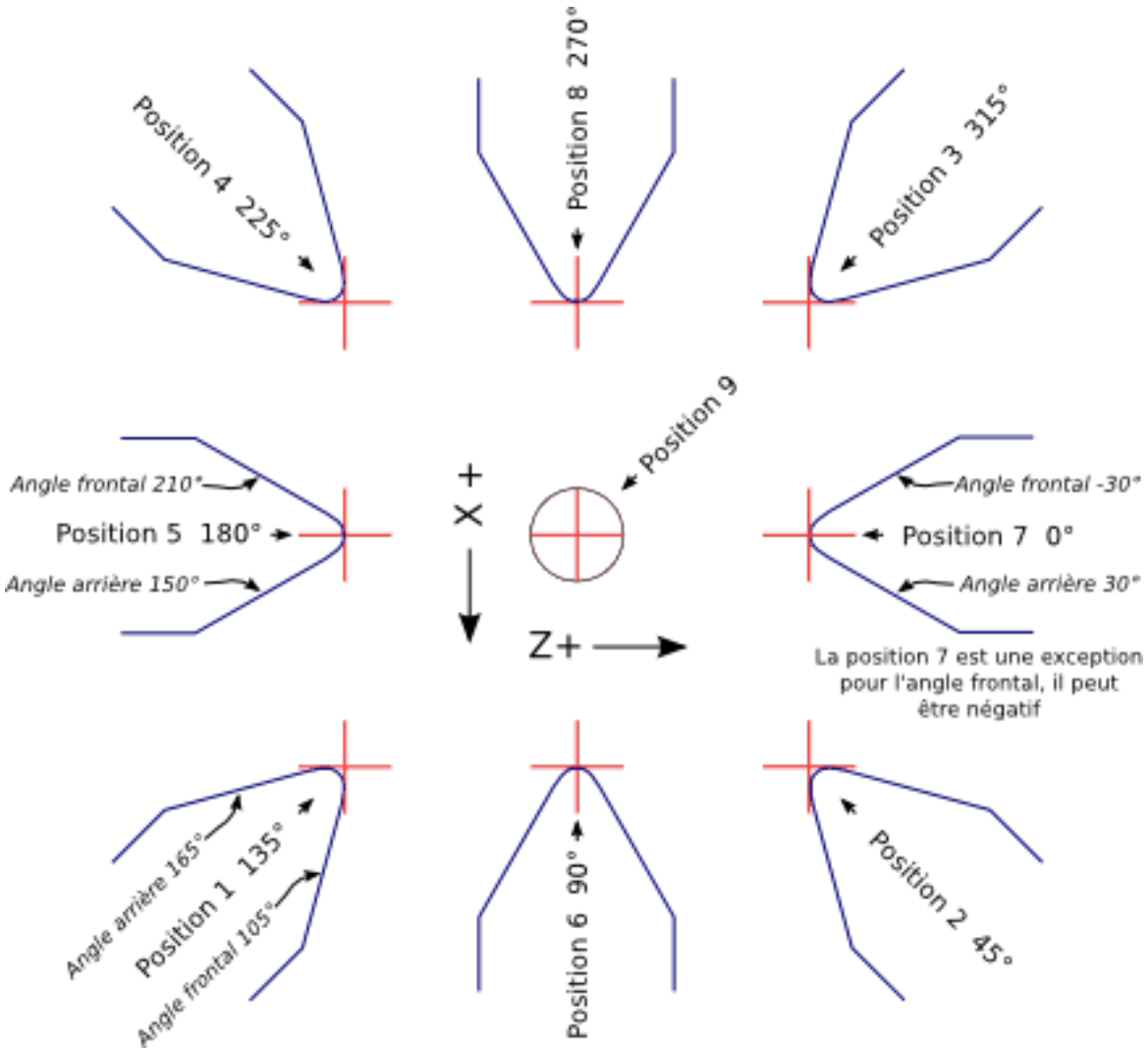
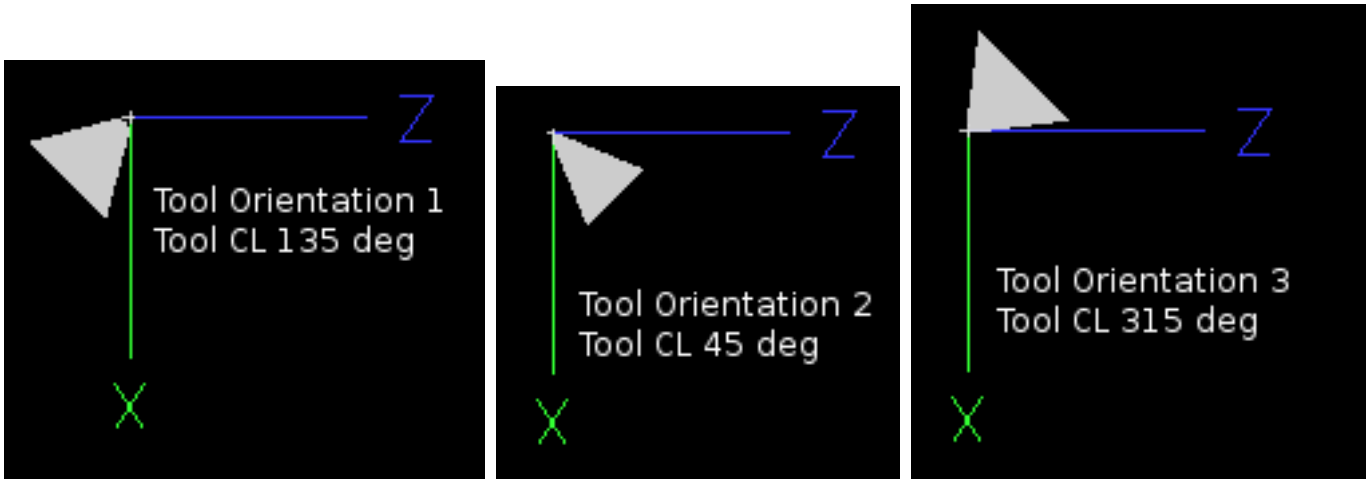
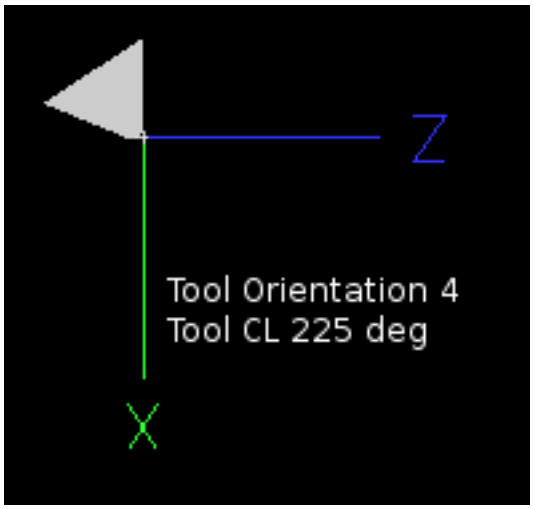


Figure 3.37: Orientations des outils de tournage

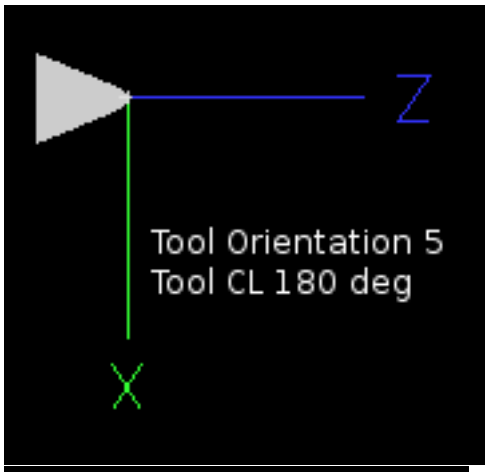
Les images ci-dessous montrent la représentation qu'Axis donne des orientations de l'outil, en se référant à la figure ci-dessus.

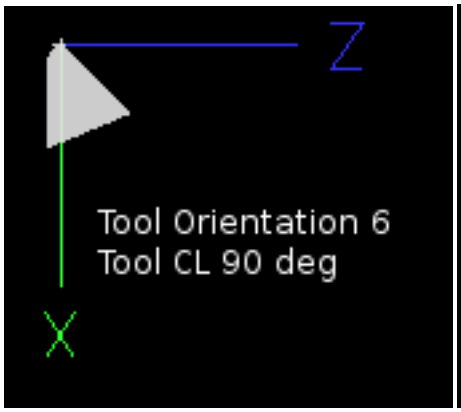
Outil dans les positions 1, 2, 3 et 4

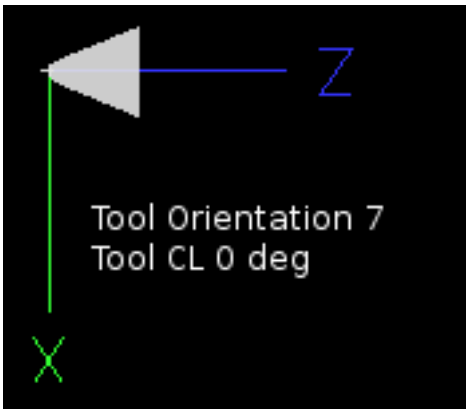


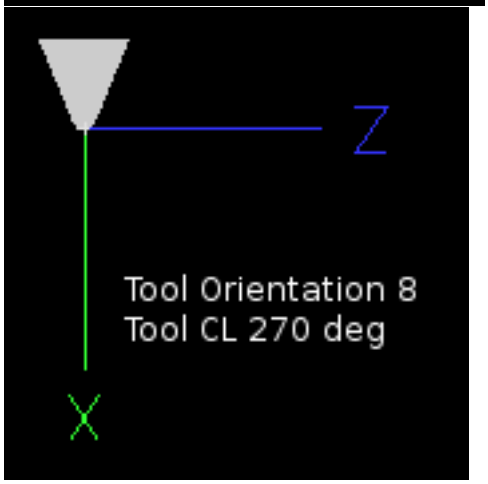


Outil dans les positions 5, 6, 7 et 8









3.8.4 Correction d’outil

Quand AXIS est utilisé sur un tour, il est possible de corriger l’outil sur les axes X et Z. Les corrections sont alors introduites directement dans la table d’outils en utilisant le bouton Toucher et sa fenêtre de dialogue.

3.8.4.1 Offset d'outil en X

L'offset X pour chaque outil correspond à un décalage de l'axe de la broche.

Une méthode consiste à prendre un outil de tournage standard et usiner un diamètre. Mesurer exactement ce diamètre puis, sans toucher à l'axe X, dans la fenêtre qui s'ouvre, après un appui sur le bouton Toucher, saisir le diamètre mesuré, ou le rayon si c'est le mode en cours. Ensuite, à l'aide d'encre à tracer ou d'un marqueur, recouvrir une zone sur la pièce, faire tangenter l'outil à cet endroit pour qu'il touche juste la surface encrée, ajuster alors l'offset X au diamètre mesuré de la pièce en utilisant le bouton Toucher. S'assurer que le rayon de bec est bien défini dans la table d'outils, pour que le point contrôlé soit correct. Le Toucher ajoute automatiquement un G43, de sorte que l'offset s'applique immédiatement à l'outil courant.

3.8.4.2 Séquence typique de correction d'outil en X:

1. Prise d'origine machine de chacun des axes, si ce n'est pas déjà fait.
2. Déclarer l'outil avec M6 Tn dans lequel n est le numéro de l'outil courant, présent en table d'outils.
3. Sélectionner l'axe X dans la fenêtre de l'onglet Contrôle manuel (F3).
4. Déplacer l'axe X sur une position connue ou prendre une passe de test puis mesurer le diamètre obtenu.
5. Cliquer le bouton Toucher et choisir l'option Table d'outils, ce qui entrera la correction directement dans la table d'outil.
6. Recommencer la même séquence pour corriger l'axe Z.

Remarque: si le mode rayon est le mode courant, il faut évidemment entrer le rayon et non pas le diamètre.

3.8.4.3 Offset d'outil en Z

L'offset de l'axe Z peut être un peu déroutant au premier abord car il est composé de deux éléments. Le premier est l'offset de la table d'outils, le second est l'offset des coordonnées machine. Nous allons d'abord examiner l'offset de la table d'outils. Une méthode consiste à utiliser un point fixe sur le tour et à ajuster l'offset Z de tous les outils à partir de ce point fixe. Certains utilisent le nez de broche ou la face du mandrin. Cela donne la possibilité de changer d'outil et d'ajuster son offset Z, sans avoir à réinitialiser tous les outils.

3.8.4.4 Séquence typique de correction d'outil en Z:

1. Prise d'origine machine de tous les axes, si ce n'est pas déjà fait.
2. S'assurer qu'aucune compensation n'est activée pour le système de coordonnées courant.
3. Déclarer l'outil avec M6 Tn dans lequel n est le numéro de l'outil courant, présent en table d'outils.
4. Sélectionner l'axe Z dans la fenêtre de l'onglet contrôle manuel (F3).
5. Placer un cylindre dans le mandrin.
6. Faire tangenter l'outil contre la face du cylindre.
7. Cliquer le bouton Toucher puis choisir Table d'outils et saisir la position à 0.0.

8. Répéter l'opération pour chaque outil, en utilisant le même cylindre.

Maintenant, tous les outils sont compensés à la même distance d'une position standard. Si un outil doit être changé, par exemple par un foret il suffira de répéter la séquence précédente pour qu'il soit synchronisé avec l'offset Z du reste des outils. Certains outils pourraient nécessiter un peu de réflexion pour déterminer le point contrôlé par rapport au point de Toucher. Par exemple, un outil de tronçonnage de 3.17mm d'épaisseur qui est touché sur le côté gauche, alors que l'opérateur veut Z0 sur le côté droit, il lui faudra alors saisir 3.17 dans la fenêtre du Toucher.

3.8.4.5 Machine avec tous les outils compensés

Une fois que tous les outils ont leurs offsets renseignés dans la table d'outils, il est possible d'utiliser n'importe quel outil présent en table d'outils pour ajuster le décalage du système de coordonnées machine.

3.8.4.6 Séquence typique de décalage du système de coordonnées:

1. Prise d'origine machine de tous les axes, si ce n'est pas déjà fait.
2. Déclarer l'outil avec M6 Tn dans lequel n est le numéro de l'outil courant, présent en table d'outils.
3. Envoyer un G43 pour que l'offset de l'outil soit activé. (voir ci-dessous)
4. Tangenter l'outil contre la pièce et fixer l'offset machine Z.

Ne pas oublier d'envoyer le G43 sur l'outil avant de définir le décalage du système de coordonnées machine, les résultats ne seraient pas ceux attendus… puisque la compensation de l'outil serait ajoutée à l'offset courant lorsque l'outil sera utilisé dans le programme.

3.8.5 Mouvements avec broche synchronisée

Sur un tour, les mouvements avec broche synchronisée nécessitent un signal de retour entre la broche et LinuxCNC. Généralement, c'est un codeur en quadrature qui fourni ce retour. Le manuel de l'intégrateur donne des explications sur l'utilisation des codeurs de broche.

Filetage Le cycle de filetage préprogrammé G76 est utilisé, tant en filetage intérieur qu'en filetage extérieur, voir la section G76.

Vitesse de coupe à surface constante La vitesse de coupe à surface constante utilise l'origine machine X modifiée par l'offset d'outil X, pour calculer la vitesse de rotation de la broche en tr/mn. La vitesse de coupe à surface constante permet de suivre les changements d'offset de l'outil. L'emplacement de l'origine machine de l'axe X doit être sur l'axe de rotation et doit se faire avec l'outil de référence (celui qui a l'offset à zéro).

Avance par tour L'avance par tour déplace l'axe Z de la valeur de F à chaque tour. Ce n'est pas destiné au filetage pour lequel il faut utiliser G76. D'autres informations sont dans la section sur G95.

3.8.6 Arcs

Le calcul des arcs peut être un exercice assez compliqué, même sur un tour, sans considérer les modes rayon et diamètre, ni l'orientation du système de coordonnées machine. Ce qui suit s'applique à des arcs au format centre. Sur un tour, il faut inclure G18 dans le préambule du programme G-code pour remplacer le G17 par défaut, le fait d'être en mode tour dans Axis ne suffit pas. Les arcs en G18, plan XZ utilisent les offsets pour I (l'axe X) et K (l'axe Z).

3.8.6.1 Les arcs et la cinématique du tour

Le tour classique a la broche à gauche de l'opérateur et l'outil entre l'opérateur et le centre de rotation du mandrin. C'est un agencement avec un axe Y(+) imaginaire pointant vers le sol.

Ce qui suit est valable pour ce type d'agencement:

- Le côté positif de l'axe Z pointe vers la droite, en s'éloignant de la broche.
- Le côté positif de l'axe X pointe vers l'opérateur, quand il est du côté de l'opérateur par rapport au centre de rotation, ses valeur sont positives.

Certains tours ont l'outil du côté arrière et un axe Y(+) imaginaire pointant vers le haut.

Les directions des arcs G2/G3 sont basées sur l'axe autour duquel ils tournent. Dans le cas des tours, il s'agit de l'axe imaginaire Y. Si l'axe Y(+) pointe vers le sol, il faut regarder vers le haut pour que l'arc paraisse aller dans la bonne direction. Alors qu'en regardant depuis le dessus il faut inverser les G2/G3 pour que l'arc semble aller dans la bonne direction.

3.8.6.2 Mode rayon et mode diamètre

Lors du calcul des arcs en mode rayon, il suffi de se rappeler la direction de rotation telle qu'elle s'applique à ce tour.

Lors du calcul des arcs en mode diamètre, X est le diamètre, l'offset X (I) est le rayon, même en mode diamètre G7.

3.8.7 Parcours d'outil

3.8.7.1 Point contrôlé

Le point contrôlé pour l'outil, suit la trajectoire programmée. Le point contrôlé est l'intersection entre deux lignes parallèles aux axes X et Z, tangentes au rayon de bec de l'outil, définies en faisant tangenter l'outil en X puis en Z. En cylindrage ou en dressage de face sur une pièce, la trajectoire de coupe et l'arête de coupe de l'outil suivent le même parcours. Lors du tournage d'un rayon ou d'un angle, l'arête de coupe de l'outil ne suit pas la trajectoire programmée, sauf si la compensation d'outil est activée. Dans la figure suivante, on voit bien que le point contrôlé n'est pas sur l'arête de coupe de l'outil comme on pourrait le supposer.

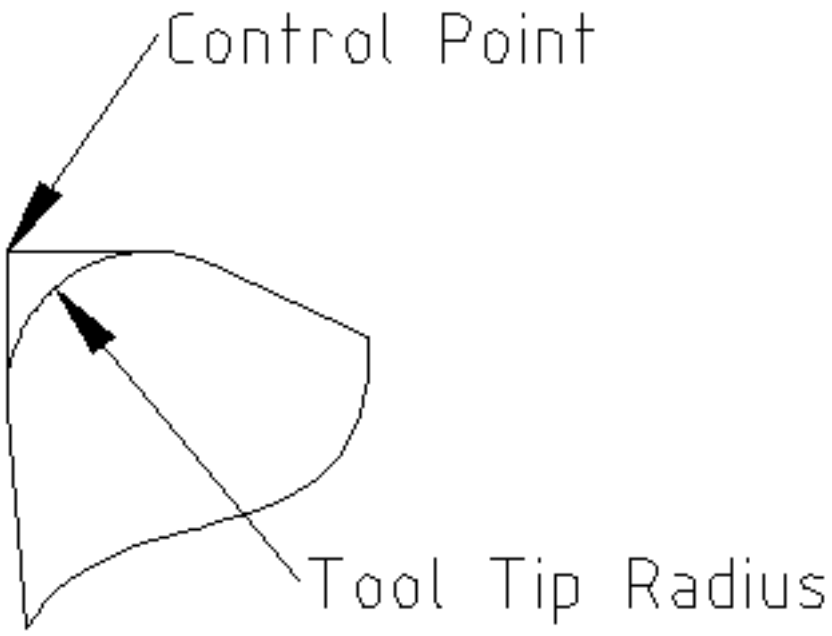


Figure 3.38: Point contrôlé

3.8.7.2 Tourner les angles sans compensation d'outil

Maintenant imaginons de programmer une rampe sans compensation d'outil. La trajectoire programmée est représentée sur la figure suivante. Comme on peut le voir, la trajectoire programmée et la trajectoire de coupe souhaitée sont identiques uniquement si les mouvements de tournage suivent les axes X et Z.

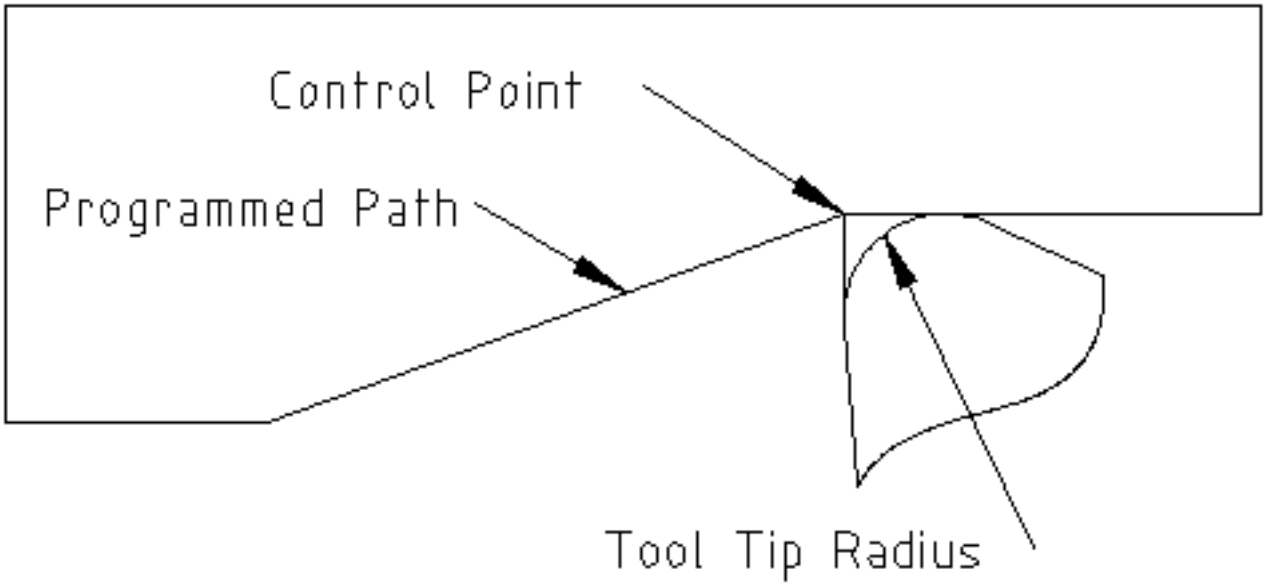


Figure 3.39: Tournage en rampe

Le point contrôlé progresse en suivant la trajectoire programmée mais l'arête de coupe ne suit pas cette trajectoire comme c'est visible sur la figure suivante. Pour résoudre ce problème, il est nécessaire d'activer la compensation d'outil et d'ajuster la trajectoire programmée pour compenser le rayon de bec de l'outil.

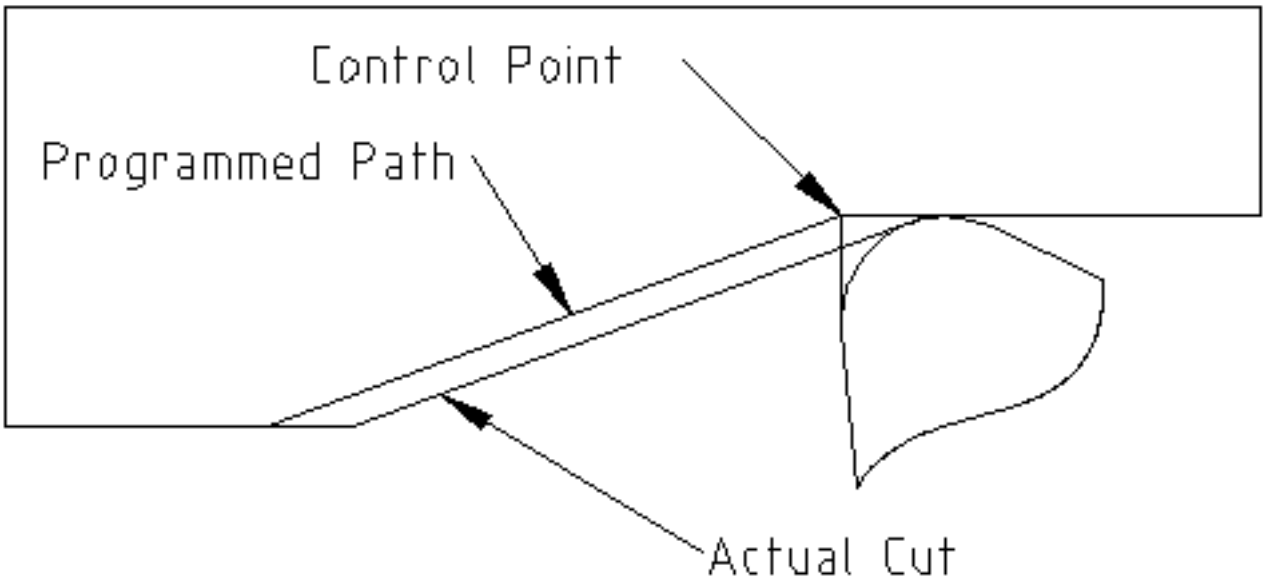


Figure 3.40: Trajectoire en rampe

Dans l'exemple ci-dessus, pour suivre la rampe programmée et obtenir la bonne trajectoire, il suffi de décaler la trajectoire de la rampe vers la gauche, de la valeur d'un rayon de bec.

3.8.7.3 Tournage avec rayon extérieur

Dans cet exemple nous allons examiner ce qui se passe durant le tournage d'un rayon extérieur sans compensation de rayon de bec. Sur la figure suivante on voit l'outil tourner un diamètre extérieur sur la pièce. Le point contrôlé de l'outil suit bien la trajectoire programmée, l'outil touche le diamètre extérieur de la pièce.

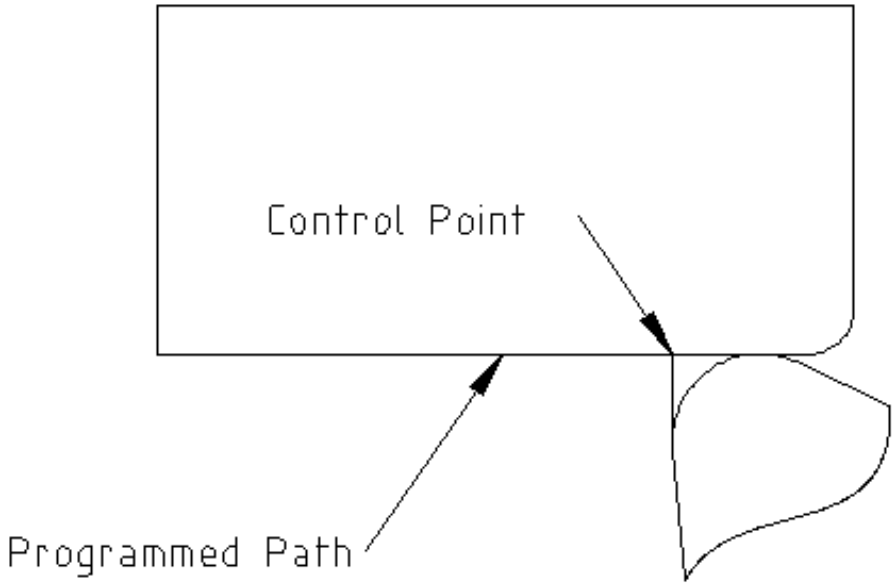


Figure 3.41: Tournage du diamètre

Sur la figure suivante, on voit que quand l'outil approche la fin la pièce, le point contrôlé continue de suivre la trajectoire alors que l'arête de coupe a déjà quitté la matière et coupe en l'air. On voit aussi que malgré qu'un rayon a été programmé, la pièce conserve son angle d'extrémité.

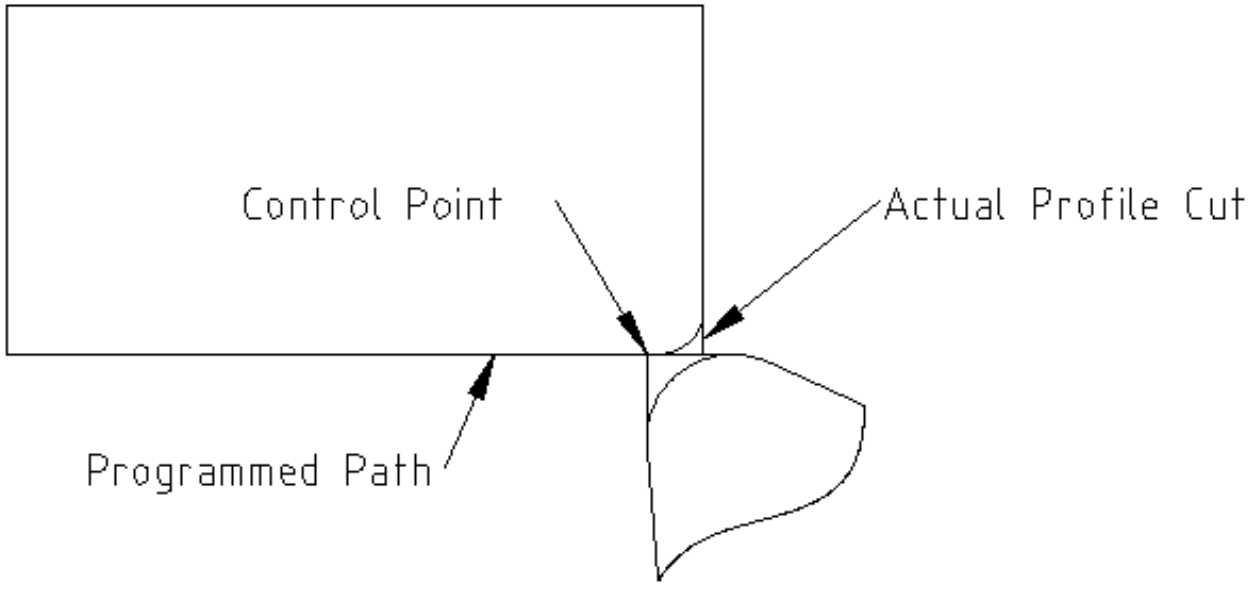


Figure 3.42: Tournage du rayon

Maintenant, comme on le voit, le point contrôlé suit bien la trajectoire programmée mais l'arête de coupe est en dehors de la matière.

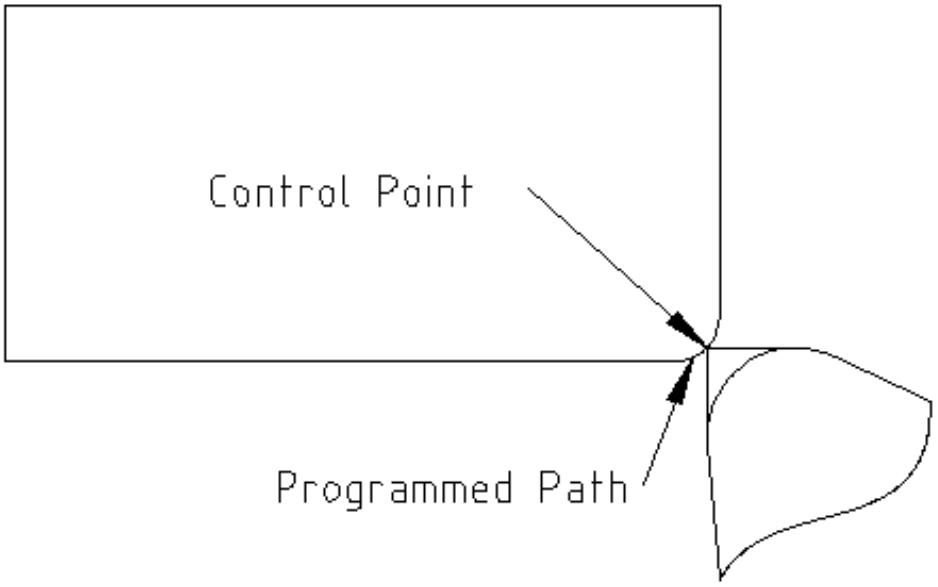


Figure 3.43: Tournage du rayon

Sur la figure finale, on voit que l'arête de coupe a terminé le dressage de la face mais en laissant un coin carré à la place du beau rayon attendu. Noter aussi que, pour la même raison, pour ne pas laisser de téton au centre de la pièce lors du dressage de sa face, il convient de dépasser le centre de rotation de la valeur d'un rayon de bec de l'outil.

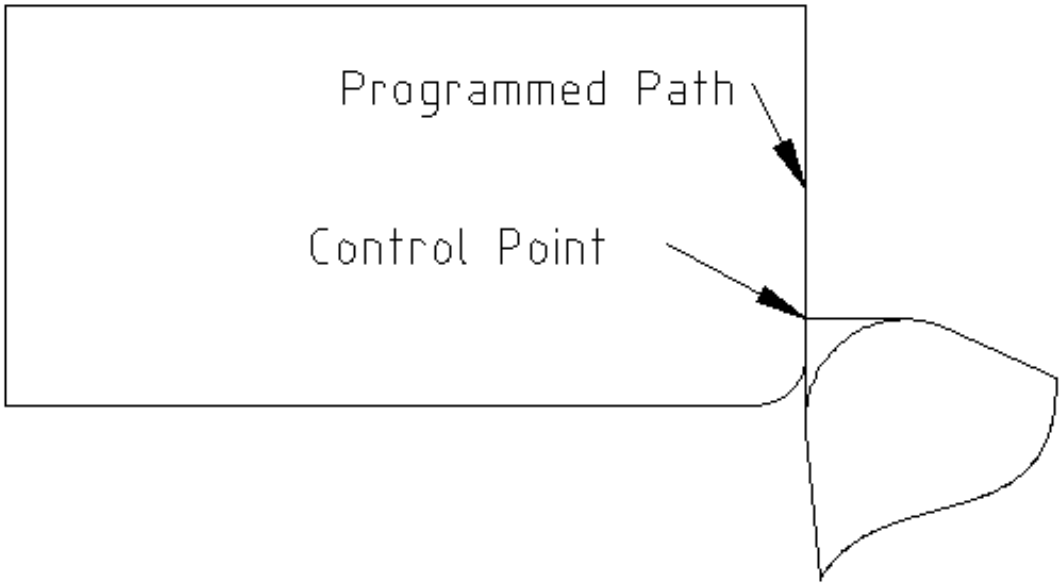


Figure 3.44: Dressage de la face

3.8.7.4 Utiliser la compensation d'outil

- Quand la compensation d'outil est utilisée sur un tour, penser à l'arête de coupe de l'outil comme étant celle d'un outil rond.

- Quand la compensation d’outil est utilisée, la trajectoire doit être suffisamment large pour qu’un outil rond n’interfère pas avec la pièce à la ligne suivante.
- Pour tourner des lignes droites sur un tour, il est préférable de ne pas utiliser la compensation d’outil. Par exemple pour aléser un trou avec une barre d’alésage un peu grosse, la place pourrait manquer pour dégager l’outil et faire le mouvement de sortie.
- Le mouvement d’entrée dans un arc avec la compensation d’outil, est important pour obtenir des résultats corrects.

Chapter 4

User Interfaces

4.1 L’interface graphique AXIS

4.1.1 Introduction

AXIS est une interface utilisateur graphique pour LinuxCNC, il offre un aperçu permanent du parcours de l’outil. Il est écrit en Python, utilise Tk et OpenGL pour afficher son interface graphique.

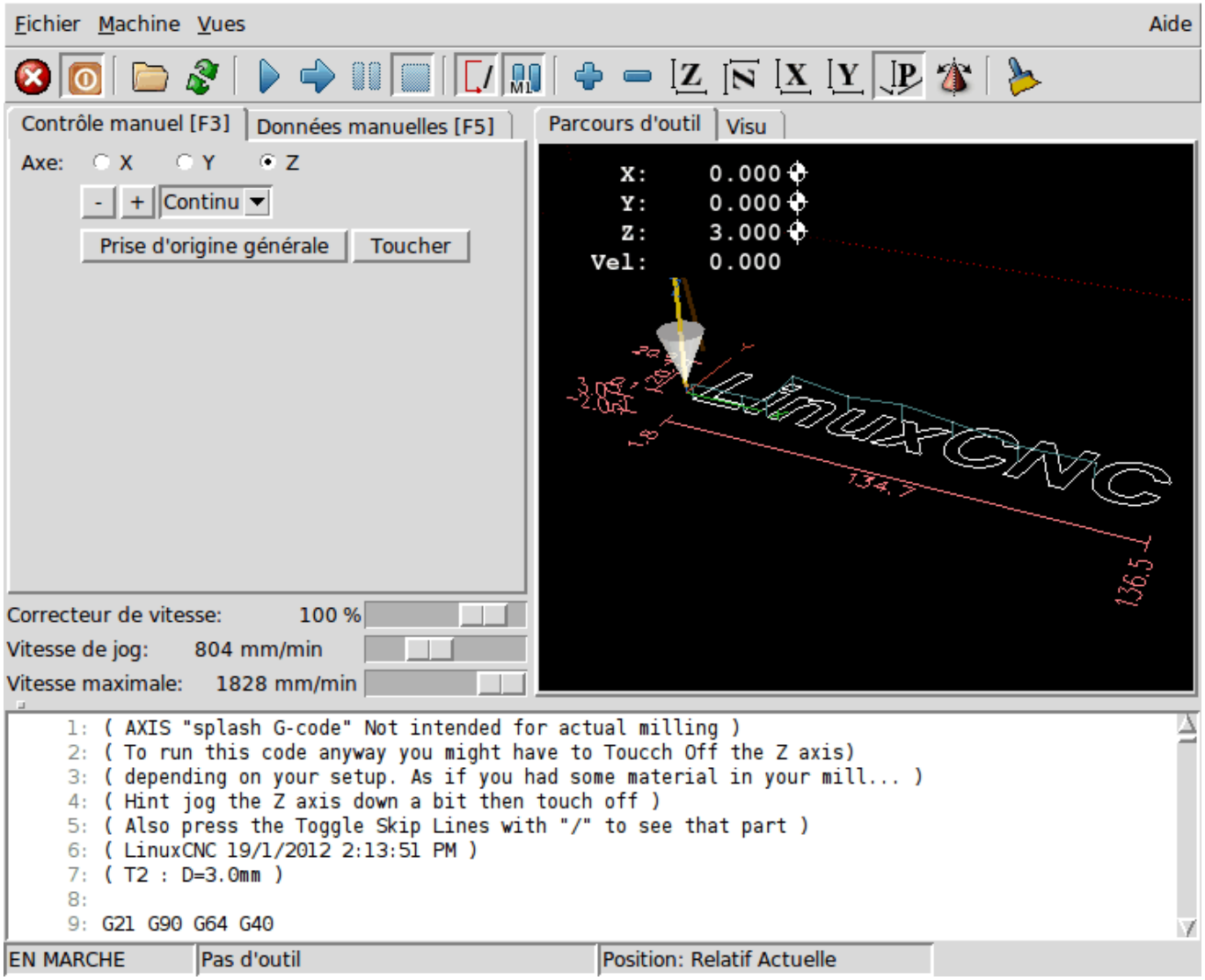


Figure 4.1: Fenêtre d'AXIS

4.1.2 Commencer avec AXIS

Pour choisir AXIS comme interface graphique de LinuxCNC, éditez le fichier .ini et dans la section [DISPLAY] changez la ligne DISPLAY comme ceci:

DISPLAY = axis

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

Quand AXIS démarre, une fenêtre telle que celle de la figure ci-dessus s'ouvre.

4.1.2.1 Une session typique avec AXIS

1. Lancer LinuxCNC et sélectionner un fichier de configuration.
2. Relâcher le bouton d'arrêt d'urgence A/U et presser celui de Marche machine.

3. Faire les prises d'origine machine POM pour chacun des axes.
4. Charger un fichier d'usinage.
5. Utiliser l'affichage du parcours d'outil pour vérifier que le programme est correct.
6. Brider le brut à usiner sur la table.
7. Faire les prises d'origine pièce POP de chacun des axes avec le jog et en utilisant le bouton Toucher.
8. Lancer le programme.
9. Pour usiner le même fichier une nouvelle fois, retourner à l'étape 6. Pour usiner un fichier différent, retourner à l'étape 4. Quand le travail est terminé, quitter AXIS.

4.1.3 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. Plus tard, cette zone affichera le parcours de l'outil déplacé par la machine, cette zone est appelée le parcours d'outil (backplot)
- Une barre de menus, une barre d'outils, des curseurs et des onglets permettant 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 (appelé aussi MDI), où les blocs de programme G-code peuvent être entrés et exécutés 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)

4.1.3.1 Options des menus

Certaines options de menu peuvent s'afficher en grisé, c'est dépendant des options du fichier de configuration ini.

Menu Fichier

- Ouvrir… - C'est une boîte de dialogue standard pour ouvrir un fichier G-code à charger dans AXIS. Si un filtre de programme a été configuré, il peut aussi être ouvert ici.
- Fichiers récents… - Affiche la liste des fichiers ouverts récemment.
- Éditer… - Ouvre le fichier G-code courant pour édition si un éditeur a été déclaré dans le fichier ini.
- Recharger… - Recharge le fichier G-code courant. Si le fichier a été édité, il doit être rechargé pour que les modifications prennent effet. Si un programme a été stoppé, pour le reprendre depuis le début, le recharger. Le bouton Recharger a le même effet que l'option de menu.

- Enregistrer le G-code sous… - Enregistre le fichier courant sous un nouveau nom.
- Propriétés… - Donne la somme des mouvements en vitesse rapide et celle en vitesse travail. Ne tient pas compte des accélérations, ni des décélérations, ni des modes de trajectoire, de sorte qu'il ne donne jamais de temps inférieur au temps réel d'exécution.
- Éditer la table d'outils… - Ouvre un dialogue permettant d'éditer les valeurs de la table d'outils.
- Recharger la table d'outils… - Après avoir édité la table d'outil, il convient de la recharger pour que les nouvelles valeurs soient prisent en compte.
- Éditeur de Ladder… - Si Classic Ladder a été chargé, il est possible de l'éditer ici.
- Quitter… - Termine la session courante de LinuxCNC.

Menu Machine

- Arrêt d'Urgence F1… - (bascule) Active/désactive l'arrêt d'urgence.
- Marche/Arrêt F2… - (bascule) Active/désactive la puissance machine.
- Démarrer le programme… - Lance l'exécution du programme G-code.
- Démarrer à la ligne sélectionnée… - Prudence avec cette commande, respecter la démarche suivante: Premièrement, sélectionner à la souris, la ligne à laquelle démarrer. Déplacer ensuite manuellement, l'outil à la position de la ligne précédente puis, cette commande exécutera le reste du code.

Warning
Ne pas utiliser la commande Démarrer à la ligne sélectionnée… si le programme G-code contient des sous-programmes.

- Pas à pas… - Avance d'un seul pas de programme.
- Pause… - Effectue une pause dans le programme.
- Reprise… - Reprends la marche après une pause.
- Stopper… - Stoppe le programme en marche.
- Arrêt sur M1… - Si M1 est rencontré et que cette option est cochée, l'exécution du programme s'interrompra à la ligne où il a été rencontré. Presser Reprise pour continuer.
- Sauter les lignes avec "/"… - Si une ligne commençant par / est rencontrée et que cette option est cochée, cette ligne sera sautée.
- Vider l'historique du MDI… - Efface l'historique des données manuelles.
- Copier depuis l'historique du MDI… - Copier l'historique des données manuelles dans le presse-papier.
- Coller dans l'historique du MDI… - Coller le contenu du presse-papier dans la fenêtre d'historique des données manuelles.
- Calibration - Lance l'assistant de réglage de PID Servo. La calibration lit le fichier HAL et pour chaque pas il utilise une variable de la section [AXIS_n] du fichier ini et crée une entrée pouvant être éditée et testée.
- Afficher configuration de HAL… - Ouvre une fenêtre sur la configuration de HAL depuis laquelle il est possible de visualiser tous les Components, Pins, Parameters, Signals, Functions et Threads de HAL.

- HAL Mètre… - Ouvre une fenêtre dans laquelle il est possible de visualiser un seul Signal, HAL Pin, ou Parameter de HAL.
- HAL Scope… - Ouvre un oscilloscope virtuel qui permet de tracer dans le temps, les valeurs de HAL.
- Afficher l'état de LinuxCNC… - Ouvre une fenêtre montrant l'état de LinuxCNC.
- Choisir le niveau de Debug… - Ouvre une fenêtre dans laquelle les niveaux de débogage sont visibles et certains réglables.
- Prise d'origine… - Effectue la prise d'origine machine d'un ou de tous les axes.
- Annulation OM… - Annule les origines d'un ou de tous les axes.
- Annulation décalages d'origine… - Annule les décalages d'origine du système de coordonnées choisi.

- L'outil touchera la pièce… - Lorsqu'un Toucher est effectué, la valeur entrée est relative au système de coordonnées pièce actuel (G5x), tel que modifié par le décalage d'axe (G92). Quand la séquence de Toucher est complète, la coordonnée relative pour l'axe choisi prendra la valeur entrée. Voir aussi G10 L10 dans le chapitre du G-code.

  L'outil touchera le porte-pièce…
  Lorsqu'un Toucher est effectué, la valeur entrée est relative au 9ème système de coordonnées (G59.3), le décalage d'axe (G92) est ignoré. Mode destiné aux machines possédant un porte-pièce référencé à un endroit, sur lequel s'effectue le Toucher. Le 9ème système de coordonnées doit être ajusté pour que la pointe d'un outil de longueur nulle (le nez de broche), soit à l'origine du porte-pièce quand les coordonnées relatives sont à 0. Voir aussi G10 L11 dans le chapitre du G-code.

Tout est dans le point de vue…

Les icônes de choix du type d'affichage et du menu Vues d'AXIS se réfèrent à des Vue de dessus, Vue de face et Vue de côté. Ces termes sont corrects si la machine CNC a un axe Z vertical, avec une valeur de Z positive en haut. C'est vrai pour les fraiseuses verticales, qui sont probablement les plus populaires, c'est également vrai pour toutes les machines d'électro-érosion et aussi les tours verticaux, sur lesquels la pièce tourne sous l'outil.

Les termes Vue de dessus, Vue de face et Vue de côté sont cependant source de confusion sur d'autres machines CNC, comme un tour standard, sur lequel l'axe Z est horizontal, ou sur une fraiseuse horizontale, qui a également l'axe Z horizontal, ou même un tour vertical inversé, sur lequel la pièce tourne au dessus de l'outil et qui a son axe Z positif vers le bas!

Il faut juste se rappeler que l'axe Z est toujours parallèle a la broche et plus positif en s'éloignant de celle-ci. Etre familiarisé avec la cinématique de ses machines, permet d'interpréterez l'affichage comme il se doit.

- Vue de dessus… - La vue de dessus (ou vue de Z) affiche l'aspect du G-code vu depuis le côté positif de l'axe Z et en regardant vers son côté négatif. Cette vue convient bien pour visualiser les axes X et Y.
- Vue de dessus basculée… - La vue de dessus basculée (ou vue de Z basculé) affiche également l'aspect du G-code vu depuis le côté positif de l'axe Z et en regardant vers son côté négatif. Mais cette fois, les axes X et Y sont représentés pivotés de 90 degrés pour mieux occuper l'espace d'affichage. Cette vue convient bien également, pour visualiser les axes X et Y.
- Vue de côté… - La vue de côté (ou vue de X) affiche l'aspect du G-code vu depuis le côté positif de l'axe X et en regardant vers son côté négatif. Cette vue convient pour visualiser les axes Y et Z.

- Vue de face… - La vue de face (ou vue de Y) affiche l'aspect du G-code vu depuis le côté positif de l'axe Y et en regardant vers son côté négatif. Cette vue convient bien pour visualiser les axes X et Z.
- Vue en perspective… - La vue en perspective (ou vue P) affiche l'aspect du G-code en regardant vers la pièce depuis un point de vue orientable, par défaut vers X+, Y-, Z+. Cette position est orientable en la sélectionnant à la souris. L'affichage est un compromis, il tente d'afficher en 3D, entre trois et neuf axes, sur un écran en deux dimensions. Il y aura donc souvent certaines caractéristiques difficiles à voir, ce qui requerra un changement de point de vue. Cette vue convient bien pour voir les trois axes à la fois.
- Affichage en pouces… - Ajuste l'échelle d'affichage d'AXIS pour les pouces.
- Affichage en mm… - Ajuste l'échelle d'affichage d'AXIS pour les millimètres.
- Afficher le programme… - L'affichage à l'écran de l'aspect du G-code peut être entièrement désactivé si l'opérateur le souhaite.
- Parcours d'outil en vitesse rapide… - L'affichage du parcours d'outil du programme G-code courant représente toujours les mouvements en vitesse travail (G1,G2,G3) en blanc. Mais l'affichage des mouvements en vitesse rapide (G0) en cyan peut être désactivé si si l'opérateur le souhaite.
- Simulation de transparence… - Cette option rends plus lisible le tracé des parcours affichés par les programmes complexes, mais il peut rendre l'affichage plus lent.
- Parcours d'outil en temps réel… - La surbrillance des chemins d'outils en vitesse travail (G1,G2,G3) quand l'outil se déplace peut être désactivée si l'opérateur le souhaite.
- Afficher l'outil… - Le symbole d'un outil, représenté par un cône ou un cylindre peut être désactivé si l'opérateur le souhaite.
- Afficher les étendues… - L'affichage des étendues du programme G-code chargé (déplacements maximum de chacun des axes), peut être désactivé si l'opérateur le souhaite.
- Afficher les offsets… - L'emplacement de l'origine du système de coordonnées pièce (G54 à G59.3) peut être représenté par un jeu de trois lignes orthogonales, une rouge, une bleue et une verte. L'affichage de cette origine pièce (ou zéro pièce), peut être désactivé si l'opérateur le souhaite.
- Afficher les limites machine… - Les limites maximales de déplacement machine pour chacun des axes, qui sont fixées dans le fichier ini, s'affichent comme une boîte rectangulaire en lignes pointillées rouges. Il est facile, au chargement d'un nouveau programme G-code, de voir si la pièce est contenue dans le volume représenté. Ou de vérifier de combien l'étau doit être décalé, pour que le G-code puisse être usiné sans dépasser les limites de déplacements de la machine. Cette option peut être désactivée si l'opérateur le souhaite.
- Afficher la vitesse d'avance… - L'affichage de la vitesse peut être utile pour voir la précision avec laquelle la machine suit la vitesse commandée. Cette option peut être désactivée si l'opérateur le souhaite.
- Afficher la distance restante… - La distance restante est une valeur très utile à suivre, au lancement d'un programme de G-code inconnu pour la première fois. En combinaison avec les curseurs des correcteurs de vitesse, des dégâts sur l'outil ou la machine peuvent être évités. Quand le programme G-code sera débogué et qu'il fonctionnera en douceur, l'affichage de la distance restante pourra être désactivée si l'opérateur le souhaite.
- Coordonnées en police large… - Les coordonnées des axes et la vitesse d'avance, s'afficheront en police large dans la vue du parcours d'outil.
- Rafraîchir le parcours d'outil… - Au fur et à mesure des déplacements de l'outil, les parcours s'affichent sur l'écran d'Axis en surbrillance. Avant de répéter le programme, ou pour avoir un affichage clair sur une zone intéressante, la surbrillance des parcours précédents peut être rafraîchie.

- Afficher la position commandée… - C’est la position que LinuxCNC cherche à atteindre. Quand le mouvement est stoppé, c’est la position que LinuxCNC cherchera à maintenir.
- Afficher la position actuelle… - La position actuelle est la position mesurée grâce aux informations issues des codeurs ou simulées par le générateur de pas. Elle peut différer légèrement de la position commandée pour diverses raisons, comme les réglages des boucles PID, les contraintes physiques ou les efforts de coupe.
- Afficher la position machine… - C’est la position par rapport à l’origine machine, telle qu’établie par la prise d’origine machine (POM).
- Afficher la position relative… - C’est la position par rapport à l’origine pièce, telle qu’établie par la prise d’origine pièce (POP). On peut aussi représenter cette position comme étant l’origine machine à laquelle on a appliqué les codes de décalages des systèmes de coordonnées G5x, G92 et G43.

Menu Aide

- A propos d’Axis… - Donne la version et quelques informations relatives au copyright.
- Aide rapide… - Affiche la liste des raccourcis clavier.

4.1.3.2 Boutons de la barre d’outils

Signification des boutons de la fenêtre d’AXIS, de gauche à droite:

-  Arrêt d’urgence (A/U) (en Anglais, E-Stop)
-  Marche/Arrêt puissance machine
-  Ouvrir un fichier
-  Recharger le fichier courant
-  Départ cycle
-  Cycle en pas à pas
-  Pause/Reprise
-  Stopper l’exécution du programme
-  Sauter ou non les lignes commençant par **/**
-  Avec ou sans pause optionnelle
-  Zoom plus
-  Zoom moins

-  Vue prédéfinie **Z** (vue de dessus)
-  Vue prédéfinie **Z basculée**
-  Vue prédéfinie **X** (vue de côté)
-  Vue prédéfinie **Y** (vue de face)
-  Vue prédéfinie **P** (vue en perspective)
-  Orienter la vue avec le bouton gauche de la souris
-  Rafraîchir le parcours d'outil

4.1.3.3 Zones d'affichage graphique du programme

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.

 Symbole de prise d'origine 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.

 Symbole de limite d'axe.

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.

 Symbole d'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).

Vue du parcours d'outil

Quand un fichier est chargé, une vue du parcours 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 travail (tels ceux produits par une commande **G1**) sont affichés en lignes continues blanches. Les arrêts temporisés (tels ceux produits par la commande **G4**) sont représentés par une petite marque **X**.

Un mouvement G0 (Vitesse rapide) avant un déplacement en vitesse travail ne sera pas affiché sur l'écran des parcours d'outil. Un mouvement en vitesse rapide, après un appel d'outil T<n>, n'apparaîtra sur l'écran des parcours d'outil qu'après le mouvement en vitesse travail suivant. Pour contourner une de ces caractéristiques, programmer un G1 sans déplacement, juste avant le G0.

Étendues du programme

Les étendues du programme sont affichées pour chacun des axes. Aux extrémités, les coordonnées minimales et maximales sont indiquées. Au centres, la différence, entre ces deux coordonnées, est indiquée.

Quand une coordonnée dépasse la limite logicielle fixée dans le fichier .ini, la coordonnée correspondante s'affiche en rouge, entourée d'un rectangle. Dans la figure ci-dessous, la limite maximale est dépassée sur l'axe X, comme l'indique le rectangle entourant la valeur de la coordonnée. Le déplacement X minimal du programme est de -1.95, la course maximale est de 1.88 en X et le programme nécessite un déplacement en X de 3.83 pouces. Le déplacement total demandé par le programme est donc possible. Pour cela, se déplacer en jog vers la gauche puis Toucher à nouveau pour corriger l'origine pièce.

Figure 4.2: Limites logicielles

Le cône d'outil Si aucun outil n'est chargé, l'emplacement de la pointe de l'outil est indiqué par le cône d'outil. Le cône d'outil ne donne aucune 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 **T1 M6**, le cône d'outil passe de conique à cylindrique, il indique alors la proportion du diamètre de l'outil lu dans le fichier de la table d'outils.

Parcours d'outil Quand la machine se déplace, elle laisse une trace appelée le parcours d'outil. La couleur des lignes indique le type de mouvement: jaune pour les mouvementq jog, 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.

Interaction avec l'affichage Par un clic gauche sur une portion du parcours d'outil, la ligne sous la souris passe en surbrillance à la fois dans le parcours d'outil 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.

4.1.3.4 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 le parcours d'outil.

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.

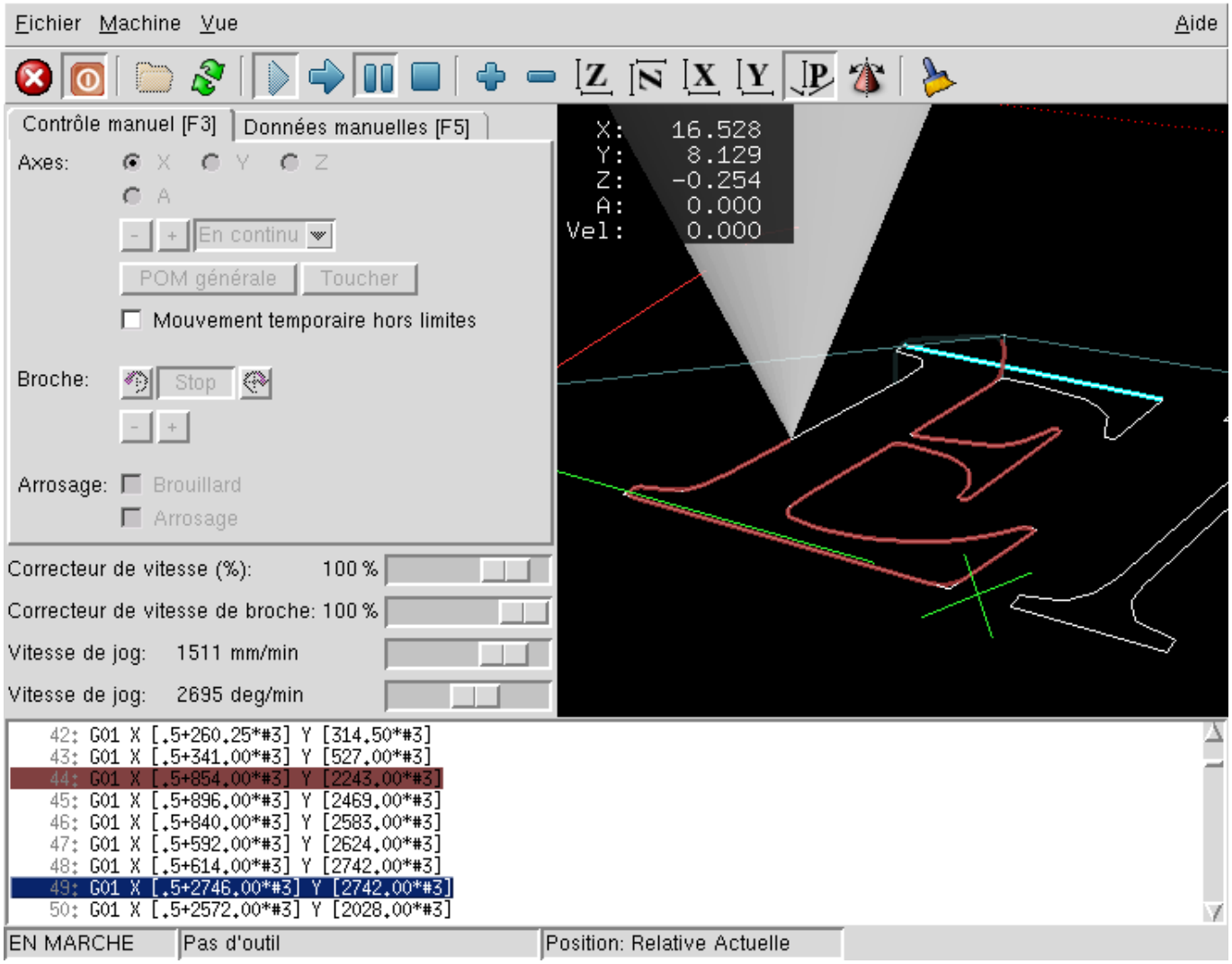


Figure 4.3: Ligne courante et ligne en surbrillance

4.1.3.5 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 la machine ou mettre en marche et arrêter ses différents organes.

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

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 spindle.0.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.

Le groupe de cases et boutons 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 sélectionner l'axe à actionner en cochant sa case. Puis cliquer sur 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, avec les touches 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
```

Voir le Manuel de l'intégrateur pour plus d'informations sur la configuration des incréments de jog.

Prise d'origine machine Si votre machine dispose de contacts d'origine machine et a une séquence de prise d'origine définie dans le fichier ini, le bouton POM générale lancera cette séquence pour tous les axes, les touches Ctrl-HOME auront le même effet.

Si votre machine dispose de contacts d'origine mais n'a pas de séquence de prise d'origine définie dans le fichier ini, le bouton POM générale effectuera uniquement la prise d'origine de l'axe sélectionné. Cette procédure doit alors être réalisée, séparément pour chacun des axes.

Si votre machine ne dispose d'aucun contact d'origine défini dans la configuration, le bouton POM générale définira la position actuelle de l'axe comme étant la position d'origine machine et l'axe sera marqué comme ayant sa prise d'origine machine faite.

Toucher

Si le bouton Toucher ou la touche FIN sont appuyés, le décalage d'origine pièce de l'axe Z, sur la figure ci-dessous: P1 G54, 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. Exemple, pour faire la prise d'origine pièce, on affleure l'outil sur une cale de 5mm d'épaisseur posée sur le bloc, on presse le bouton Toucher et on saisi 5 dans le champ de la boîte de dialogue. La pointe de l'outil sera alors référencée à 0 sur la surface du bloc.

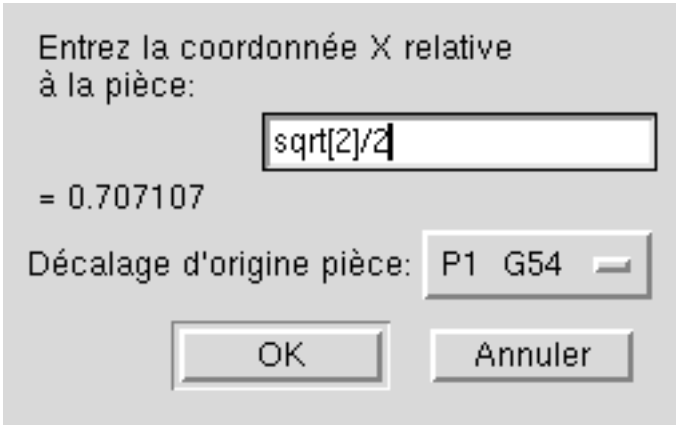


Figure 4.4: Fenêtre du Toucher

Voir aussi les options du menu Machine: Toucher la pièce et Toucher le porte-pièce.

Dépassement de limite En appuyant sur Dépassement de limite, la machine sera temporairement autorisée à se déplacer au delà d'un contact de limite physique. Cette case à cocher n'est disponible que lorsque un fin de course est pressé. Elle est désactivée après chaque mouvement de jogging. Si l'axe est configuré avec des contacts positifs et négatifs séparés, LinuxCNC permettra le jogging uniquement dans le sens du dégagement. Dépassement de limite ne permettra pas un jogging au delà

d'une limite logicielle. La seule façon de désactiver une limite logicielle sur un axe est d'annuler sa prise d'origine.

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.

Le groupe Arrosage

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

4.1.3.6 Données manuelles (MDI)

L'onglet d'entrée de données manuelles (encore appelé MDI), permet d'entrer et d'exécuter 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.

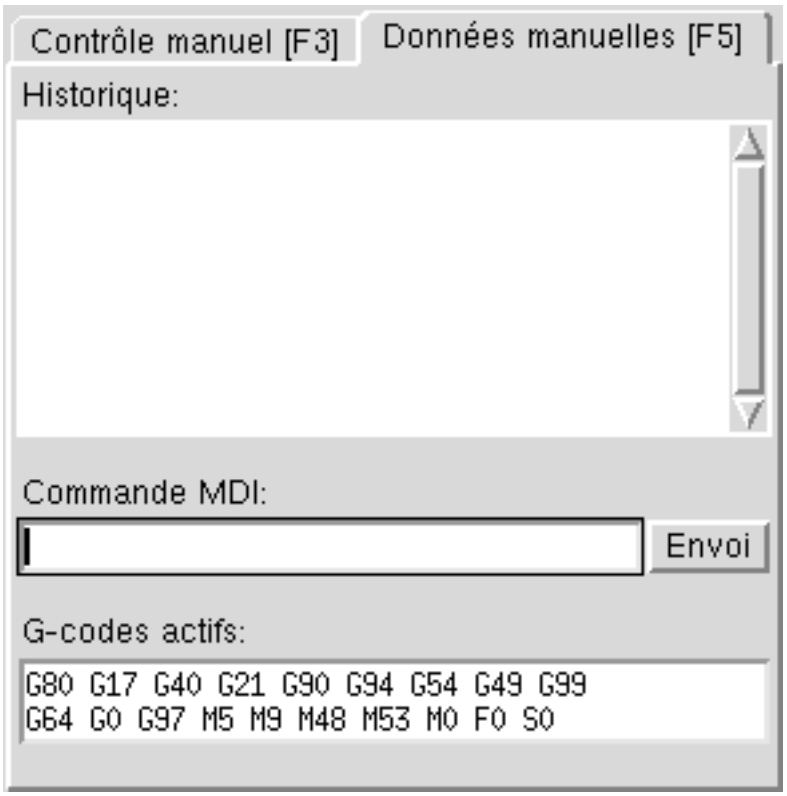


Figure 4.5: L'onglet Données manuelles

- Historique - Affiche les commandes précédemment tapées et au cours des session précédentes.
- Commande 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 appui sur le bouton Envoi.
- G-Codes actifs - Cette fenêtre affiche les codes modaux actuellement actifs dans l'interpréteur. Par exemple, **G54** indique que le système de décalage d'origine actuel est **G54** qui s'appliquera à toutes les coordonnées qui seront entrées.

4.1.3.7 Correcteurs de vitesse

En déplaçant le curseur, la vitesse de déplacement programmée peut être modifiée. Par exemple, si un programme requiert une vitesse à **F600** et que le curseur est placé sur 120%, alors la vitesse résultante sera de **F720**.

4.1.3.8 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 **F6400**. Cet élément n'apparaît que si la HAL pin spindle.0.speed-out est connectée dans le fichier .hal.

4.1.3.9 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).

4.1.3.10 Vitesse Maxi

En déplaçant ce curseur, la vitesse maximale peut être réglée. Ceci couvre la vitesse maximale de tous les mouvements programmés, sauf les mouvements avec broche synchronisée.

4.1.4 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.

Touches des correcteurs de vitesse.

- Les touches des correcteurs de vitesse se comportent différemment en mode manuel. Les touches 12345678 sélectionneront l'axe correspondant, si il est programmé.
- Si vous avez 3 axes, alors * sélectionnera l'axe 0, 1 sélectionnera l'axe 1, et 2 sélectionnera l'axe 2.
- Pendant l'exécution d'un programme, les touches 1234567890 fixeront la correction de vitesse travail entre 10% et 100%.

Les raccourcis clavier les plus fréquents sont visibles dans la table ci-dessous.

Table 4.1: Raccourcis clavier usuels

Touches	**Actions produites**	**Mode**
F1	Bascule l'arrêt d'urgence	Tous
F2	Bascule le marche/arrêt machine	Tous

Table 4.1: (continued)

Touches	**Actions produites**	**Mode**
`, 1 .. 9, 0	Correcteurs de vitesse de 10% à 100%	Varie
X, *	Active le premier axe	Manuel
Y, 1	Active le deuxième axe	Manuel
Z, 2	Active le troisième axe	Manuel
A, 3	Active le quatrième axe	Manuel
I	Sélection d'incrément du jog	Manuel
C	jog en mode continu	Manuel
Ctrl+origine	Lance une séquence de POM	Manuel
Fin	Toucher: valide l'offset G54 de l'axe actif	Manuel
Gauche, Droite	Jog du premier axe	Manuel
Up, Down	Jog du deuxième axe	Manuel
Pg Up, Pg Dn	Jog du troisième axe	Manuel
[,]	Jog du quatrième axe	Manuel
O	Ouvrir un fichier	Manuel
Ctrl+R	Recharger le fichier courant	Manuel
R	Exécuter le programme	Manuel
P	Pause dans l'exécution du programme	Auto
S	Reprise de l'exécution du programme	Auto
ESC	Stopper l'exécution	Auto
Ctrl+K	Rafraichi le tracé d'outil	Auto/Manuel
V	Défilement cyclique des vues prédéfinies	Auto/Manuel
Maj-gauche,droite	Axe X vitesse rapide	Manuel
Maj-haut, bas	Axe Y vitesse rapide	Manuel
Maj-PgUp, PgDn	Axe Z vitesse rapide	Manuel

4.1.5 Affichage de l'état de LinuxCNC (LinuxCNCtop)

AXIS inclut un programme appelé linuxcnctop qui affiche en détail l'état de LinuxCNC. Ce programme est accessible dans le menu Machine > Fenêtre d'état de LinuxCNC.

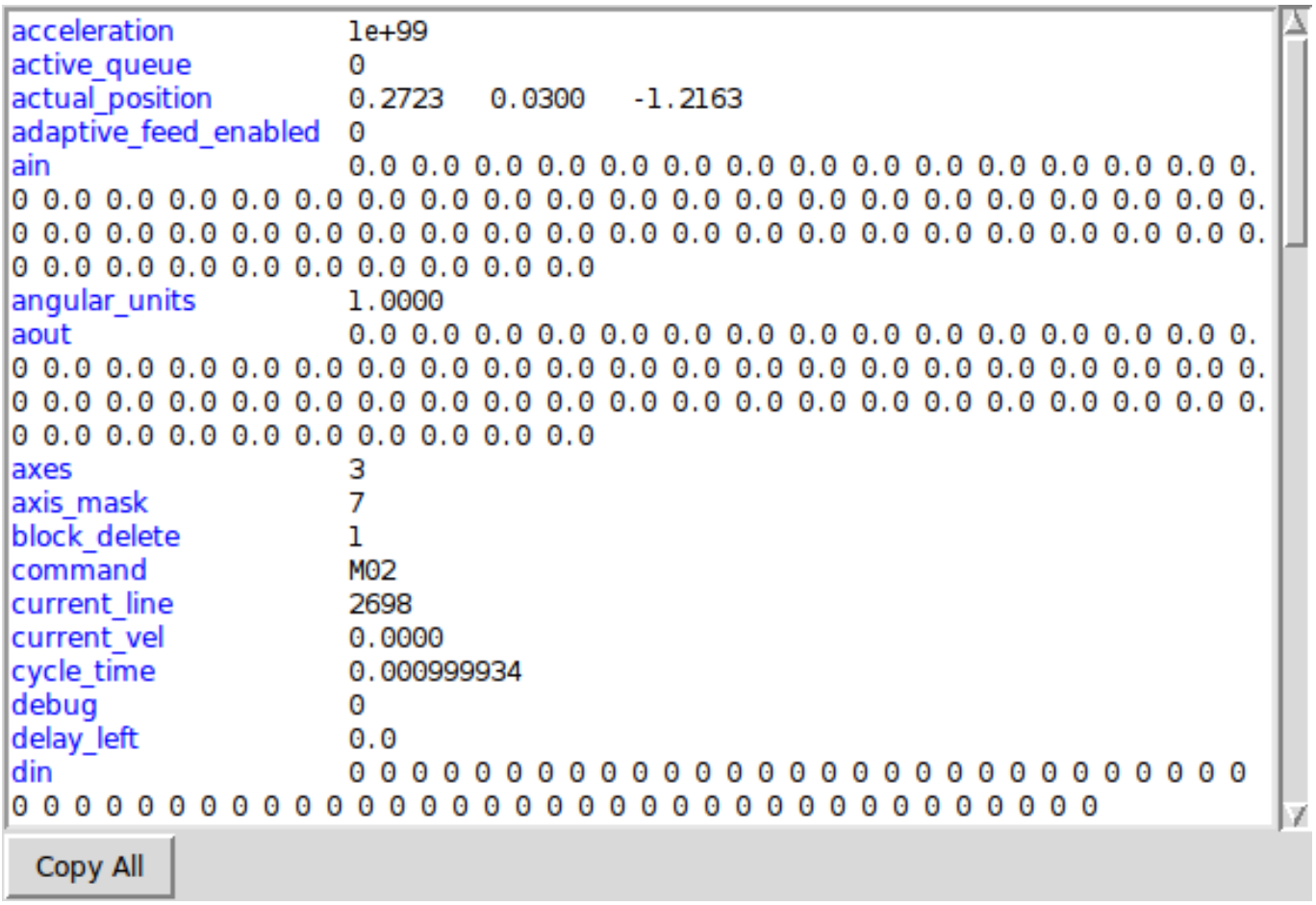


Figure 4.6: Fenêtre d'état de LinuxCNC

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.

4.1.6 Entrée de données en texte (MDI)

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

```
mdi /chemin/vers/linuxcnc.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 LinuxCNC qui l'exécute.

Voici un exemple de session MDI.

```
$ MDI ~/linuxcnc/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)
```

4.1.7 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).

4.1.8 hal_manualtoolchange: Dialogue de changement d'outil manuel

LinuxCNC inclut un composant userspace HAL de appelé hal_manualtoolchange, il ouvre une fenêtre d'appel d'outil visible ci-dessous, quand la commande **M6** est invoquée. Dés que le bouton Continuer est 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. Cette composante est particulièrement utile si vous avez des outils de pré-réglage et que vous utilisez la table d'outils.

Important
Le parcours d'outil d'un mouvement en vitesse rapide ne sera pas visible s'il suit un changement d'outil T<n> et ce jusqu'au prochain mouvement en vitesse travail. Ceci peut être source de confusion pour l'opérateur. Pour contourner cette particularité, ajoutez toujours un G1 sans mouvement après un M6 T<n>.

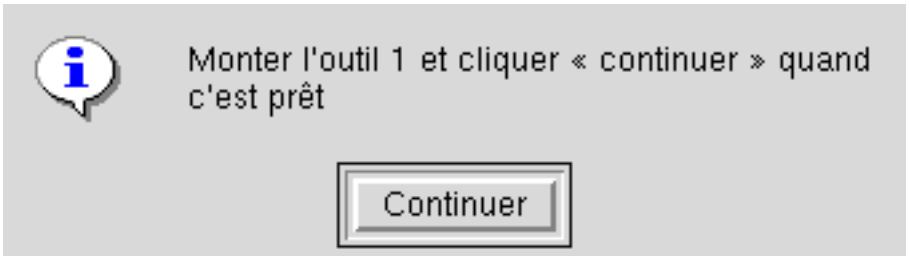


Figure 4.7: La fenêtre de changement manuel d'outil

4.1.9 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:

- LinuxCNC fournit l'accès aux commandes de LinuxCNC, 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 de LinuxCNC, ça se fera automatiquement. Avec une version installée en in-place, ça peut être fait avec l'aide de: /scripts/rip-environment.

4.1.10 Utiliser AXIS en mode tour

En incluant la ligne

```
LATHE = 1
```

dans la section [DISPLAY] du fichier ini, AXIS sélectionnera le mode tour. L'axe Y ne s'affiche pas parmi les coordonnées, la vue est modifiée pour placer la broche à gauche, l'axe Z horizontalement avec son côté positif vers la droite **(Z+)** et l'axe X s'étendant vers le bas de l'écran **(X+)**. Plusieurs contrôles (tels que ceux des vues prédéfinies) sont supprimés. Les lectures de coordonnées pour X sont désormais en diamètre et en rayon.

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

En mode tour (lathe), la forme et l'orientation de l'outil chargé sont représentés.

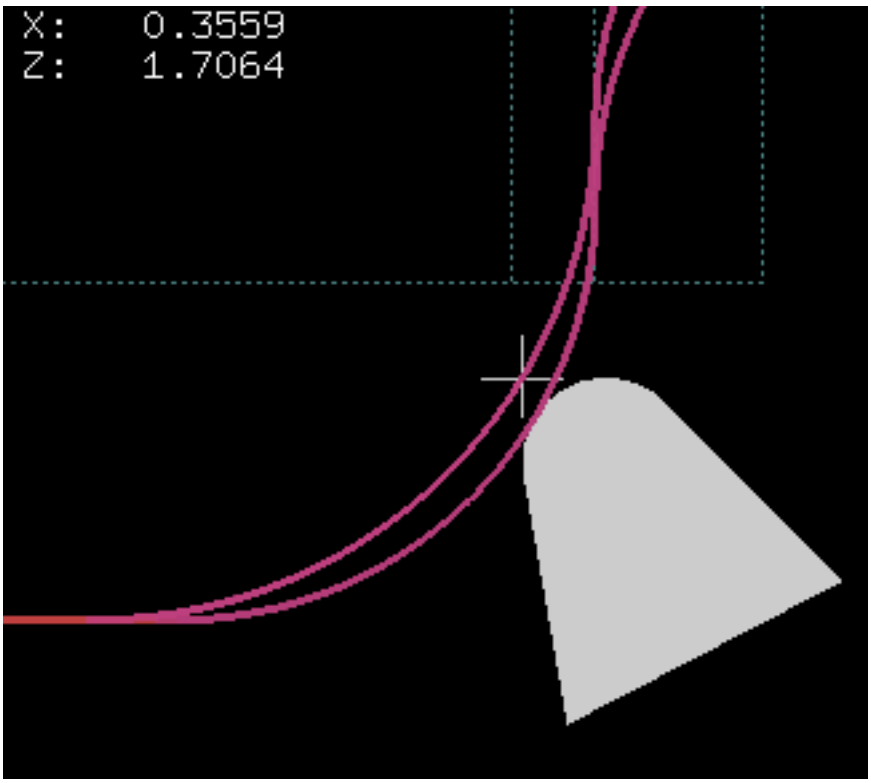


Figure 4.8: Représentation de l'outil en mode tour

4.1.11 Configuration avancée d'AXIS

Pour plus d'informations sur les paramètres du fichier ini pouvant modifier le fonctionnement d'AXIS, voir le fichier ini, sections [DISPLAY] et le chapitre sur la configuration dans le manuel de l'intégrateur.

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

```
[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 suivant un arc de cercle.

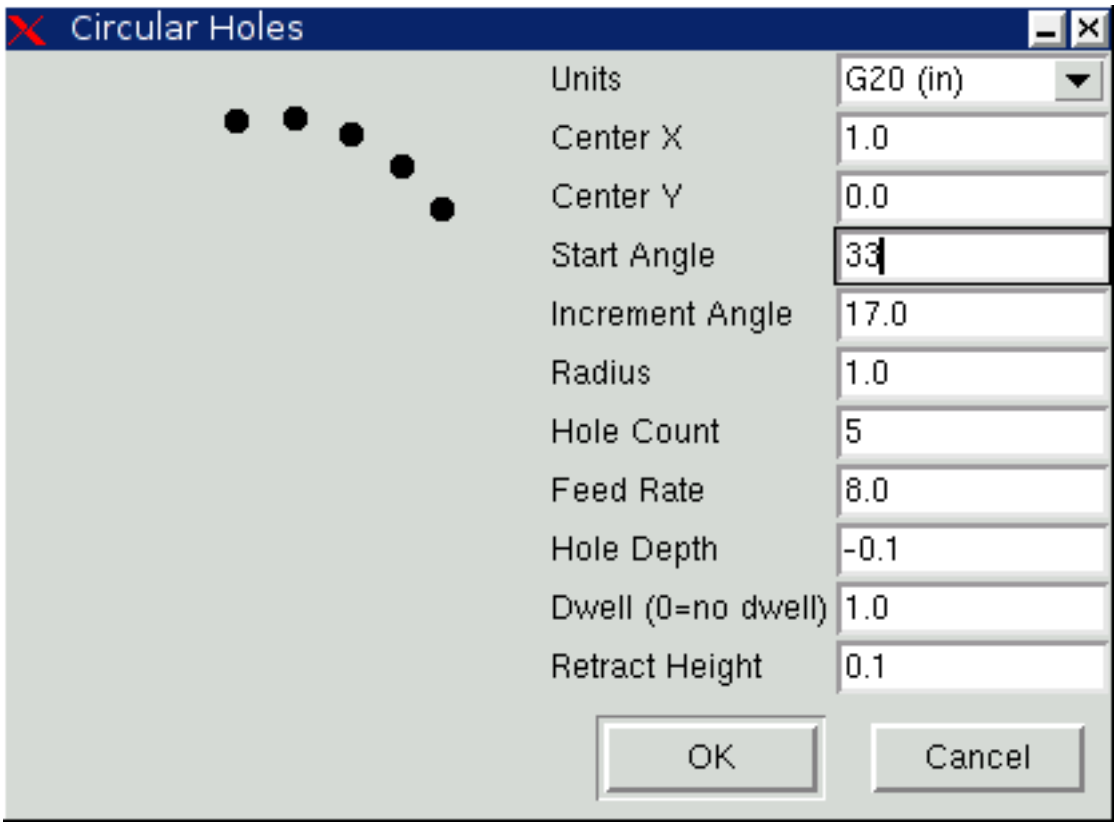


Figure 4.9: Perçages circulaires

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.

4.1.11.2 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 du parcours d'outil sur le modèle lignes noires et fond blanc, il sert aussi de référence des éléments configurables dans l'écran graphique. L'exemple de fichier axis_big_dro évolution de la position de lecture à une police plus grande taille. Pour utiliser ces fichiers:

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

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

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.

4.1.11.3 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.

4.1.11.4 ~/.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 ci-dessous ajoutent un Ctrl+Q comme raccourci clavier pour Quitter et activer l'option Distance restante par défaut.

Exemple de fichier .axisrc

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

4.1.11.5 Éditeur externe

En définissant: [DISPLAY]EDITOR , les options de menu: Fichier → Éditer ainsi que Fichier → Éditer la table d'outils, deviennent accessibles. Deux valeurs qui fonctionnent bien: EDITOR=gedit et EDITOR=gnome-terminal -e nano.

4.1.11.6 Panneau de contrôle virtuel

AXIS peut afficher un panneau de commande virtuel personnalisé dans le volet de droite. Il est possible d'y placer des boutons, des indicateurs qui afficheront des données et plus encore. Voir le manuel de l'intégrateur.

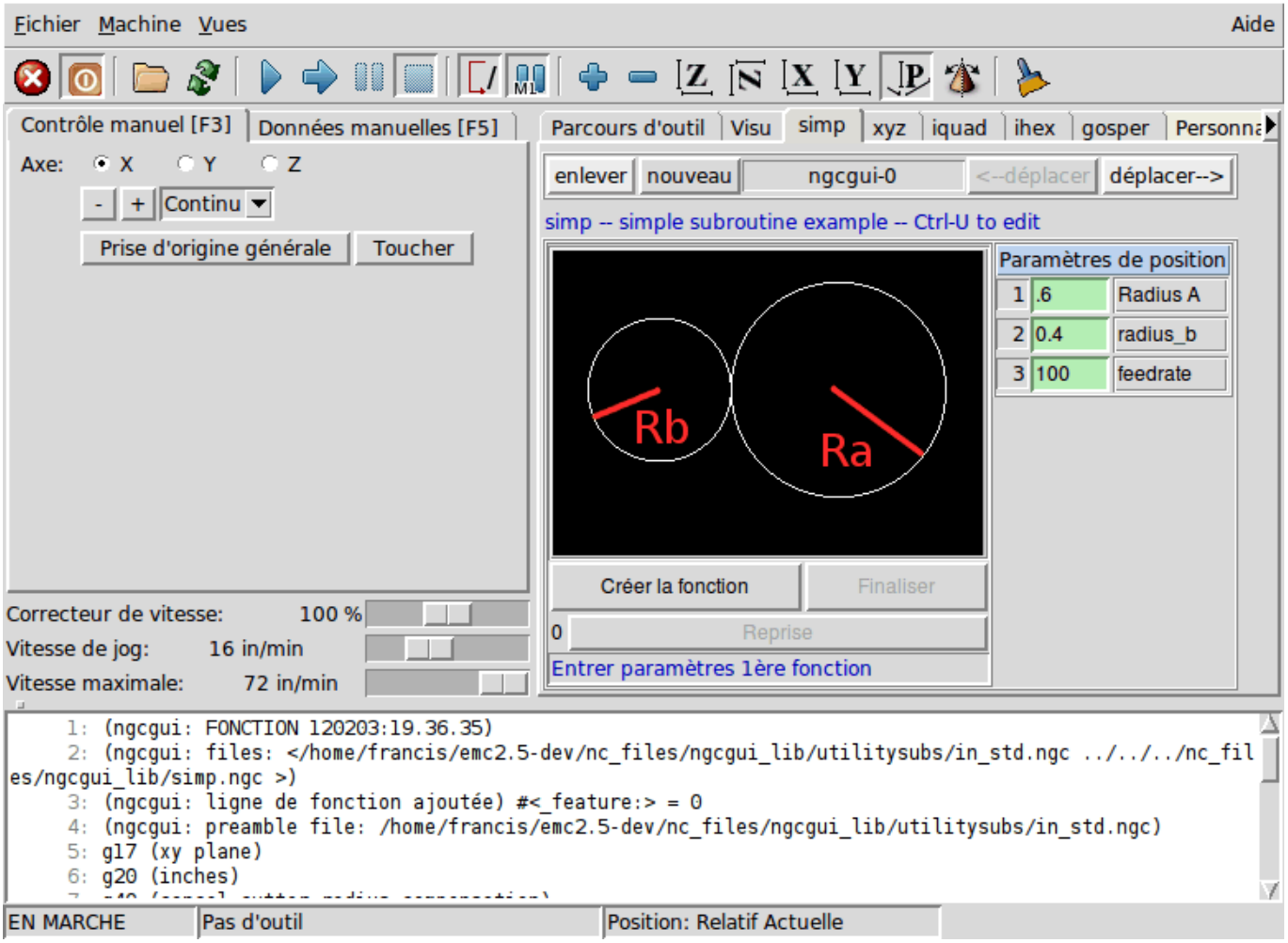
4.1.11.7 Commentaires spéciaux

Les commentaires spéciaux peuvent être insérés dans le fichier de G-code pour contrôler le comportement de l'affichage d'AXIS. Pour limiter l'aperçu au seul affichage du parcours d'outil, utiliser ces commentaires spéciaux. Rien ne s'affichera entre le commentaire (AXIS,hide) et le commentaire (AXIS,show) sauf le parcours d'outil. Les (AXIS,hide) et (AXIS,show) doivent être utilisés par paires avec le (AXIS, hide) en premier. Tout ce qui est après un (AXIS,stop) ne sera pas visible.

Ces commentaires sont utiles pour désencombrer l'affichage d'aperçu (Par exemple pendant le débogage d'un grand fichier G-code, on peut désactiver l' aperçu sur certaines parties qui sont déjà fonctionnelles).

- (AXIS,hide) Arrête le parcours d'outil (à placer en premier)
- (AXIS,show) Reprend le parcours d'outil (il faut suivre un cache)
- (AXIS,stop) Arrête le parcours d'outil d'ici à la fin du fichier.
- (AXIS,notify,le_texte) Affiche le_texte à l'écran, comme une info. Cet affichage peut être très utile lors du pré-affichage du parcours d'outil, quand les commentaires (debug,message) ne sont pas affichés.

4.2 L’utilitaire graphique NGCGUI



4.2.1 Vue d’ensemble

- NGCGUI est un utilitaire pour écrire et utiliser les sous-programmes avec LinuxCNC.
- NGCGUI peut être utilisé en autonome ou intégré, dans ce dernier cas, il crée des onglets multiples sur la page de l’interface graphique Axis.

Ngcgui est un outil puissant pour construire les programmes de G-code en sous-programmes.

- Les sous-programmes peuvent être concaténés pour fournir un programme de G-code complet.
- De multiples instances d’un sous-programme peuvent être utilisées pour fournir la même tâche à différents emplacements sur la même pièce.
- N’importe quel G-code valide peut être utilisé dans un sous-programme.
- De nouveaux sous-programmes peuvent être ajoutés à la volée.

4.2.2 Configurations fournies en exemple.

Trois configurations sont fournies, elles se trouve dans le répertoire sim du sélecteur de configuration de LinuxCNC. Le sélecteur de configuration se trouve quand à lui dans le menu Applications → CNC → LinuxCNC.

- ngcgui - Un exemple facile à comprendre utilisant ces sous-programmes:
  - simp - Un exemple simple créant deux cercles.
  - xyz - Crée une boîte basée sur deux coins opposés.
  - iquad - Crée un quadrilatère interne.
  - db25 - Crée la découpe pour une fiche DB25.
  - ihex - Crée un hexagone interne.
  - gosper - Une démo sur la récursion.
  - Custom - Crée des onglets personnalisés.
  - ttt - Traceur True Type, pour créer des textes à graver.
- ngcgui-lathe - Un exemple de sous-programme pour un tour:
  - id - Alésage intérieur.
  - od - Cylindrage extérieur.
  - taper-od - Tourne un cône mâle.
  - Custom - Crée des onglets personnalisés.
- ngcgui-simple - Un exemple simple:
  - simp - Un exemple simple créant deux cercles.
  - xyz - Crée une boîte basée sur deux coins opposés.

Pour visualiser les sous-programmes presser l'**A**rrêt d'**U**rgence  puis, activer la Marche Machine  et réaliser la Prise d'origine générale. Cliquer sur un onglet de ngcgui et presser Créer la fonction puis Finaliser. Enfin, presser sur le bouton  Départ cycle pour exécuter le G-code.

Note
Les sous-programmes d'exemples fournis avec la distribution doivent tous fonctionner avec la configuration de la machine simulée. Un utilisateur doit toujours comprendre le comportement et les implications d'un programme avant de tenter de l'exécuter sur une machine réelle.

4.2.3 Librairies

Les configurations en simulation pour ngcgui utilisent les liens suivants vers des librairies de LinuxCNC protégées en écriture:

- Sous-fichiers compatibles ngcgui - ngcgui_lib
- Sous-programme d'aide - ngcgui_lib/utilitysubs
- Fichiers M utilisateurs - ngcgui_lib/mfiles

Ces librairies sont définies dans le fichier ini par les items:

```
[RS274NGC]
SUBROUTINE_PATH = ../../../nc_files/ngcgui_lib:../../../nc_files/ngcgui_lib/utilitysubs
USER_M_PATH     = ../../../nc_files/ngcgui_lib/mfiles
```

C'est une longue ligne (ne pas continuer sur de multiples lignes) qui spécifie les répertoires utilisés dans le chemin de recherche. Les noms de répertoires sont séparés par le caractère (:).

L'utilisateur peut créer de nouveaux répertoires pour ses propres sous-programmes et fichiers M et les ajouter dans le chemin de recherche.

Par exemple, un utilisateur pourrait créer ces répertoires à partir de la console.

```
mkdir /home/myusername/mysubs
mkdir /home/myusername/mymfiles
```

Puis y créer ou y copier des fichiers qui lui seront accessibles en écriture. Par exemple, créer un sous-fichier compatible ngcgui nommé:

```
/home/myusername/mysubs/exemple.ngc
```

Le fichier ini doit être édité pour lui inclure les nouveaux sous-fichiers et les ajouter au chemin. Pour cet exemple:

```
[RS274NGC]
SUBROUTINE_PATH = /home/myusername/mysubs:../../../nc_files/ngcgui_lib:../../../nc_files/ ↩
    ngcgui_lib/utilitysubs
USER_M_PATH     = /home/myusername/mymfiles:../../../nc_files/ngcgui_lib/mfiles

[DISPLAY]
NGCGUI_SUBFILE = exemple.ngc
```

LinuxCNC et ngcgui utilisent le premier fichier trouvé lors d'une recherche dans les répertoires du chemin de recherche. Avec ce comportement, Il est possible de substituer un sous-fichier ngcgui_lib en plaçant un sous-fichier avec un nom identique plus tôt dans le chemin de recherche pour qu'il soit trouvé avant. Plus d'informations peuvent être trouvées au chapitre INI dans le Manuel de l'intégrateur.

4.2.4 Intégration de ngcgui dans Axis

D'autres exemples de sous-programmes se trouvent dans le répertoire sim/ngcgui

Les items de fichier INI pour NGCGUI vont dans la section [DISPLAY].

- TKPKG = Ngcgui 1.0 - Le paquet principal de NGCGUI (doit précéder Ngcguittt).
- TKPKG = Ngcguittt 1.0 - Le paquet du traceur True Type pour générer des textes à graver.
- NGCGUI_FONT = Helvetica -12 normal - Spécifie la police utilisée.
- NGCGUI_PREAMBLE = in_std.ngc - Le fichier de préambule à ajouter au début du sous-programme. Quand plusieurs sous-programmes sont concaténés, un seul est ajouté.
- NGCGUI_SUBFILE = simp.ngc - Crée un onglet depuis le sous-programme nommé.
- NGCGUI_SUBFILE = "" - Crée un onglet personnalisé.
- #NGCGUI_OPTIONS = opt1 opt2 … - Options Ngcgui
  - # opt items:

  - # nonew — interdit la création d'un nouvel onglet personnalisé
  - # noremove — interdit l'effacement d'une page d'onglet
  - # noauto — pas d'envoi auto (makeFile, puis envoi manuel)
  - # noiframe — no internal image, image on separate top level

- TTT = Le programme True-type Tracer
- TTT_PREAMBLE = in_std.ngc - Optionnel, spécifie le nom de fichier de préambule utilisé par ttt pour créer les sous-fichiers.

Voici un exemple d'intégration de NGCGUI dans Axis. Les sous-programmes doivent être placés dans un répertoire spécifié par la variable [RS274NGC]SUBROUTINE_PATH. Certains exemples de sous-programmes utilisent d'autres sous-programmes, bien vérifier pour être sur d'avoir les bonnes dépendances, le cas échéant, dans un répertoire SUBROUTINE_PATH. Certains sous-programmes peuvent utiliser des fichiers M (Mfiles) personnalisés qui doivent se trouver dans un répertoire spécifié par [RS274NGC]USER_M_PATH.

Note
Il ne s'agit pas d'un fichier ini complet, les items montrés sont ceux utilisés par ngcgui. D'autres items sont requis par LinuxCNC pour obtenir un fichier ini complet.

Simple fichier.ini

```
[RS274NGC]
SUBROUTINE_PATH   = ../../../nc_files/ngcgui_lib:../../../ngcgui_lib/utilitysubs
USER_M_PATH       = ../../../nc_files/ngcgui_lib/mfiles

[DISPLAY]
TKPKG             = Ngcgui    1.0
TKPKG             = Ngcguittt 1.0
# Ngcgui must precede Ngcguittt

NGCGUI_FONT       = Helvetica -12 normal
# specifie seulement les noms de fichiers, doit être dans [RS274NGC]SUBROUTINE_PATH
NGCGUI_PREAMBLE   = in_std.ngc
NGCGUI_SUBFILE    = simp.ngc
NGCGUI_SUBFILE    = xyz.ngc
NGCGUI_SUBFILE    = iquad.ngc
NGCGUI_SUBFILE    = db25.ngc
NGCGUI_SUBFILE    = ihex.ngc
NGCGUI_SUBFILE    = gosper.ngc
# specifie "" pour une page d'onglet personnalisée
NGCGUI_SUBFILE    = ""
#NGCGUI_SUBFILE   = "" utilisé quand une trame d'image est spécifiée si
#                     ouvrir d'autres fichiers est requis
#                     les images seront mises dans une fenêtre de haut niveau
NGCGUI_OPTIONS    =
#NGCGUI_OPTIONS   = opt1 opt2 ...
# opt items:
#   nonew      -- interdit la création d'un nouvel onglet personnalisé
#   noremove   -- interdit l'effacement d'une page d'onglet
#   noauto     -- pas d'envoi auto (makeFile, puis envoi manuel)
#   noiframe   -- no internal image, image on separate top level

TTT               = truetype-tracer
TTT_PREAMBLE      = in_std.ngc

PROGRAM_PREFIX    = ../../nc_files
```

4.2.4.1 Traceur Truetype

Ngcgui_ttt fourni le support pour truetype-tracer (v4). Il crée un onglet sur Axis qui permet à l'utilisateur de créer ses propres textes dans de nouveaux onglets ngcgui et en choisissant leurs fontes et autres paramètres. (Truetype-tracer doit être installé indépendamment).

L'intégration de ngcgui_ttt dans Axis, nécessite les items suivants en plus de ceux de ngcgui:

```
Item:    [DISPLAY]TKPKG = Ngcgui_ttt numéro_de_version
Exemple: [DISPLAY]TKPKG = Ngcgui_ttt 1.0
Note:    Obligatoire, spécifie le chargement de ngcgui_ttt dans un onglet
d'Axis nommé ttt. Doit suivre l'item TKPKG = Ngcgui.

Item:    [DISPLAY]TTT = chemein_de_truetype-tracer
Exemple: [DISPLAY]TTT = truetype-tracer
Note:    Optionnel, s'il n'est pas spécifié, utilisera
/usr/local/bin/truetype-tracer. Spécifier avec un chemin absolut ou simplement
le nom de l'exécutable, dans ce cas, la variable d'environnement PATH de
l'utilisateur sera utilisée pour rechercher le programme.

Item:    [DISPLAY]TTT_PREAMBLE = nom_fichier_préambule
Exemple: [DISPLAY]TTT_PREAMBLE = in_std.ngc
Note:    Optionnel, spécifie le nom du fichier de préambule utilisé pour
les sous-fichiers créés par ttt.
```

4.2.4.2 Exemples d'INI

Ngcgui utilise le chemin de recherche de LinuxCNC pour chercher les fichiers.

Le chemin de recherche commence avec le répertoire standard spécifié par:

```
[DISPLAY]PROGRAM_PREFIX
```

suivi par les répertoires multiples spécifiés par:

```
[RS274NGC]SUBROUTINE_PATH
```

Répertoires Les répertoires peuvent être spécifiés comme des chemins absolus ou des chemins relatifs.

```
Exemple: [DISPLAY]PROGRAM_PREFIX = /home/myname/linuxcnc/nc_files
Exemple: [DISPLAY]PROGRAM_PREFIX = ~/linuxcnc/nc_files
Exemple: [DISPLAY]PROGRAM_PREFIX = ../../../nc_files
```

Chemins absolus Un chemin absolu commence avec un "/" qui indique un emplacement par rapport au système de fichiers complet. Un chemin qui commence par "~/" indique un chemin commençant depuis le répertoire home de l'utilisateur. Un chemin qui commence par "~nomutilisateur/" indique un chemin commençant dans le répertoire utilisateur.

Chemins relatifs Un chemin relatif commence dans le répertoire de démarrage qui est celui contenant le fichier ini. L'usage des chemins relatifs facilite l'accès aux configurations mais requiert une bonne compréhension de la façon dont les chemins sont spécifiés sous Linux.

```
./d0      est le même que d0, ex: un répertoire nommé d0 dans le répertoire de
départ.
../d1     se réfère au répertoire d1 dans le répertoire parent.
../../d2  se réfère au répertoire d2 dans le répertoire parent du parent.
../../../d3 etc.
```

Des répertoires multiples peuvent être spécifiés par la variable: [RS274NGC]SUBROUTINE_PATH suivie des chemins séparés par le signe ":". L'exemple suivant illustre le format utilisé pour les chemins multiples et montre l'utilisation de répertoires relatifs et absolus.

Exemple: [RS274NGC]SUBROUTINE_PATH = ../../../nc_files/ngcgui_lib: ../../../nc_files/ngcgui_lib/utilitysubs

C'est une longue ligne, ne pas continuer sur de multiples lignes. Quand LinuxCNC et/ou Ngcgui cherchent un fichier, c'est le premier trouvé qui est utilisé.

LinuxCNC (et NGCGUI) doivent pouvoir trouver tous les sous-programmes avec les routines additionnelles qui sont appelées depuis les sous-fichiers NGCGUI. Il est pratique de placer les fichiers utilitaires dans un répertoire séparé comme indiqué dans l'exemple précédent.

La distribution inclus le répertoire ngcgui_lib et les fichiers de préambule , sous-fichiers, postambule et fichiers d'aide pour les démos. Pour modifier le comportement des fichiers, il est possible de copier n'importe quel fichier et de le placer en avant du chemin de recherche. Le premier répertoire recherché est: [DISPLAY]PROGRAM_PREFIX. Il est possible de l'utiliser mais c'est une meilleure pratique de créer un répertoire dédié en le plaçant au début du chemin donné par [RS274NGC]SUBROUTINE_PATH.

Dans l'exemple suivant, les fichiers dans /home/myname/emc2/mysubs seront trouvés avant ceux étant dans ../../../nc_files/ngcgui_lib.

Exemple: [RS274NGC]SUBROUTINE_PATH = /home/myname/emc2/mysubs:../../../nc_files/ ngcgui_lib:../../.

Les débutants pourraient essayer par inadvertance d'utiliser des fichiers non structurés comme le nécessite ngcgui. Ngcgui déclencherait alors rapidement de nombreuses erreurs si les fichiers ne répondent pas à ses conventions. Une bonne pratique suggère que les sous-fichiers compatibles ngcgui doivent être placés dans un répertoire dédié à cette fin et que les préambules, postambules et fichiers d'aide doivent être dans un répertoire séparés pour dissuader toute tentative d'utilisation de ces sous-fichiers.

Pour intégrer ngcgui dans Axis, spécifier les items suivants dans le fichier ini:

```
Item:    [DISPLAY]PROGRAM_PREFIX = dirname
Exemple: [DISPLAY]PROGRAM_PREFIX = ../../../nc_files
Note:    Obligatoire et nécessaire pour de nombreuses fonctions de LinuxCNC.
         C'est le premier répertoire utilisé lors de la recherche de fichiers.

Item:    [RS274NGC]SUBROUTINE_PATH = dirname1:dirname2:dirname3 ...
Exemple: [RS274NGC]SUBROUTINE_PATH = ../../../nc_files/ngcgui_lib:../../../ ↩
   nc_files/ngcgui_lib/utilitysubs
Note:    Optionnel, mais très utile pour organiser les sous-fichiers et les
         fichiers utilitaires.

Item:    [DISPLAY]TKPKG=Ngcgui version_number
Exemple: [DISPLAY]TKPKG=Ngcgui 1.0
Note:    Obligatoire, spécifie le chargement des onglets ngcgui dans axis.

Item:    [DISPLAY]NGCGUI_FONT = font_descriptor
Exemple: [DISPLAY]NGCGUI_FONT = Helvetica -12 normal
Note:    Optionnel, descripteur de fontes compatible avec celui de Tcl.
         Avec des items pour le type de fonte -fontsize fontweight
         Par défaut c'est la police: Helvetica -10 normal

Item:    [DISPLAY]NGCGUI_SUBFILE = subfile_filename
Exemple: [DISPLAY]NGCGUI_SUBFILE = simp.ngc
Exemple: [DISPLAY]NGCGUI_SUBFILE = xyz.ngc
Exemple: [DISPLAY]NGCGUI_SUBFILE = ""
Note:    Utilise un ou plusieurs items pour spécifier les fichiers compatibles
         avec les sous-fichiers ngcgui qui requiert un onglet dans Axis au
         départ. Un onglet "personnalisé" est créé quand le nom de fichier est "".
         Un utilisateur peut utiliser l'onglet "Personnalisé" pour lire un
```

```
         fichier système et identifier un préambule, un sous-fichier ou un
         postambule.

Item:    [DISPLAY]NGCGUI_PREAMBLE = preamble_filename
Exemple: [DISPLAY]NGCGUI_PREAMBLE = in_std.ngc
Note:    Optionnel, si spécifié, alors ce fichier sera prépondérant à tous les
         sous-fichiers. Les fichiers créés avec l'onglet "Personnalisé"
         utilisent le préambule spécifié avec cette page.

Item:    [DISPLAY]NGCGUI_POSTAMBLE = postamble_filename
Exemple: [DISPLAY]NGCGUI_POSTAMBLE = bye.ngc
Note:    Optionnel, si spécifié, le fichier est ajouté à tous les sous-fichiers.
         Les fichiers créés avec l'onglet "Personnalisé" utilisent le postambule
         spécifié avec cette page.

Item:    [DISPLAY]NGCGUI_OPTIONS = opt1 opt2 ...
Exemple: [DISPLAY]NGCGUI_OPTIONS = nonew noremove
Note:    De multiples options séparées par des blancs.
         Par défaut,  ngcgui gère les onglets de cette manière:
            1) Un utilisateur peut créer de nouveaux onglets.
            2) Un utilisateur peut enlever des onglets (excepté le dernier
         restant)
            3) La finalisation envoie automatiquement les fichiers à Axis.
            4) Une trame d'image (iframe) est rendue disponible pour afficher
            une image pour le sous-fichier.

         Les options _nonew_, _noremove_, _noauto_, _noiframe_ respectivement,
         désactivent ces comportements par défaut.

         Par défaut, Si un fichier d'image (.png, .gif, .jpg, .pgm)
         est trouvé dans le même répertoire que le sous-fichier, l'image
         est affichée dans une iframe.  Spécifier l'option
         _noiframe_ rendra disponibles d'autres boutons pour sélectionner
         un préambule, un sous-fichier ou un postambule et
         des cases à cocher additionnelles. Les cases à cocher sont toujours
         disponibles avec les touches spéciales suivantes:
           Ctrl-R Bascule "Conserver les valeurs à la lecture du sous-fichier"
           Ctrl-E Bascule "Déployer le sous-programme"
           Ctrl-a Bascule "EnvoiAuto"
          (Ctrl-k lists all keys and functions)

         Si _noiframe_ est spécifié et qu'un fichier image est trouvé,
         l'image est affichée dans une fenêtre séparée et
         toutes les fonctions sont disponibles dans la page de l'onglet.

         Les _NGCGUI_OPTIONS_ s'appliquent à tous les onglets ngcgui excepté
         ceux sur lesquels les options _nonew_, _noremove_, et _noiframe_ ne
         sont pas applicables pour l'onglet "Personnalisé".
         Ne pas utiliser l'onglet "Personnalisé" si les utilisateurs doivent
         avoir des possibilités de sélection de sous-fichiers et de création
         d'onglet additionnels limitées.
```

4.2.5 Besoins des sous-programmes

Un sous-fichiers compatible NGCGUI contient une simple définition de sous-programme. Le nom du sous-programme doit être le même que celui du fichier (non inclus l'extension .ngc). LinuxCNC supporte les sous-programmes nommés ou numérotés, mais seuls les sous-programmes nommés sont compatible avec NGCGUI. Pour plus d'informations voir le chapitre sur les O-Codes.

La première ligne, autre qu'un commentaire, doit être une déclaration sub. La dernière ligne, autre qu'un commentaire, doit être une déclaration endsub.

exemple.ngc:

```
o<exemple> sub
  CORPS DU SOUS-PROGRAMME
o<exemple> endsub
```

Le corps du sous-programme doit commencer par un jeu de déclarations définissant les paramètres nommés locaux pour chaque paramètre positionnel attendu pour l'appel du sous-programme. Ces définitions doivent être consécutives, commencer par #1 et finir avec le numéro du dernier paramètre utilisé. Les définitions doivent être fournies pour chacun de ces paramètres (aucune omissions).

Numérotation des paramètres

```
#<xparm> = #1
#<yparm> = #2
#<zparm> = #3
```

LinuxCNC considère tous les paramètres numérotés entre #1 est #30 comme étant des paramètres appelables, de même, ngcgui fourni des dialogues de saisie pour n'importe quel paramètres dans cette fourchette. Il est de bonne pratique d'éviter d'utiliser un paramètre numéroté de #1 jusqu'à #30 n'importe où ailleurs dans le sous-programme. L'utilisation de paramètres nommés locaux est recommandée pour toutes le variables internes.

Chaque définition de déclaration peut optionnellement inclure un commentaire spécial et une valeur par défaut pour le paramètre.

Prototypage de déclaration

```
#<vname> = #n (=valeur_par_défaut)
ou
#<vname> = #n (texte_de_commentaire)
ou
#<vname> = #n (=valeur_par_défaut texte_de_commentaire)
```

Exemples de paramètres

```
#<xparm> = #1 (=0.0)
#<yparm> = #2 (Ystart)
#<zparm> = #3 (=0.0 Z start setting)
```

Si une valeur_par_défaut est donnée, elle sera placée au démarrage, dans la boîte de saisie pour le paramètre.

Si un texte_de_commentaire est inclus, il sera utilisé pour identifier l'entrée à la place du nom du paramètre.

Paramètres nommés globaux Note sur les paramètres nommés globaux (#<nom_global>) avec ngcgui:

Comme dans de nombreux langages de programmation, l'utilisation de variables globales est puissante, mais peut souvent mener à des conséquences inattendues. Dans LinuxCNC, les paramètres nommés globaux existants seront valides lors de l'exécution du sous-programme et les sous-programmes peuvent les modifier ou en créer.

L'utilisation de paramètres nommé globaux comme entrées dans un sous-programme est déconseillé parce-que de tels usages requiert l'établissement et la maintenance d'un contexte global bien défini, ce qui est problématique à maintenir. L'utilisation de paramètres numérotés en #1 et #30 devrait être suffisant pour satisfaire les besoins les plus exigeants.

Ngcgui supporte quelques entrées par paramètres nommés globaux mais leurs usage est obsolète et non documenté ici.

Bien que les entrées par paramètres nommés globaux soient déconseillées, les sous-programmes LinuxCNC doivent utiliser des paramètres nommés globaux pour retourner les résultats. Puisque les sous-fichiers compatibles ngcgui sont destinés à l'usage de l'interface graphique, les valeurs de retour n'ont pas d'exigence commune. Toutefois, ngcgui est utile comme outil de test pour les sous-programmes qui ne retournent pas de paramètres nommés globaux et il est commun pour les sous-fichiers compatibles ngcgui d'appeler des fichiers de sous-programmes utilitaires qui eux retournent des résultats avec des paramètres nommés globaux.

Pour supporter ces usages, ngcgui ignore les paramètres nommés globaux qui incluent le caractère (:) dans leur nom. Utilisation des deux points (:) dans un nom prévient ngcgui de créer une bîte de saisie pour ces paramètres.

Paramètres nommés globaux

```
o<exemp> sub
...
#<_exemp:result> = #5410       (retourne le diamètre de l'outil courant)
...
o<helper> call [#<x1>] [#<x2>] (appel d'un sous-programme)
#<xresult> = #<_helper:answer> (localise immédiatement le résultat du
fichier d'aide)
#<_helper:answer> = 0.0 (rend nul le paramètre nommé global utilisé par le
sous-programme)
...
o<exemp> endsub
```

Dans l'exemple précédent, le sous-programme utilitaire sera trouvé dans un fichier séparé nommé helper.ngc. Le sous-programme d'aide retourne un résultat dans un paramètre nommé global nommé #<_helper:answer>.

Pour une bonne pratique, le sous-fichier appelant localise immédiatement le résultat pour une utilisation ailleurs dans le sous-fichier et le paramètre nommé global, utilisé pour retourner le résultat est mis à zéro pour diminuer les chances qu'il soit utilisé par inadvertance ailleurs dans le contexte global. (La mise à zéro avec 0.0 n'est pas toujours le meilleur choix).

Ngcgui supporte la création et la concaténation de multiples fonctions pour un sous-fichier et pour de multiples sous-fichiers. Il est parfois pratique pour les sous-fichiers de déterminer leur ordre au début de l'exécution afin que ngcgui insère un paramètre global spécial qui pourra être testé par tous les sous-programmes. Ce paramètre est nommé #<_feature:>. Sa valeur commence avec 0 et est incrémentée avec chaque fonction qui lui est ajoutée.

Fonctions additionelles Un commentaire spécial info peut être inclus quelque part dans les sous-fichier compatibles ngcgui. Le format est le suivant:

```
(info: info_text)
```

La chaine info_text est affichée vers le haut de la page de l'onglet ngcgui dans Axis.

Les fichiers non destinés à servir de sous-fichiers peuvent inclure le commentaire spécial: "(not_a_subfile)" de sorte que ngcgui les rejette automatiquement avec un message explicatif.

```
(not_a_subfile)
```

Un fichier image optionnel (.png, .gif, .jpg, .pgm) peut accompagner un sous-fichier. Le fichier image peut aider à clarifier les paramètres utilisés par le sous-fichier. Le fichier image doit être dans le

même répertoire que le sous-fichier et doit avoir le même nom avec une extension appropriée au fichier image, ex: le sous-fichier exemp.ngc doit être accompagné par l'image exemp.png. Ngcgui tente de redimensionner de grandes images par sous-échantillonnage à une largeur maximale de 320 et une hauteur maximum de 240 pixels.

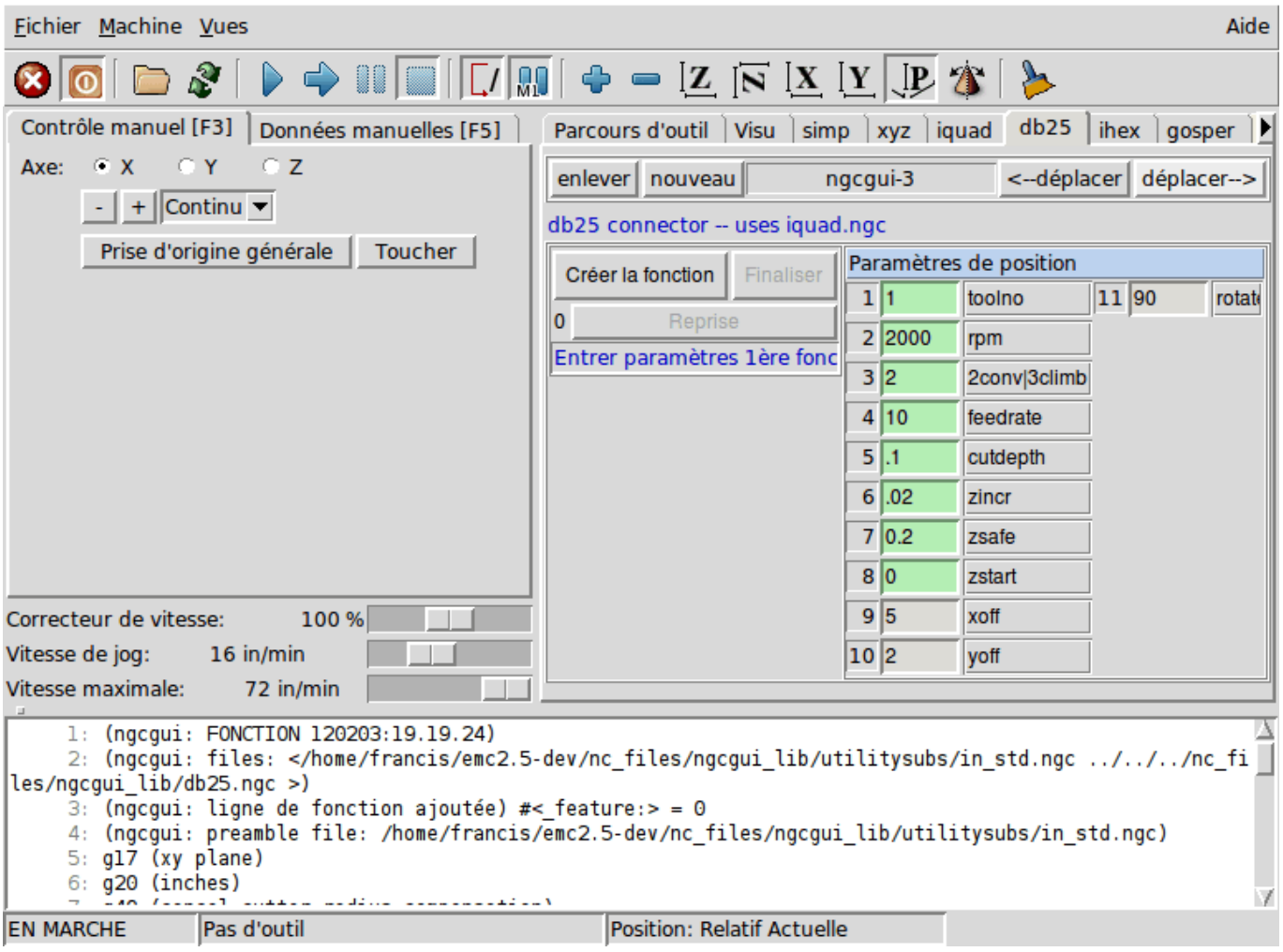
Aucune des conventions nécessaires pour faire une sous-fichier compatible ngcgui n'empêche son utilisation en tant que fichier de sous-programme pour LinuxCNC.

La distribution LinuxCNC inclus une librairie (répertoire ngcgui_lib) qui contient plusieurs exemples de sous-fichiers et de fichiers utilitaires compatibles ngcgui pour illustrer les fonctions des sous-programmes de LinuxCNC et l'usage de ngcgui.

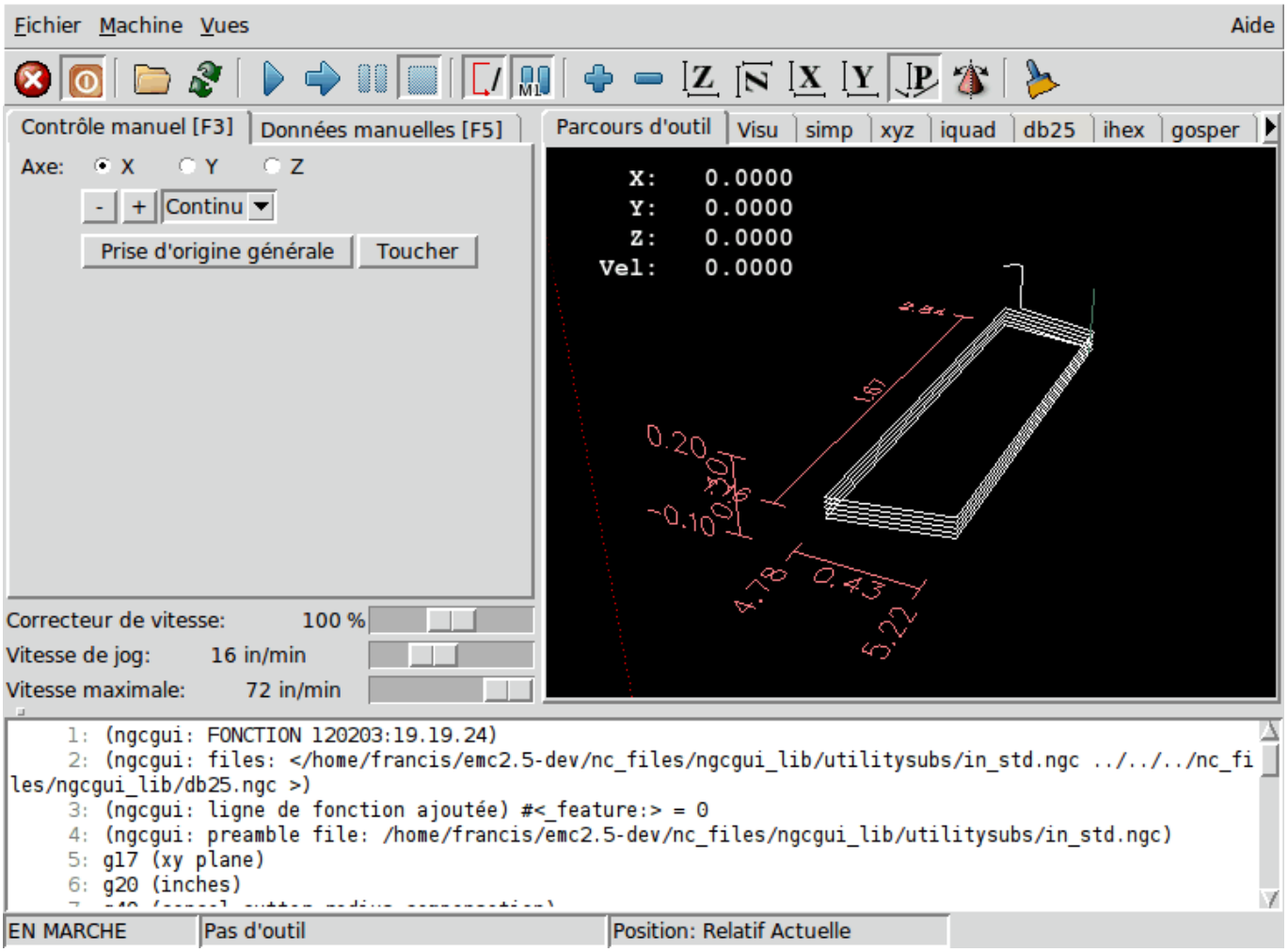
Des sous-programmes additionnels soumis par les utilisateurs se trouvent dans le forum dans la section Subroutines.

4.2.6 Exemple, découpe pour DB25

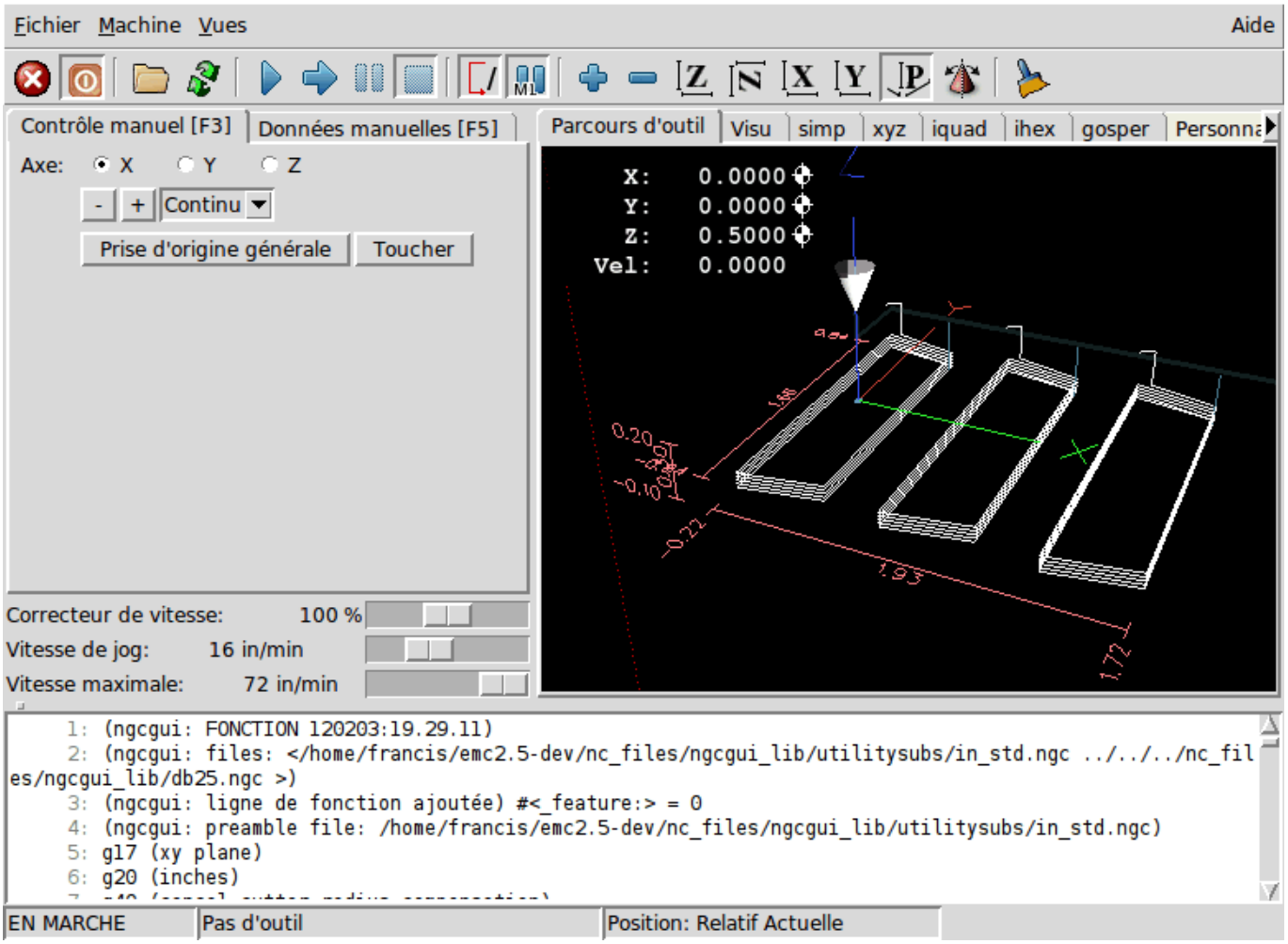
L'exemple ci-dessous montre l'utilisation du sous-programme DB25. Dans la première image on voit les champs remplis pour chaque variable.



Cette image montre le parcours d'outil du sous-programme DB25.



Cette image montre l’action du bouton Nouveau et de l’onglet personnalisé pour créer très facilement la découpe de trois DB25 en un seul programme.



4.2.7 Création d'un sous-programme

- Pour la création d'un sous-programme à utiliser avec Ngcgui, le nom de fichier et le nom du sous-programme doivent être les mêmes.
- Le fichier doit être placé dans le sous-répertoire pointé dans le fichier ini.
- À la première ligne peut se trouver un commentaires de type info: qui doit être placé au début du sous-programme.
- Le sous-programme doit être entouré par les balises `sub` et `endsub`.
- Les variables utilisées doivent être des variables numérotées et ne doivent pas sauter de numéro.
- Des commentaires et presets peuvent être inclus.

```
(info: simp -- simple exemple de sous-programme -- Ctrl-U pour éditer)
o<simp> sub
  #<ra>       = #1 (=.6 Rayon A) ;Example de paramètre avec un commentaire
  #<radius_b> = #2 (=0.4)         ;Example de paramètre sans commentaire
  #<feedrate> = #3 (Feedrate)     ;Example de paramètre sans preset
  g0x0y0z1
  g3 i#<ra> f#<feedrate>
  g3 i[0-#<radius_b>]
o<simp> endsub
```

4.3 L'interface graphique Touchy

Touchy est une interface utilisateur pour LinuxCNC, destinée à être utilisée avec les panneaux de commande de machines. Il ne nécessite ni clavier, ni souris.

Il a été conçu pour fonctionner également sur les écrans tactiles, en combinaison avec une manivelle électronique (MPG) et des boutons et interrupteurs.

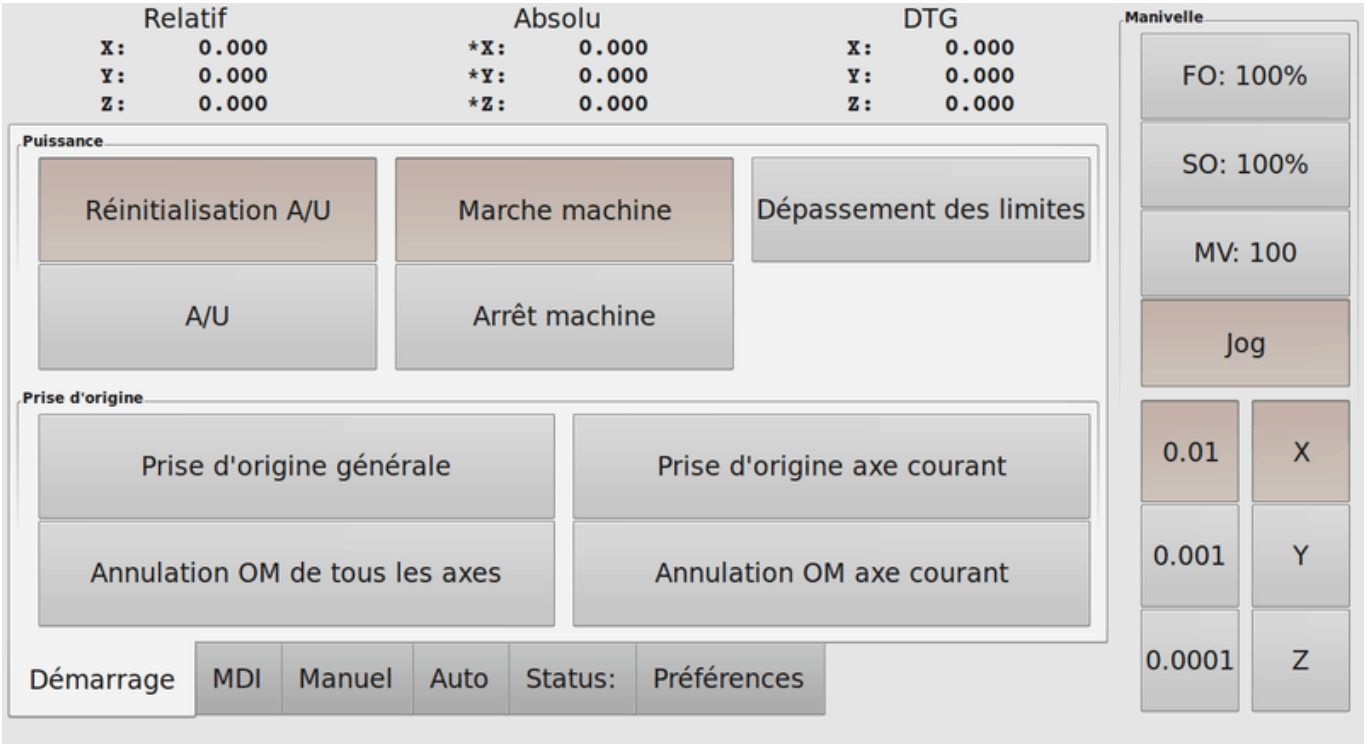


Figure 4.10: L'interface tactile Touchy

4.3.1 Panneau de Configuration

4.3.1.1 Connections avec HAL

Touchy requiert qu'un fichier nommé touchy.hal soit créé, dans le même répertoire de configuration que le fichier ini, pour y connecter ses contrôles. Touchy exécute les commandes de HAL dans ce fichier après qu'il a créé ses propres pins, lesquelles sont disponibles pour connexion.

Touchy dispose de plusieurs pins de sortie destinées à être connectées au contrôleur de mouvement pour gérer les manivelles de jog:

- touchy.jog.wheel.increment, qui doit être connecté à la pin axis.N.jog-scale de chacun des N axes.
- touchy.jog.wheel.N, qui doit être connecté à axis.N.jog-enable pour chacun des N axes.
- En plus d'être connecté à touchy.wheel-counts, le compteur d'impulsions de la manivelle doit aussi être connecté à axis.N.jog-counts pour chacun des N axes. Si le composant de HAL ilowpass est utilisé pour adoucir les mouvements de jog à la manivelle, il faut l'appliquer uniquement sur axis.N.jog-counts et non sur touchy.wheel-counts.
- Un bouton Abandon (contact momentané) connecté sur la HAL pin touchy.abort

- Un bouton de Départ cycle (contact momentané) connecté à touchy.cycle-start
- Volant/Manivelle/MPG, connecté à touchy.wheel-counts et à la pin de mouvement comme décrit précédemment.
- Un bouton à bascule simple (contact à deux positions) connecté à touchy.single-block
- Pour le jog continu, un interrupteur à trois positions avec retour au centre (ou deux boutons momentanés) pour chacun des axes concernés, attaché à touchy.jog.continuous.x.negative et à touchy.jog.continu etc. pour les autres axes.
- Si un bouton de genouillère est nécessaire, (pour jogger Z en haut de sa course en grande vitesse), un bouton à contact momentané sera connecté à touchy.quill-up.
- touchy.jog.active indique quand les contrôles de jog du panneau sont actifs.
- touchy.status-indicator est allumé en continu quand la machine exécute un G-code et clignote quand la machine est en marche mais en pause, ou en vitesse à zéro.

4.3.1.2 Recommandé pour toutes les configurations

- Un bouton d'Arrêt d'Urgence (A/U) câblé physiquement, dans la chaîne d'arrêt d'urgence.

4.3.2 Réglages

Pour utiliser Touchy, dans la section [DISPLAY] du fichier ini de la machine, modifier la ligne de cette manière: DISPLAY = touchy

Quand Touchy démarre pour la première fois, vérifier l'onglet Préférences. Dans le cas d'un écran tactile, choisir de cacher le pointeur dans les options, pour obtenir les meilleurs résultats.

La fenêtre d'état est fixée en haut, ajustée par la taille d'une police fixe. La résolution de Gnome peut affecter cela. Si le bas de l'écran est coupé, aller dans le gestionnaire de résolution de Gnome pour revenir au réglage d'origine, à 96 DPI.

4.3.2.1 Macros

Touchy peut invoquer les macros avec mots O au travers de l'interface MDI. Pour configurer cette possibilité, dans la section [TOUCHY] du fichier ini, ajouter une ou plusieurs lignes avec MACRO. Chacune de ces lignes doit respecter le format suivant:

```
MACRO=increment xinc yinc
```

Dans lequel, increment est le nom de la macro, laquelle accepte deux paramètres, nommés ici xinc et yinc.

Maintenant, placer la macro proprement dite dans un fichier nommé increment.ngc. La variable PROGRAM_PREFIX du fichier ini pourra être utilisée pour identifier le répertoire contenant ce fichier. Il est également possible de le déclarer dans la variable SUBROUTINE_PATH.

Elle pourrait ressembler à cela:

```
O<increment> sub
G91 G0 X#1 Y#2
G90
O<increment> endsub
```

Noter que le nom du sous-programme, le nom de la macro ainsi que le nom du fichier .ngc doivent correspondre exactement, y compris les minuscules/ majuscules des noms.

Quand la macro est invoquée en pressant le bouton Macro dans l'onglet MDI de Touchy, il est possible de saisir des valeurs pour xinc et yinc, lesquelles seront passées à la macro comme étant respectivement **#1** et **#2**. Les paramètres laissées vides sont passés comme des valeur **0**.

Si il y a plusieurs macros différentes, presser le bouton Macro répétitivement pour les faire défiler.

Dans notre petit exemple, si -1 est entré pour xinc puis que le Départ cycle est pressé, un **G0**, sera invoqué, provoquant un déplacement en vitesse rapide, vers la gauche, de une unité machine.

Cette capacité d'utilisation des macros est très utile pour le palpage de contours ou d'orifices ainsi que pour d'autres opérations simples pré-configurées, de fraisage ou de perçage, qui pourront être réalisées depuis le panneau de Touchy sans avoir, pour cela, à écrire de programme G-code.

4.4 L'interface graphique TkLinuxCNC

4.4.1 Introduction

TkLinuxCNC est l'interface utilisateur graphique la plus populaire après Axis, c'est l'interface traditionnelle de LinuxCNC. Elle est écrite en Tcl et utilise le toolkit Tk pour l'affichage. Le fait d'être écrite en TCL la rend vraiment très portable (elle fonctionne sur une multitude de plateformes).

Figure 4.11: L'affichage de TkLinuxCNC

4.4.2 Utiliser TkLinuxCNC

Pour sélectionner l'interface graphique TkLinuxCNC avec LinuxCNC, éditer le fichier .ini et dans la section [DISPLAY] modifier l'affichage comme ci-dessous:

```
DISPLAY = tklinuxcnc
```

Puis, lancer LinuxCNC et choisir ce fichier ini. La configuration qui se trouve dans sim/tklinuxcnc/tklinuxcnc.ini est déjà configurée pour utiliser TkLinuxCNC comme interface utilisateur.

Quand LinuxCNC est lancé avec TkLinuxCNC, une fenêtre comme celle-ci s'affiche.

4.4.2.1 Une session typique avec TkLinuxCNC

1. Lancer LinuxCNC et sélectionner un fichier de configuration.
2. Libérer l'Arrêt d'Urgence et mettre la machine en marche (en pressant F1 puis F2).
3. Faire l'Origine Machine de chacun des axes.
4. Charger un fichier d'usinage.

5. Brider le brut à usiner sur la table.
6. Faire l'Origine Pièce de chacun des axes, à l'aide du jog ou en introduisant une valeur de décalage d'origine après un clic droit sur le nom d'un axe.
7. Lancer le programme.
8. Pour refaire une autre pièce identique, reprendre à l'étape 6. Pour usiner une pièce différente, reprendre à l'étape 4. Quand c'est terminé, quitter LinuxCNC.

4.4.3 Éléments affichés par TkLinuxCNC

La fenêtre TkLinuxCNC contient les éléments suivants:

- Une barre de menu permettant diverses actions;
- Un jeu de boutons permettant d'agir sur le mode de travail, Marche/Arrêt de la broche et autres éléments;
- Une barre de statut pour l'affichage des différents offsets;
- Une zone d'affichage des coordonnées;
- Un jeu de curseurs pour contrôler la vitesse de jog, le Correcteur de vitesse d'avance et le Correcteur de vitesse broche qui permettent d'augmenter ou de diminuer ces vitesses ;
- Une boîte d'entrée de données manuelles;
- Une barre de statut affichant le bloc de programme actif, G-codes, M-codes, mots F et S;
- Les boutons relatifs à l'interpréteur;
- Une zone d'affichage de texte montrant le G-code du programme chargé.

4.4.3.1 Boutons principaux

Dans la première ligne de la gauche vers la droite et cycliquement:

1. Marche Machine: Arrêt d'Urgence Arrêt d'Urgence relâché / Marche
2. Bascule gouttelettes
3. Broche moins vite
4. Direction de rotation de la broche Arrêt broche / Broche sens horaire / Broche sens anti-horaire
5. Broche plus vite
6. Annuler

puis dans la deuxième ligne:

1. Mode de marche: MANUEL / MDI / AUTO
2. Bascule d'arrosage
3. Bascule du contrôle frein de broche

4.4.3.2 Barre de statut des différents offsets

Elle affiche, l'offset de rayon de l'outil courant (sélectionné avec Txx M6), l'offset éventuel de longueur d'outil si il est actif et les offsets de travail (ajustables par un clic droit sur les coordonnées).

4.4.3.3 Zone d'affichage des coordonnées

La partie principale affiche la position courante de l'outil. La couleur varie selon l'état de l'axe. Si l'axe n'est pas référencé il est affiché en caractères jaunes. Si il est référencé il s'affiche en vert. Si il est en erreur, TkLinuxCNC l'affiche en rouge pour montrer un défaut. (par exemple si un contact de fin de course est activé).

Pour interpréter correctement les différentes valeurs, se référer aux boutons de droite. Si la position est Machine, alors la valeur affichée est en coordonnées machine. Si elle est Relative, la valeur affichée est en coordonnées pièce. Deux autres en dessous indiquent actuelle ou commandée. Actuelle fait référence aux valeurs retournées par les codeurs (si la machine est équipée de servomoteurs) et commandée fait référence à la position à atteindre envoyée aux moteurs. Ces valeurs peuvent différer pour certaines raisons: Erreur de suivi, bande morte, résolution d'encodeur ou taille de pas. Par exemple, si un mouvement est commandé vers X0.08 sur une fraiseuse, mais qu'un pas moteur fait 0.03, alors la position Commandée sera 0.03 mais la position Actuelle sera soit 0.06 (2 pas) soit 0.09 (3 pas).

Deux autres boutons permettent de choisir entre la vue Articulation et la vue Globale. Cela a peu de sens avec les machines de type normal (cinématiques triviales), mais se révèle très utile sur les machines avec des cinématiques non triviales telles que les robots ou plateforme de Stewart. (Des informations plus complètes se trouvent dans le manuel de l'intégrateur).

Quand la machine se déplace, elle laisse un tracé appelé parcours d'outil. La fenêtre d'affichage du parcours d'outil s'active via le menu Vues → Parcours d'outil.

4.4.3.4 Contrôle en automatique

Figure 4.12: Interpréteur de TkLinuxCNC

Les boutons de contrôle de la partie inférieure de TkLinuxCNC, visibles sur l'image ci-dessus, sont utilisés pour l'exécution du programme:

- Ouvrir pour charger un fichier,
- Lancer pour commencer l'usinage,
- Pause pour stopper temporairement l'usinage,
- Reprise pour reprendre un programme mis en pause,
- Pas à pas pour avancer d'une seule ligne de programme,

- Vérifier pour vérifier si il contient des erreurs,
- Arrêt optionnel pour basculer l'arrêt optionnel, si ce bouton est vert l'exécution du programme est stoppée quand un code M1 est rencontré.

Quand un programme est lancé, la ligne courante est affichée en surbrillance blanche. L'affichage du texte défile automatiquement pour montrer la ligne courante.

4.4.3.5 Contrôle en manuel

TkLinuxCNC permet les déplacements manuels de la machine. Cette action s'appelle le jog. Premièrement, sélectionner l'axe à déplacer en cliquant dessus. Puis, cliquer et maintenir les boutons **+** ou **-** selon la direction du mouvement souhaité. Les quatre premiers axes peuvent aussi être déplacés à l'aide des touches fléchées pour les axes X et Y, Pg.préc et Pg.suiv pour l'axe Z et les touches [et] pour l'axe A.

Si Continu est activé, le mouvement sera continu tant que la touche sera pressée, si une valeur d'incrément est sélectionnée, le mobile se déplacera exactement de cette valeur à chaque appui sur la touche ou à chaque clic. Les valeurs disponibles sont:

`1.0000 0.1000 0.0100 0.0010 0.0001`

En cliquant le bouton Origine ou en pressant la touche Origine, l'axe actif est référencé sur son origine machine. Selon la configuration, la valeur de l'axe peut être simplement mise à la position absolue 0.0, ou la machine peut se déplacer vers un point spécifique matérialisé par le contact d'origine. Voir le manuel de l'intégrateur pour plus de détails sur les prises d'origine.

En cliquant le bouton Dépassement de limite, la machine permet un jog temporaire pour même si l'axe à franchi une limite d'axe fixée dans le fichier .ini. Noter que si Dépassement de limite est activé il s'affiche en rouge.

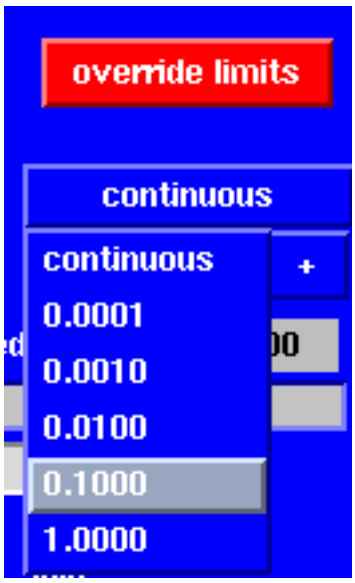


Figure 4.13: Exemple de dépassement de limite et incréments de jog

Le bouton central du dessus sélectionne le sens de rotation de la broche: Anti-horaire, Arrêt, Horaire. Les boutons fléchés augmentent ou diminuent la vitesse de rotation. Le bouton central du dessous permet d'engager ou de relâcher le frein de broche. Selon la configuration de la machine, les items de ce groupe ne sont peut être pas tous visibles.

Ces deux boutons permettent d'activer ou non les lubrifiants Gouttelettes et Arrosage. Selon la configuration de la machine, les items de ce groupe ne sont peut être pas tous visibles.

4.4.3.6 Entrée manuelle de G-code (MDI)

L'entrée manuelle de données (aussi appelée MDI), permet d'entrer et d'exécuter des lignes de G-code, une à la fois. Quand la machine n'est pas en marche ni mise en mode MDI, l'entrée de code n'est pas possible.

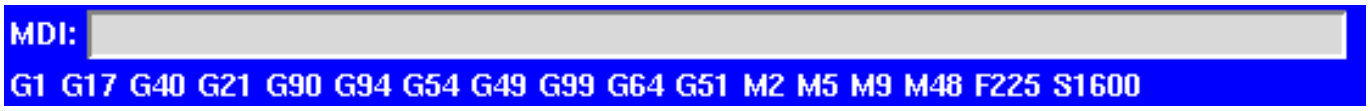


Figure 4.14: Le champ de saisie des entrées manuelles

docs: image alt-tags added Signed-off-by: Thoren Seufl <t_seufl@gmx.de>==== MDI:

Le mode MDI permet d'exécuter une commande en G-code en pressant la touche Entrée.

Ce champs montre les codes modaux actuellement actifs dans l'interpréteur. Par exemple, **G54** indique que le système de coordonnées courant est celui de G54 et qu'il s'applique à toutes les coordonnées entrées.

4.4.3.7 Vitesse de Jog

En déplaçant ce curseur, la vitesse de jog peut être modifiée. Le nombre indique une vitesse en unités par minute. Le champs de texte est cliquable. Un clic ouvre un dialogue permettant d'entrer un nombre.

4.4.3.8 Correcteur de vitesse d'avance travail

En déplaçant ce curseur, la vitesse d'avance travail peut être modifiée. Par exemple, si la vitesse d'avance travail du programme est **F600** et que le curseur est placé sur 120%, alors la vitesse d'avance travail sera de 720. Le champs de texte est cliquable. Un clic ouvre un dialogue permettant d'entrer un nombre.

4.4.3.9 Correcteur de vitesse de broche

Le fonctionnement de ce curseur est le même que celui de la vitesse d'avance, mais il contrôle la vitesse de rotation de la broche. Si le programme demande S500 (broche à 500 tr/mn) et que le curseur est placé sur 80%, alors la vitesse de broche résultante sera de 400 tr/mn. Le minimum et le maximum pour ce curseur sont définis dans le fichier ini. Par défaut le curseur est placé sur 100%. Le champs de texte est cliquable. Un clic ouvre un dialogue permettant d'entrer un nombre.

4.4.4 Raccourcis clavier

La plupart des actions de TkLinuxCNC peuvent être accomplies au clavier. Beaucoup des raccourcis clavier ne sont pas accessibles en mode MDI.

Les raccourcis clavier les plus fréquemment utilisés sont montrés dans la table ci-dessous.

Table 4.2: Les raccourcis clavier les plus utilisés

Touche	**Action**
F1	Bascule de l'Arrêt d'Urgence
F2	Marche/Arrêt machine
*, 1 .. 9, 0	Correcteur vitesse d'avance 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
Origine	POM de l'axe actif
Gauche, Droite	Jog du premier axe
Haut, Bas	Jog du deuxième axe
Pg.prec, Pg.suiv	Jog du troisième axe
[,]	Jog du quatrième axe
Echap	Arrête l'exécution

Chapter 5

Programming

5.1 Systèmes de coordonnées et décalages

5.1.1 Introduction

Dans ce chapitre, nous allons tenter de démystifier les systèmes de coordonnées. C'est une notion capitale pour maitriser le fonctionnement d'une machine CNC, sa configuration et son utilisation.

Nous montrerons également, qu'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 donc introduire les décalages et comment ils sont utilisés par LinuxCNC:

- Les coordonnées machine.(G53)
- Les neuf décalages d'origine pièce.(G54 à G59.3)
- Un jeu de décalages globaux (G92) and local offsets (G52)

5.1.2 Commande en coordonnées machine: G53

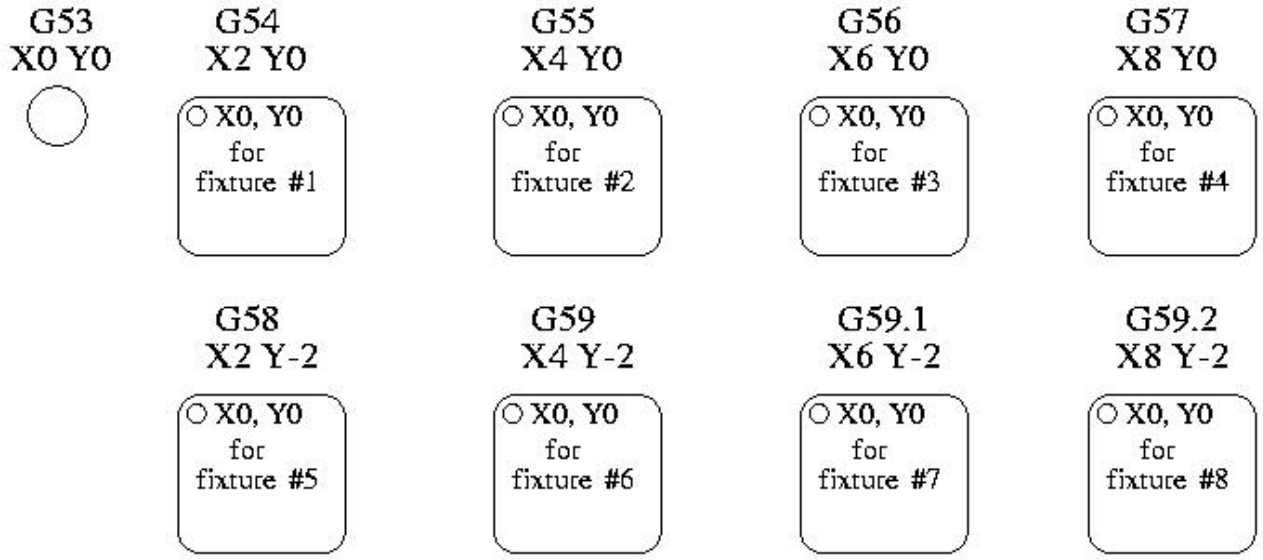
Indépendamment de tout décalage pouvant être actif, un G53 dans une ligne de code indique à l'interpréteur de se déplacer aux positions réelles des axes (positions absolues), commandées dans la ligne. Par exemple:

```
G53 G0 X0 Y0 Z0
```

déplacera le mobile depuis la position actuelle vers la position où les coordonnées machine 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.

5.1.3 Décalages pièce (G54 à G59.3)



Exemple de décalages pour 8 ilots 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 décalé du système de coordonnées machine (absolu). Il permet, entre autre, dans le cas de pièces multiples mais semblables, de créer en décalant ses origines, le système de coordonnées de chaque pièce, le programme restant le même. Un cas typique d'utilisation de cette fonctionnalité, pour usiner huit ilots identiques sur la même pièce, est illustré sur la figure ci-dessus.

Les valeurs des décalages sont enregistrées dans le fichier VAR qui est requis par le fichier INI durant le démarrage de LinuxCNC. Dans l'exemple ci-dessous, qui utilise G55, la valeur de chacun des axes de G55 est enregistrée dans une variable numérotée.

Variable	Valeur
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 que LinuxCNC lise les nouvelles valeurs. Pour notre exemple, nous allons éditer directement le fichier pour que G55 prenne les valeurs suivantes:

Variable	Valeur
5241	2.000000
5242	1.000000
5243	-2.000000
5244	0.000000
5245	0.000000

Variable	Valeur
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 figure des décalages d'ilots, 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 précédemment.

```
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(((G54)))
G55     utilise les réglages du système de coordonnées 2(((G55)))
G56     utilise les réglages du système de coordonnées 3(((G56)))
G57     utilise les réglages du système de coordonnées 4(((G57)))
G58     utilise les réglages du système de coordonnées 5(((G58)))
G59     utilise les réglages du système de coordonnées 6(((G59)))
G59.1   utilise les réglages du système de coordonnées 7(((G59.1)))
G59.2   utilise les réglages du système de coordonnées 8(((G59.2)))
G59.3   utilise les réglages du système de coordonnées 9(((G59.3)))
```

5.1.3.1 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 LinuxCNC 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, ou l'absence de la variable 5220, provoquera au démarrage le retour de LinuxCNC à la valeur par défaut de 1.00000.

5.1.3.2 Réglage des décalages avec G10

La commande G10 L2x peut être utilisée pour modifier les valeurs des décalages d'un système de coordonnées pièce: (Nous donnons seulement ici un bref aperçu, se reporter aux sections du G-code pour une description complète).

- G10 L2 P(pièce 1-9) - Ajuste les valeurs d'offset. La position courante reste inchangée. (voir la sectionG10 L2 pour les détails)
- G10 L20 P(pièce 1-9) - Ajuste les valeurs d'offset de sorte que la position courante devienne la position donnée en paramètre. (Voir la section G10 L20 pour les détails)

5.1.4 Local and Global Offsets

5.1.4.1 The G52 command

G52 is used in a part program as a temporary "local coordinate system offset" within the workpiece coordinate system. An example use case is when machining several identical features at different locations on a part. For each feature, G52 programs a local reference point within workpiece coordinates, and a subprogram is called to machine the feature relative to that point.

G52 axis offsets are programmed relative to workpiece coordinate offsets G54 through G59.3. As a local offset, G52 is applied after the workpiece offset, including rotation. Thus, a part feature will be machined identically on each part regardless of the part's orientation on the pallet.

Caution
As a temporary offset, set and unset within the localized scope of a part program, in other G-code interpreters G52 does not persist after machine reset, M02 or M30. In LinuxCNC, G52 shares parameters with G92, which, for historical reasons, **does** persist these parameters. See G92 Persistence Cautions below.

```
'G52' and 'G92' share the same offset registers.  Therefore, setting
'G52' will override any earlier 'G92' setting, and 'G52' will persist
across machine reset when 'G92' persistence is enabled.  These
interactions may result in unexpected offsets.
See <<sec:g92-g52-interaction-cautions,G92 and G52 Interaction Cautions>> below.
```

Programming G52 X1 Y2 offsets the current workpiece coordinate system X axis by 1 and Y axis by 2. Accordingly, on the DRO, the current tool position's X and Y coordinates will be reduced by 1 and 2, respectively. Axes unset in the command, such as Z in the previous example, will be unaffected: any previous G52 Z offset will remain in effect, and otherwise the Z offset will be zero.

The temporary local offset may be canceled with G52 X0 Y0. Any axes not explicitly zeroed will retain the previous offset.

G52 shares the same offset registers as G92, and thus G52 is visible on the DRO and preview labeled with G92.

5.1.5 Décalages d'axes G92

G92 est la plus incomprise et la plus maligne des commandes programmables avec LinuxCNC. 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.

5.1.5.1 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.

L'utilisateur doit bien comprendre le fonctionnement des valeurs de G92. 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 tests 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.

Lors du démarrage de LinuxCNC, si des offsets existent dans les variables de G92, ils seront appliqués lors de la prise d'origine des axes concernés. Il est donc de bonne pratique de mettre les offsets de G92 à zéro par G92.1 ou un G92.2 à la fin de leur utilisation.

5.1.5.2 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 tklinuxcnc, 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 U V W 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=2.000, un G92 X0 fixera un décalage de -2.0000, de sorte que l'emplacement actuel de X devienne X=0.000. Un nouveau G92 X5.000 fixera un décalage de 3.000 et l'affichage indiquera une position courante X=5.000.

5.1.5.3 Précautions 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 actives. Les variables sont numérotées:

Variable	Valeur
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.

Si des valeurs G92 existent dans le fichier VAR quand LinuxCNC démarre, ces valeurs seront appliquées aux valeurs courantes des emplacements d'axe. Si c'est sa position d'origine et que l'origine est définie au zéro machine, tout sera correct. Une fois que l'origine machine a été établie en utilisant les contacts d'origine machine, ou en déplaçant chaque axe à une position connue, puis en envoyant la commande de prise d'origine de l'axe, tous les décalages G92 seront appliqués. Si un X1 G92 est actif lors de la prise d'origine machine de l'axe X, la visu affichera X: 1.000 au lieu du X: 0.000 attendu, c'est parce-que le G92 a été appliqué à l'origine machine. Si vous passez un G92.1 et que la visu affiche tous à zéro, alors c'est que vous avez encore l'effet de l'offset G92 de la dernière session de LinuxCNC.

Sauf si votre intention est d'utiliser les mêmes décalages G92 dans le prochain programme, la meilleure pratique consiste à envoyer un G92.1 à la fin de tout fichier de G-code dans lequel vous utilisez les compensations G92.

5.1.6 Exemple de programme utilisant les décalages d'axes

Cet exemple de projet de gravure, usine 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-0.1 Y0 Z0
G1 F1 Z-0.25
G3 X-0.1 Y0 I0.1 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 (décalages G55 X la valeur de 0.5 pouces)
G10 L2 P3 X-0.5 (décalages G56 X valeur de -0.5 pouces)
G10 L2 P4 Y0.5 (décalages G57 valeur Y de 0.5 pouces)
G10 L2 P5 Y-0.5 (décalages G58 valeur Y de -0.5 pouces)
```

Nous mettons ces ensembles 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 (décalages G55 X la valeur de 0.5 pouces)
G10 L2 P3 X-0.5 (décalages G56 X la valeur de -0.5 pouces)
G10 L2 P4 Y0.5 (décalages G57 X la valeur de 0.5 pouces)
G10 L2 P5 Y-0.5 (décalages G58 X la valeur de -0.5 pouces)

G54 G0 X-0.1 Y0 Z0 (cercle du centre)
G1 F1 Z-0.25
G3 X-0.1 Y0 I0.1 J0
G0 Z0

G55 G0 X-0.1 Y0 Z0 (premier cercle compensé)
G1 F1 Z-0.25
```

```
G3 X-0.1 Y0 I0.1 J0
G0 Z0

G56 G0 X-0.1 Y0 Z0 (deuxième cercle compensé)
G1 F1 Z-0.25
G3 X-0.1 Y0 I0.1 J0
G0 Z0

G57 G0 X-0.1 Y0 Z0 (troisième cercle compensé)
G1 F1 Z-0.25
G3 X-0.1 Y0 I0.1 J0
G0 Z0

G58 G0 X-0.1 Y0 Z0 (quatrième cercle compensé)
G1 F1 Z-0.25
G3 X-0.1 Y0 I0.1 J0
G54 G0 X0 Y0 Z0

M2
```

Maintenant c'est le moment d'appliquer une série de décalages G92 à ce programme. Vous verrez que c'est fait dans chaque cas de Z0. Si la machine était à la position zéro, un G92 Z1.0000 placé en tête de programme le décalerait 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, les 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 LinuxCNC et donc, d'éviter de les recharger quand vous démarrez à nouveau le programme.

5.2 Vue générale du langage G-codes de LinuxCNC

5.2.1 Brève description du G-code de LinuxCNC

Le G-code est le langage de programmation des machines numériques. Le G-code utilisé par LinuxCNC est basé sur le langage RS274/NGC. Cette documentation le décrit de manière exhaustive, c'est donc un gros morceau mais il contient beaucoup de concepts qui seront assimilés par le lecteur dès la première lecture. C'est notamment le cas de ce chapitre. Par la suite, l'utilisateur reviendra ici, d'abord pour chaque détail de création de son G-code, puis plus tard, seulement pour vérifier la syntaxe des codes les moins courants. Il aura alors perçu la puissance de ce langage et de LinuxCNC qui le met à profit.

5.2.2 Format des paramètres du G-code

Le langage G-code est basé sur des lignes de code. Chaque ligne (également appelée un bloc) peut inclure des commandes pour faire produire diverses actions à la machine. Plusieurs lignes de code peuvent être regroupées dans un fichier pour créer un programme G-code.

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 G-code commencent avec une lettre G ou M (G pour Général et M pour Miscellaneous (auxiliaire)). Les termes pour ces commandes sont G-codes et M-codes.

Le langage G-code 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 pour-cent 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 pour-cent au début, mais pas à la fin. Le contenu utile d'un fichier délimité par pour-cent s'arrête après la seconde ligne pour-cent. Tout le reste est ignoré.

Le langage G-code 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 pas.

5.2.3 Format d'une ligne

Une ligne de G-code typique est construite de la façon suivante, dans l'ordre avec la restriction à un maximum de 256 caractères sur la même ligne.

1. Un caractère optionnel d'effacement de bloc, qui est la barre oblique /.
2. Un numéro de ligne optionnel.
3. Un nombre quelconque 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.

5.2.4 Caractère d'effacement de bloc

Le caractère optionnel d'effacement de bloc qui est la barre oblique /, quand il est placé en premier sur une ligne, peut être utilisé par certaines interfaces utilisateur pour sauter, si besoin, des lignes de code. Dans Axis, la combinaison de touches Alt-m-/ est une bascule qui active ou désactive l'effacement de bloc. Quand l'effacement de bloc est actif, toutes les lignes commençant par / sont sautées.

Dans Axis il est également possible de basculer l'activation d'effacement de bloc avec l'icône: 

5.2.5 Numéro de ligne

Un numéro de ligne commence par la lettre N suivie d'un nombre entier non signé. Les numéros de ligne peuvent se suivre, être répétés 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, ni même recommandée, mais si ils sont utilisés, il doivent être placés en début de ligne.

5.2.6 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 ci-dessous. Ce 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.

Table 5.1: 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 des codes modaux)
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 des codes modaux)
N	Numéro de ligne
P	Temporisation utilisée dans les cycles de perçage et avec G4.
	Mot clé utilisé avec G10.
Q	Incrément Delta en Z dans un cycle G73, 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

5.2.7 Les 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:

  - un signe plus ou un signe moins optionnel, suivi de
  - zéro à plusieurs chiffres, peut être suivis par,
  - un point décimal, suivi de
  - zéro à plusieurs chiffres, il doit au moins y avoir un chiffre.

- Il existe deux types de nombres:
  - Les entiers, qui n'ont pas de point décimal.
  - Les décimaux, qui ont un point décimal.
- 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 G-code est proche d'une valeur entière à moins de quatre décimales, il est considéré comme entier, par exemple 0.9999.

5.2.8 Paramètres (Variables)

Le langage RS274/NGC supporte les paramètres, qui sont appelés variables dans d'autres langages de programmation. Il existe plusieurs types de paramètres ayant différents usages et différentes formes. Le seul type de nombre supporté par les paramètres est le flottant, il n'y a pas de string, pas de boolean ni d'entier dans le G-code comme dans d'autres langages de programmation. Toutefois, les expressions logiques peuvent être formulées avec les opérateurs booléens (AND, OR, XOR et les opérateurs de comparaison EQ, NE, GT, GE, LT, LE) ainsi que MOD, ROUND, FUP et FIX les fonctions qui supportent l'arithmétique entière.

Les paramètres différent par leur syntaxe, leur portée, leur comportement quand ils ne sont pas encore initialisés, leur mode, leur persistance et l'usage pour lequel ils sont prévus.

Syntaxes
Il y a trois sortes d'apparences syntaxiques:

- numéroté - #4711
- nommé local - #<valeurlocale>
- nommé global - #<_valeurglobale>

La portée
La portée d'un paramètre est soit globale, ou locale à l'intérieur d'un sous-programme. Les paramètres de sous-programme et les paramètres nommés ont une portée locale. Les paramètres nommés globaux et les paramètres numérotés commencent par un nombre, exemple: 31 a une portée globale. RS274/NGC utilise une portée lexicale, dans un sous-programme, seules sont locales les variables qui y sont définies et toutes les variables globales y sont visibles. Les variables locales à un appel de procédure, ne sont pas visibles dans la procédure appelée.

Le comportement des paramètres non encore initialisés

1. Les paramètres globaux non initialisés et les paramètres de sous-programmes inutilisés, retournent la valeur zéro quand ils sont utilisés dans une expression.
2. Les paramètres nommés signalent une erreur quand ils sont utilisés dans une expression.

Le mode
La plupart des paramètres sont en lecture/écriture et peuvent être assignés dans une instruction d'affectation. Cependant, pour beaucoup de paramètres prédéfinis, cela n'a pas de sens, ils sont alors en lecture seule. Ils peuvent apparaître dans les expressions, mais pas sur le côté gauche d'une instruction d'affectation.

La persistance
Quand LinuxCNC s'arrête, les paramètres volatiles perdent leurs valeurs. Tous les paramètres sont volatiles, excepté les paramètres numérotés dans l'étendue courante de persistance [1]. Les paramètres persistants sont enregistrés dans un fichier .var et restaurés à leurs valeurs précédentes quand LinuxCNC est relancé. Les paramètres numérotés volatiles sont remis à zéro.

Utilisation prévue

1. Paramètres utilisateur:: paramètres numérotés dans l'étendue 31 à 5000, paramètres nommés globaux et locaux excepté les paramètres prédéfinis. Sont disponibles pour une utilisation générale de stockage de valeurs flottantes, comme des résultats intermédiaires, des drapeaux, etc. durant l'exécution d'un programme. Ils sont en lecture/écriture (une valeur peut leur être attribuée).
2. Paramètres de sous-programme - Ils sont utilisés pour conserver les paramètres actuels passés à un sous-programme.
3. paramètres numérotés - la plupart de ces paramètres sont utilisés pour accéder aux offsets des systèmes de coordonnées.
4. paramètres nommés prédéfinis - utilisés pour déterminer l'état de l'interpréteur et de la machine, par exemple #<_relative> retourne 1 si G91 est actif et 0 si G90 est activé. Ils sont en lecture seule.

5.2.9 Paramètres numérotés

Un paramètre numéroté commence par le caractère # suivi par un entier compris entre 1 et (actuellement) 5602. 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 (la valeur 15 est stockée dans le paramètre numéro 3)
```

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.

31 à 5000
Paramètres des G-Code utilisateur. Ces paramètres sont globaux dans le fichier G-code.

5061 à 5069
Résultat du palpage G38.2 pour X Y Z A B C U V W. Volatile.

5161 à 5169
Coordonnées d'un G28 pour X Y Z A B C U V W. Persistant.

5181 à 5189
Origine G30 pour X Y Z A B C U V W. Persistant.

[1] L'étendue de persistance courante des paramètres évolue en même temps qu'évolue le développement. Cette étendue est actuellement de 5161 à 5390. Elle est définie par _required_parameters array dans le fichier src/linuxcnc/rs274ngc/interp_array.cc .

5211 à 5219
Offset G52 et G92 pour X Y Z A B C U V W. Persistant.

5220
Système de coordonnées 1 à 9 pour G54 à G59.3. Persistant.

5221 à 5229
Système de coordonnées 1, G54 pour X Y Z A B C U V W R. Persistant.

5241 à 5249
Système de coordonnées 2, G55 pour X Y Z A B C U V W R. Persistant.

5261 à 5269
Système de coordonnées 3, G56 pour X Y Z A B C U V W R. Persistant.

5281 à 5289
Système de coordonnées 4, G57 pour X Y Z A B C U V W R. Persistant.

5301 à 5309
Système de coordonnées 5, G58 pour X Y Z A B C U V W R. Persistant.

5321 à 5329
Système de coordonnées 6, G59 pour X Y Z A B C U V W R. Persistant.

5341 à 5349
Système de coordonnées 7, G59.1 pour X Y Z A B C U V W R. Persistant.

5361 à 5369
Système de coordonnées 8, G59.2 pour X Y Z A B C U V W R. Persistant.

5381 à 5389
Système de coordonnées 9, G59.3 pour X Y Z A B C U V W R. Persistant.

5399
Résultat de M66 - Surveille ou attends une entrée. Volatile.

5400
Numéro de l'outil courant. Volatile.

5401 à 5409
Offset d'outil pour X Y Z A B C U V W. Volatile.

5410
Diamètre de l'outil courant. Volatile.

5411
Angle frontal de l'outil courant. Volatile.

5412
Angle arrière de l'outil courant. Volatile.

5413
Orientation de l'outil. Volatile.

5420 à 5428
Positions courantes incluant les offsets, dans l'unité courante du programme pour X Y Z A B C U V W.

5.2.10 Paramètres de sous-programme

- 1-30 - Paramètres d'appel d'arguments, locaux au sous-programme. Voir la section des O-codes.

5.2.11 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>]
```

5.2.12 Paramètres nommés prédéfinis

Les paramètres globaux suivants sont disponibles en lecture seule, pour accéder aux états internes de l'interpréteur et de la machine. Ils peuvent être utilisés dans les expressions quelconques, par exemple pour contrôler le flux d'un programme avec les instructions if-then-else.

- #<_vmajor> - Version majeure de LinuxCNC. Si la version courante est 2.5.2, 2.5 est retourné.
- #<_vminor> - Version mineure du LinuxCNC. Si la version courante est 2.6.2, 0.2 est retourné.
- #<_line> - Numéro de séquence. Si un fichier G-code est en cours, le numéro de la ligne courante est retourné.
- #<_motion_mode> - Retourne le mode mouvement courant de l'interpréteur:

Mode mouvement	**Valeur retournée**
G1	10
G2	20
G3	30

Mode mouvement	Valeur retournée
G33	330
G38.2	382
G38.3	383
G38.4	384
G38.5	385
G5.2	52
G73	730
G76	760
G80	800
G81	810
G82	820
G83	830
G84	840
G85	850
G86	860
G87	870
G88	880
G89	890

- #<_plane> - Retourne une valeur désignant le plan courant:

Plan	Valeur retournée
G17	170
G18	180
G19	190
G17.1	171
G18.1	181
G19.1	191

- #<_ccomp> - Statut de la compensation d'outil. Retourne une valeur:

Mode	Valeur retournée
G40	400
G41	410
G41.1	411
G41	410
G42	420
G42.1	421

- #<_metric> - Retourne 1 si G21 est on, sinon 0.
- #<_imperial> - Retourne 1 si G20 est on, sinon 0.
- #<_absolute> - Retourne 1 si G90 est on, sinon 0.
- #<_incremental> - Retourne 1 si G91 est on, sinon 0.

- #<_inverse_time> - Retourne 1 si le mode inverse du temps (G93) est on, sinon 0.
- #<_units_per_minute> - Retourne 1 si le mode unités par minute (G94) est on, sinon 0.
- #<_units_per_rev> - Retourne 1 si le mode Unités par tour (G95) est on, sinon 0.
- #<_coord_system> - Retourne l'index du système de coordonnées courant (G54 à G59.3).

Mode	Valeur re-tournée
G54	0
G55	1
G56	2
G57	3
G58	4
G59	5
G59.1	6
G59.2	7
G59.3	8

- #<_tool_offset> - Retourne 1 si l'offset d'outil (G43) est on, sinon 0.
- #<_retract_r_plane> - Retourne 1 si G98 est actif, sinon 0.
- #<_retract_old_z> - Retourne 1 si G99 est on, sinon 0.

5.2.13 Paramètres système

- #<_spindle_rpm_mode> - Retourne 1 si la broche est en mode tr/mn (G97), sinon 0.
- #<_spindle_css_mode> - Retourne 1 si la broche est en mode vitesse de coupe constante (G96), sinon 0.
- #<_ijk_absolute_mode> - Retourne 1 si le mode de déplacement en arc est absolu (G90.1), sinon 0.
- #<_lathe_diameter_mode> - Retourne 1 pour un tour configuré en mode diamètre (G7), sinon 0.
- #<_lathe_radius_mode> - Retourne 1 pour un tour configuré en mode rayon (G8) , sinon 0.
- #<_spindle_on> - Retourne 1 si la broche tourne (M3 ou M4 en cours), sinon 0.
- #<_spindle_cw> - Retourne 1 si la broche est dans le sens horaire (M3) sinon 0.
- #<_mist> - Retourne 1 si l'arrosage par gouttelettes est activé (M7).
- #<_flood> - Retourne 1 si l'arrosage fluide est activé (M8).
- #<_speed_override> - Retourne 1 si un correcteur de vitesse d'avance travail est activé (M48 ou M50 P1), sinon 0.
- #<_feed_override> - Retourne 1 si un correcteur de vitesse broche est activé (M48 ou M51 P1), sinon 0.
- #<_adaptive_feed> - Retourne 1 si un correcteur de vitesse adaptative est activé (M52 ou M52 P1), sinon 0.
- #<_feed_hold> - Retourne 1 si le contrôle de coupure vitesse est activé (M53 P1), sinon 0.
- #<_feed> - Retourne la valeur courante d'avance travail (F).

- #<_rpm> - Retourne la valeur courante de vitesse broche (S).
- #<_x> - Retourne la coordonnée machine courante en X. Identique à #5420.
- #<_y> - Retourne la coordonnée machine courante en Y. Identique à #5421.
- #<_z> - Retourne la coordonnée machine courante en Z. Identique à #5422.
- #<_a> - Retourne la coordonnée machine courante en A. Identique à #5423.
- #<_b> - Retourne la coordonnée machine courante en B. Identique à #5424.
- #<_c> - Retourne la coordonnée machine courante en C. Identique à #5425.
- #<_u> - Retourne la coordonnée machine courante en U. Identique à #5426.
- #<_v> - Retourne la coordonnée machine courante en V. Identique à #5427.
- #<_w> -Retourne la coordonnée machine courante en W. Identique à #5428.
- #<_current_tool> - Retourne le N° de l'outil courant monté dans la broche. Identique à #5400.
- #<_current_pocket> - Retourne le N° de poche de l'outil courant.
- #<_selected_tool> - Retourne le N° de l'outil sélectionné par le mot T. Par défaut -1.
- #<_selected_pocket> - Retourne le N° de poche sélectionné par le mot T. Par défaut -1 (pas de poche sélectionnée).
- #<_value> - Retourne la valeur du dernier O-code `return` ou `endsub`. Valeur 0 par défaut si pas d'expression après `return` ou `endsub`. Initialisé à 0 au démarrage du programme.
- #<_value_returned> - 1.0 si le dernier O-code `return` ou `endsub` a retourné une valeur, 0 autrement. Effacé par le prochain appel à un O-code.
- #<_task> - 1.0 si l'instance en cours d'exécution par l'interpréteur fait partie d'une tâche de fraisage, 0.0 autrement. Il est parfois nécessaire de traiter ce cas particulier pour conserver un chemin d'outil propre, par exemple quand on teste le succès d'une mesure au palpeur (G38.x), en examinant #5070, ce qui ratait toujours dans le chemin d'outil de l'interpréteur (ex: Axis).
- #<_call_level> - current nesting level of O-word procedures. Pour débogage.
- #<_remap_level> - current level of the remap stack. Each remap in a block adds one to the remap level. Pour débogage.

5.2.14 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]]].

5.2.15 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: le ou (OR), le ou exclusif (XOR) et le et logique (AND). Le huitième opérateur est le modulo (MOD). Le neuvième opérateur est l'élévation à la puissance (**) qui élève 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. 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).

Précédence des opérateurs

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

5.2.16 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 le tableau ci-dessous. 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.

La fonction FIX arrondi un nombre vers la gauche, (moins positif ou plus négatif) par exemple, FIX[2.8]=2 et FIX[-2.8]=-3. La fonction FUP à l'inverse, arrondi un nombre vers la droite (plus positif ou moins négatif) par exemple, FUP[2.8]=3 et FUP[-2.8]=-2.

La fonction EXISTS vérifie l'existence d'un simple paramètre nommé. Il reçoit le paramètre à vérifier en argument, il retourne 1 si celui-ci existe et 0 sinon. C'est une erreur si un paramètre numéroté ou une expression est utilisé.

Table 5.2: Fonctions

Nom de fonction	**Fonction**
ATAN[Y]/[X]	Tangente quatre quadrants
ABS[arg]	Valeur absolue
ACOS[arg]	Arc cosinus
ASIN[arg]	Arc sinus
COS[arg]	Cosinus
EXP[arg]	Exposant
FIX[arg]	Arrondi à l'entier immédiatement inférieur

Table 5.2: (continued)

Nom de fonction	**Fonction**
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
EXISTS[arg]	Vérifie l'existence d'un paramètre nommé

5.2.17 Répétitions d'items

Une ligne peut contenir autant de mots G que voulu, mais un seul du même groupe modal.

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.

5.2.18 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).

5.2.19 Commandes et modes machine

En G-code, 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.

5.2.20 Coordonnées polaires

Des coordonnées polaires peuvent être utilisées pour spécifier les coordonnées XY d'un mouvement. Le @n est la distance et le ^n est l'angle. L'avantage est important, par exemple: Pour faire très simplement un cercle de trous tangents:

- Passer un point situé au centre du cercle
- Régler la compensation de longueur d'outil
- Déplacer l'outil vers le premier trou
- Enfin, lancer le cycle de perçage.

Les coordonnées polaires sont toujours données à partir de la position X0, Y0. Pour décaler les coordonnées polaires machine utilisez le décalage pièce ou sélectionnez un système de coordonnées.

En mode absolu, la distance et l'angle sont donnés à partir de la position X0, Y0 et l'angle commence à 0 sur l'axe X positif et augmente dans la direction trigonométrique (anti-horaire) autour de l'axe Z. Le code `G1 @1 ^90` est la même que `G1 Y1`.

En mode relatif, la distance et l'angle sont également donnés à partir de la position XY zéro, mais ils sont cumulatifs. Ce fonctionnement en mode incrémental peut être déroutant au début.

Par exemple: si vous avez le programme suivant, vous vous attendez à obtenir une trajectoire carré.

```
F100 G1 @.5 ^90
G91 @.5 ^90
@.5 ^90
@.5 ^90
@.5 ^90
G90 G0 X0 Y0 M2
```

Vous pouvez voir sur la figure suivante que la sortie n'est pas celle à laquelle vous vous attendiez, parce-que avons ajouté 0.5 à la distance de la position XY zéro à chaque début de ligne.

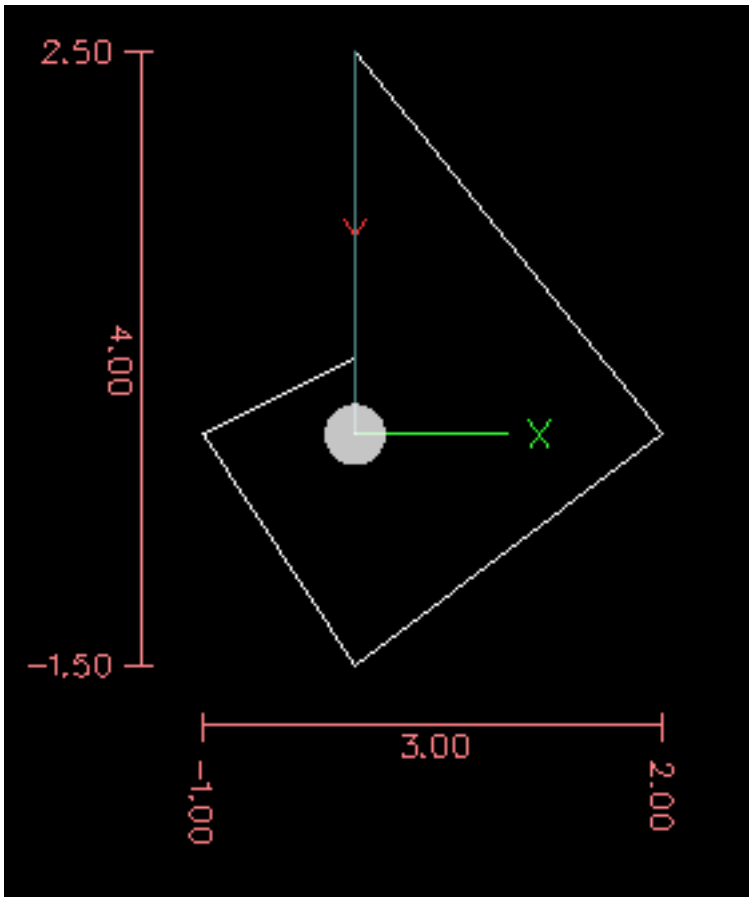


Figure 5.1: Spirale polaire

Le code suivant va produire notre modèle carré.

```
F100 G1 @.5 ^90
G91 ^90
^90
^90
^90
G90 G0 X0 Y0 M2
```

Comme vous pouvez le voir, en ajoutant seulement l'angle de 90 degrés à chaque ligne. La distance du point final est la même pour chaque ligne.

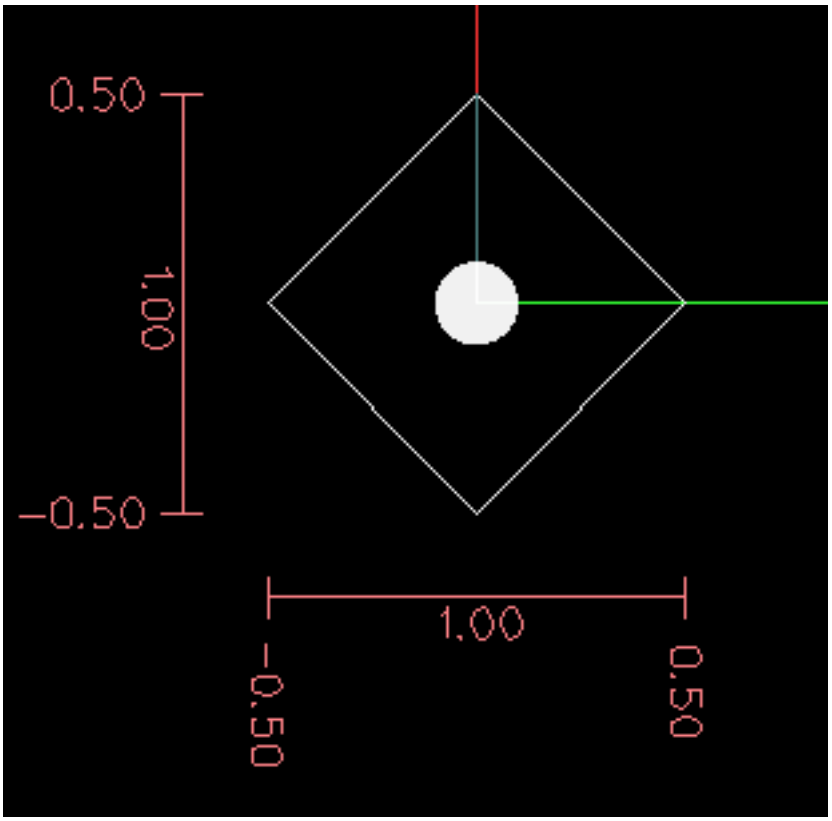


Figure 5.2: Carré polaire

C'est une erreur si:

- Un mouvement incrémental est lancé à l'origine.
- Un mélange de mots polaires et de X ou Y est utilisé.

5.2.21 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 un seul mode pour chaque groupe est actif. Les groupes modaux sont visibles dans le tableau ci-dessous.

Table 5.3: Groupes modaux des G-codes

Signification du groupe modal	**Mots G**
Codes non modaux (Groupe 0)	G4, G10, G28, G30, G53, G52, G92, G92.1, G92.2, G92.3
Mouvements (Groupe 1)	G0, G1, G2, G3, G33, G38.x, G73, G80, G81, G82, G83, G84, G85, G86, G87, G88, G89
Choix du plan de travail (Groupe 2)	G17, G18, G19, G17.1, G18.1, G19.1
Mode déplacement (Groupe 3)	G90, G91
Mode déplacement en arc IJK (Groupe 4)	G90.1, G91.1

Table 5.3: (continued)

Signification du groupe modal	**Mots G**
Mode de vitesses (Groupe 5)	G93, G94, G95
Unités machine (Groupe 6)	G20, G21
Compensation de rayon d'outil (Groupe 7)	G40, G41, G42, G41.1, G42.1
Compensation de longueur d'outil (Groupe 8)	G43, G43.1, G49
Plan de retrait cycle de perçage (Groupe 10)	G98, G99
Systèmes de coordonnées (Groupe 12)	G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3
Mode contrôle de trajectoire (Groupe 13)	G61, G61.1, G64
Mode contrôle vitesse broche (Groupe 14)	G96, G97
Mode diamètre/rayon sur les tours (Groupe 15)	G7, G8

Groupes modaux des M-codes

Signification du groupe modal	**Mots M**
Types de fin de programme (Groupe 4)	M0, M1, M2, M30, M60
On/Off I/O (Groupe 5)	M6 Tn
Appel d'outil (Groupe 6)	M6 Tn
Commande de broche (Groupe 7)	M3, M4, M5, M19
Arrosages (Groupe 8)	(M7, M8, peuvent être actifs simultanément), M9
Boutons de correction de vitesse (Groupe 9)	M48, M49, M50, M51
Définis par l'utilisateur (Groupe 10)	M100 à M199

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 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, G52 et G92.

C'est une erreur d'inclure des mots sans rapport sur une ligne avec le contrôle de flux O.

5.2.22 Commentaires

Des commentaires peuvent être ajoutés aux lignes de G-code pour clarifier l'intention du programmeur. Les commentaires peuvent être placés sur une ligne en les encadrant par des parenthèses. Ils peuvent aussi occuper tout le reste de la ligne à partir d'un point virgule. Le point virgule n'est pas traité comme un début de commentaire si il se trouve entre deux parenthèses.

Voici un exemple de programme commenté:

```
G0 (Rapide à démarrer.) X1 Y1
G0 X1 Y1 (Rapide à démarrer; mais n'oubliez pas l'arrosage.)
M2 ; Fin du programme.
```

Les commentaires peuvent se trouver entre des mots, mais pas entre des mots et leur paramètre correspondant. Ainsi, cette ligne est correcte:

```
S100(vitesse broche)F200(vitesse d'avance)
```

mais celle-ci est incorrecte:

```
S(speed)100F(feed)200
```

Les commentaires sont seulement informatifs, ils n'ont aucune influence sur la machine.

Il y a plusieurs commentaires actif qui ressemblent à un commentaire mais qui produit certaines actions, comme (debug,...) ou (print,...), expliqués plus loin. Si plusieurs commentaires se trouvent sur la même ligne, seul le dernier sera interprété selon les règles. Par conséquent, un commentaire normal suivant un commentaire actif aura pour effet de désactiver le commentaire actif. Par exemple, (foo) (debug,#1) affichera la valeur du paramètre #1, mais (debug,#1) (foo) ne l'affichera pas.

Un commentaire commençant par un point virgule est par définition le dernier commentaire sur cette ligne et sera toujours interprété selon la syntaxe des commentaires actifs.

5.2.23 Messages

- (MSG,) - 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 permises. Le reste du texte avant la parenthèse fermante est considéré comme un message. Les messages sont affichés sur la visu de l'interface utilisateur.

Exemple de message

```
(MSG, Ceci est un message)
```

5.2.24 Enregistrement des mesures

- (PROBEOPEN filename.txt) - ouvrira le fichier filename.txt et y enregistrera les 9 coordonnées de XYZABCUVW pour chacune des mesures réussie.
- (PROBECLOSE). - fermera le fichier de log palpeur.

Voir la section sur la mesure au palpeur pour d'autres informations sur le palpage avec G38.

5.2.25 Log général

- (LOGOPEN,filename.txt) - Ouvre le fichier de log filename.txt. Si le fichier existe déjà, il sera tronqué.
- (LOGAPPEND,filename.txt) - Ouvre le fichier de log filename.txt. Si le fichier existe déjà, il sera ajoutées.
- (LOGCLOSE) - Si le fichier est ouvert, il sera fermé.
- (LOG,message) - Le message placé derrière la virgule est écrit dans le fichier de log si il est ouvert. Supporte l'extension des paramètres comme décrit plus loin.

5.2.26 Messages de débogage

- (DEBUG,commentaire) sont traités de la même façon que ceux avec (msg,reste du commentaire) avec l'ajout de possibilités spéciales pour les paramètres, comme décrit plus loin.
- (PRINT,commentaire) vont directement sur la sortie stderr avec des possibilités spéciales pour les paramètres, comme décrit plus loin.

5.2.27 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 le contenu d'une variable nommée globale sur la sortie stderr (la fenêtre de la console par défaut), ajouter une ligne au G-code comme:

Exemple de paramètres en commentaire

```
(print,diamètre fraise 3 dents = #<_troisdents_dia>)
(print,la valeur de la variable 123 est: #123)
```

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>.

5.2.28 Exigences des fichiers

Un programme G-code doit contenir une ou plusieurs lignes de G-code puis se terminer par une ligne defin de programme. Tout G-code, placé après cette ligne de fin de programme, sera ignoré.

Si le programme n'utilise pas G-code de fin de programme, une paire de signes pourcent % peut être utilisées. Le premier signe % doit dans ce cas se trouver sur la première ligne du fichier, suivi par une ou plusieurs lignes de G-code, puis du second signe %. Tout G-code placé après le second signe % sera ignoré.

Note
Les fichiers de G-code doivent être créés avec un éditeur de texte comme Gedit et non avec un traitement de texte comme Open Office. Les traitements de texte ajoutent de nombreux caractères de contrôle dans les fichiers, ce qui les rends inutilisables comme programmes G-code.

5.2.29 Taille des fichiers

L'interpréteur et le gestionnaire de tâches ont été écrits, de sorte que la taille des fichiers n'est limité que par la capacité du disque dur. Les interfaces graphiques TkLinuxCNC et Axis affichent tous les deux le programme G-code à l'écran pour l'utilisateur, cependant, la RAM devient un facteur limitant. Dans Axis, parce-que l'aperçu du parcours d'outil est affiché par défaut, le rafraîchissement de l'écran devient une limite pratique à la taille des fichiers. Le tracé du parcours d'outil peut être désactivé dans Axis pour accélérer le chargement des fichiers conséquents. L'aperçu peut être désactivé en passant un commentaire spécial.

5.2.30 Ordre d'exécution

L'ordre d'exécution des éléments d'une ligne est défini, non pas par sa position dans la ligne mais par la liste suivante:

- Commandes O-code, optionnellement suivies par un commentaire mais aucun autre mot n'est permis sur la même ligne.
- Commentaire (message inclus).
- Positionnement du mode de vitesses (G93, G94).
- Réglage de la vitesse travail (F).
- Réglage de la vitesse de rotation de la broche (S).
- Sélection de l'outil (T).
- pin I/O de HAL (M62 à M68).
- Appel d'outil (M6).
- Marche/Arrêt broche (M3, M4, M5).
- Enregistrer l'état (M70, M73), restaurer l'état (M72), invalider l'état (M71).
- Marche/Arrêt arrosages (M7, M8, M9).
- Activation/Inhibition des correcteurs de vitesse (M48, M49, M50, M51, M52, M53).
- Commandes définies par l'opérateur (M100 à M199).
- Temporisation (G4).
- Choix du plan de travail (G17, G18, G19).
- Choix des unités de longueur (G20, G21).
- Activation/Désactivation de la compensation de rayon d'outil (G40, G41, G42)
- Activation/Désactivation de la compensation de longueur d'outil (G43, G49)
- Sélection du système de coordonnées (G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3).
- Réglage du mode de trajectoire (G61, G61.1, G64)
- Réglage du mode de déplacement (G90, G91).
- Réglage du mode de retrait (G98, G99).
- Prise d'origine (G28, G30) ou établissement du système de coordonnées (G10) ou encore, réglage des décalages d'axes (G52, G92, G92.1, G92.2, G94).
- Effectuer un mouvement (G0 à G3, G33, G80 à G89), tel que modifié (éventuellement) par G53.
- Arrêt (M0, M1, M2, M30, M60).

5.2.31 G-Code: Bonnes pratiques

5.2.31.1 Utiliser un nombre de décimales approprié

Utiliser 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.

5.2.31.2 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.

5.2.31.3 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.

5.2.31.4 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, système 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 de perçage peuvent être importantes.

5.2.31.5 Ne pas mettre trop de choses sur une ligne

Ignorer le contenu de la section ordre d'exécution et ne pas écrire de ligne de code qui laisse la moindre ambiguïté.

5.2.31.6 Ne pas régler et utiliser un paramètre sur la même ligne

Ne pas utiliser et définir un paramètre sur la même ligne, même si la sémantique est bien définie. Mettre à jour une variable, à une nouvelle valeur, telle que #1 = [#1 + #2] est autorisé.

5.2.31.7 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.

5.2.31.8 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'enlèvement de matière souhaité.

5.2.32 Axes rotatifs et linéaires

La signification du mot F-, exprimé en vitesse par minute, étant différente selon l'axe concerné par la commande de déplacement et parce-que la quantité de matière enlevée ne dépend pas seulement de la vitesse d'avance, il est facile d'utiliser le mode inverse du temps G93 pour atteindre la quantité de matériaux à enlever, souhaitée.

5.2.33 Messages d'erreur courants

- G code hors d'étendue - Un G-code supérieur à G99 a été utilisé. L'étendue des G-codes dans LinuxCNC est comprise entre 0 et 99. Toutefois, les valeurs entre 0 et 99 ne sont pas toutes celle d'un G-code valide.
- Utilisation d'un G code inconnu - Un G-code à été utilisé qui n'appartient pas aux langage G-code de LinuxCNC.
- Mot i, j, k sans Gx l'utilisant - Les mots i, j et k doivent être utilisés sur la même ligne que leur G-code.
- Impossible d'employer des valeurs d'axe sans G code pour les utiliser - Les valeurs d'axe ne peuvent pas être utilisées sur une ligne sans qu'un G-code ne se trouve sur la même ligne ou qu'un G-code modal soit actif.
- Le fichier se termine sans signe pourcent ni fin de programme - Tout fichier G-code doit se terminer par un M2, un M30 ou être encadré par le signe %.

5.3 Tout le G-code de LinuxCNC

5.3.1 Conventions d'écriture du G-code

Dans une commande type, le tiret (-) signifie une valeur réelle et les signes (<>) indiquent un item facultatif.

Si L- est écrit dans une commande, le signe - fera référence à Lnombre. De la même manière, le signe - dans H- peut être appelé le Hnombre et ainsi de suite pour les autres lettres. Une valeur facultative sera écrite <L->.

Dans les blocs de G-code, le mot axes signifie n'importe quel axe défini dans la configuration.

Une valeur réelle peut être:

- - un nombre explicite, 4 par exemple.
- - une expression, [2+2] par exemple.
- - une valeur de paramètre, #88 par exemple.
- - une fonction unaire de la valeur, acos[0] par exemple.

Dans la plupart des cas, si des mots d'axes sont donnés parmi XYZABCUVW, 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).

Les axes sont facultatifs, tout axe omis gardera sa valeur courante.

Tout item dans un bloc de G-code, non explicitement décrit comme facultatif, 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.

5.3.2 Table d'index du G-code

Sections	Descriptions
G0	Interpolation linéaire en vitesse rapide
G1	Interpolation linéaire en vitesse travail
G2/G3	Interpolation circulaire sens horaire/anti-horaire
G4	Temporisation
G5	Spline cubique
G5.1	B-Spline quadratique
G5.2 G5.3	NURBS, ajout point de contrôle
G7	Mode diamètre (sur les tours)
G8	Mode rayon (sur les tours)
G10 L1	Ajuste les valeurs de l'outil en table d'outils
G10 L10	Modifie les valeurs de l'outil dans la table d'outils
G10 L11	Fixe les valeurs de l'outil dans la table d'outils
G10 L2	Fixe l'origine d'un système de coordonnées
G10 L20	Fixe l'origine du système de coord. aux valeurs calculées
G17 G18 G19	Choix du plan de travail
G20 G21	Unités machine
G28 G28.1	Aller à une position prédéfinie
G30 G30.1	Aller à une position prédéfinie
G33	Mouvement avec broche synchronisée
G33.1	Taraudage rigide
G38	Mesures au palpeur
G40	Révocation de la compensation de rayon d'outil
G41 G42	Compensation de rayon d'outil
G41.1 G42.1	Comp. dynamique de rayon d'outil à gauche/à droite
G43	Compensation de longueur d'outil d'après une table d'outils
G43.1	Compensation dynamique de longueur d'outil
G49	Révocation de la compensation de longueur d'outil
G52	Offset du systéme de coordonnenées
G53	Déplacements en coordonnées machine (Absolues)
G54 à G59.3	Choix du système de coordonnées (1 à 9)
G61 G61.1	Mode trajectoire exacte/mode arrêts exacts
G64	Mode trajectoire continue avec tolérance
G73	Cycle de perçage avec brise copeau
G76	Cycle de filetage multipasses (tour)
G80	Révocation des codes modaux
G81	Cycle de perçage
G82	Autres cycles de perçage
G83	Perçage avec débourrage

Sections	Descriptions
G84	Taraudage à droite (pas encore implémenté)
G85	Alésage, retrait en vitesse travail
G86	Alésage, retrait en vitesse rapide
G87	Cycle d'alésage arrière (pas encore implémenté)
G88	Cycle alésage, Stop, Retrait manuel (pas encore implémenté)
G89	Cycle d'alésage avec tempo, recul vitesse travail
G90	Types de déplacement
G90.1 G91.1	Arc I,J,K, centre absolu ou relatif
G92	Décalages d'origines avec mise à jour des paramètres
G92.1 G92.2	Révocation des décalages d'origine
G92.3	Applique contenu des paramètres aux déc. d'origine
G93	Modes de vitesse
G96	Vitesse de coupe constante (IPM ou m/mn)
G97	Vitesse en tours par minute
G98	Options de retrait des cycles de perçage

5.3.3 G0 Interpolation linéaire en vitesse rapide

```
G0 axes
```

Pour un mouvement linéaire en vitesse rapide, programmer G0 axes, tous les mots d'axe sont facultatifs. Le G0 est facultatif si le mode mouvement courant est déjà G0. Cela produit 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. Un G0 seul peut être utilisé pour passer le mode de mouvement courant en G0.

Exemple avec G0:

```
G90 (Fixe les déplacements en mode absolu)
G0 X1 Y-2.3 (mouvement linéaire en vitesse rapide du point courant à X1 Y-2.3)
M2 (fin de programme)
```

- Voir les sections G90 et M2 pour plus d'informations.

Si la compensation d'outil est active, le mouvement sera différent de celui décrit ci-dessus, voir la section sur la compensation de d'outil.

Si G53 est programmé sur la même ligne, le mouvement sera également différent, voir la section sur les mouvements en coordonnées absolues.

C'est une erreur si:

- Un mot d'axe est indiqué sans valeur réelle.
- Un mot d'axe est indiqué qui n'est pas configuré.

5.3.4 G1 Interpolation linéaire en vitesse travail

```
G1 axes
```

Pour un mouvement linéaire en vitesse travail, (pour usiner ou non) programmer G1 axes, tous les mots d'axe sont facultatifs. Le G1 est facultatif si le mode de 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). Un G1 seul peut être utilisé pour passer le mode de mouvement courant en G1.

Exemple avec G1:

```
G90 (Fixe les déplacements en mode absolu)
G1 X1.2 Y-3 F10 (mouvement linéaire à 10 unités/mn du point courant à X1.2 Y-3)
Z-2.3 (mouvement linéaire à 10 unités/mn du point courant à Z-2.3)
Z1 F25 (mouvement linéaire de l'axe Z à 25 unités/mn vers Z1)
M2 (Fin de programme)
```

- Voir les sections G90 et M2 pour plus d'informations.

Si la compensation d'outil est active, le mouvement sera différent de celui décrit ci-dessus, voir la section sur la compensation d'outil. Si G53 est programmé sur la même ligne, le mouvement sera également différent, voir la section sur les mouvements en coordonnées absolues.

C'est une erreur si:

- - Aucune vitesse d'avance travail n'est fixée.
- - un mot d'axe est indiqué sans valeur réelle.
- - un mot d'axe est indiqué qui n'est pas configuré.

5.3.5 G2, G3 Interpolation circulaire en vitesse travail

```
G2 ou G3 axes décalages (format centre)
G2 ou G3 axes R- (format rayon)
G2 ou G3 décalages <P-> (cercles complet)
```

Un mouvement circulaire ou hélicoïdal est spécifié en sens horaire avec G2 ou en sens anti-horaire avec G3. La direction est vue depuis le côté positif de l'axe autour duquel le mouvement se produit.

Les axes de cercle ou les hélicoïdes, 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). Les plans 17,1, 18,1 et 19,1 ne sont pas actuellement pris en charge. Si l'arc est circulaire, il se trouve dans un plan parallèle au plan sélectionné.

Pour programmer un hélicoïde, inclure le mot d'axe perpendiculaire au plan de l'arc. Par exemple, si nous sommes dans le plan G17, inclure un mot Z, ceci provoquera un mouvement de l'axe Z vers valeur programmée durant tout le mouvement circulaire XY.

Pour programmer un arc supérieur à un tour complet, utiliser un mot P spécifiant alors le nombre de tours complets en plus de l'arc. Si P n'est pas spécifié, le comportement sera comme si P1 avait été donné: ceci étant, un seul tour complet ou partiel sera effectué, donnant un arc plus petit ou égal à un tour complet. Par exemple, si un arc de 180° est programmé avec P2, le mouvement résultant sera d'un tour et demi. Pour chaque incrément de P au delà de 1, un tour complet sera ajouté à l'arc programmé. Les arcs hélicoïdaux multitours sont supportés ce qui donne des mouvements très intéressants pour usiner des alésages ou des filetages.

Si une ligne de G-code crée un arc et inclus 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 sont rarement programmées.

Si la compensation d'outil est active, le mouvement sera différent de celui décrit ci-dessus, voir les sections sur G40 et sur G41-G42.

Le centre de l'arc est absolu ou relatif, tel que fixé par G90.1 ou G91.1, respectivement.

C'est une erreur si:

- Aucune vitesse d'avance travail n'est spécifiée.

Deux formats sont possibles pour spécifier un arc: Le format centre et le format rayon.

5.3.5.1 Arc au format centre (format recommandé)

Les arcs au format centre sont plus précis que les arcs au format rayon, c'est le format à privilégier.

La distance entre la position courante et le centre de l'arc et, facultativement, le nombre de tours, sont utilisés pour programmer des arcs inférieurs au cercle complet. Il est permis d'avoir le point final de l'arc égal à la position courante.

Le décalage entre le centre de l'arc et la position courante ainsi que facultativement, le nombre de tours, sont utilisés pour programmer des cercles complets.

Une erreur d'arrondi peut se produire quand un arc est programmé avec une précision inférieure à 4 décimales (0.0000) pour les pouces et à moins de 3 décimales (0.000) pour les millimètres.

Arc en mode distance relative Les décalages par rapport au centre de l'arc sont des distances relatives au point de départ de l'arc. Le mode distance relative de l'arc est le mode par défaut.

Un ou plusieurs mots d'axe et un ou plusieurs décalages doivent être programmés pour un arc qui fait moins de 360 degrés.

Aucun mot d'axe mais un ou plusieurs décalages doivent être programmés pour un cercle complet. Le mot P, par défaut à 1, est facultatif.

Pour d'avantage d'information sur les arcs en mode relatif, voir la section G91.1.

Arc en mode distance absolue Les décalages par rapport au centre de l'arc sont des distances absolues depuis la position 0 courante des axes (origine machine).

Un ou plusieurs mots d'axe et tous les décalages doivent être programmés pour les arcs de moins de 360 degrés.

Aucun mots d'axe mais tous les décalages doivent être programmés pour un cercle complet. Le mot P, par défaut à 1, est facultatif.

Pour d'avantage d'information sur les arcs en mode absolu, voir la section G90.1.

Plan XY (G17)

```
G2 ou G3 <X- Y- Z- I- J- P->
```

- Z - hélicoïde
- I - décalage en X
- J - décalage en Y
- P - nombre de tours

Plan XZ (G18)

```
G2 ou G3 <X- Z- Y- I- K- P->
```

- Y - hélicoïde
- I - décalage en X
- K - décalage en Z
- P - nombre de tours

YZ-plane (G19)

```
G2 ou G3 <Y- Z- X- J- K- P->
```

- X - hélicoïde
- J - décalage en Y
- K - décalage en Z
- P - nombre de tours

C'est une erreur si:

- Aucune vitesse d'avance travail n'est fixée avec le mot F.
- Aucun décalage n'est programmé.
- Quand l'arc est projeté dans le plan courant, la distance depuis le point courant et le centre diffère de la distance entre le point final et le centre, de plus de (.05 pouce/.5 mm) OU ((.0005 pouce/.005mm) ET .1% du rayon).

Déchiffrer le message d'erreur Le rayon à la fin de l'arc diffère de celui du début:

- début - position courante
- centre - la position du centre telle que calculée avec les paramètres I,J ou K
- fin - le point final programmé
- r1 - le rayon entre le point de départ et le centre
- r2 - le rayon entre le point final et le centre

5.3.5.2 Exemples d'arcs au format centre

Calculer des arcs à la main peut être difficile. Il est possible de dessiner l'arc à l'aide d'un programme de DAO pour obtenir les coordonnées et les décalages. Garder à l'esprit les tolérances, il pourrait être nécessaire de modifier la précision de la DAO pour obtenir les résultats souhaités. Une autre option consiste à calculer les coordonnées et les décalages en utilisant des formules. Comme vous pouvez le voir sur la figure suivante un triangle peut être formé à partir de la position courante, de la position de fin et du centre de l'arc.

Sur la figure suivante, vous voyez que la position de départ est X0 Y0, la position finale est X1 Y1. La position du centre de l'arc est X1 Y0. Ceci donne un décalage de 1 depuis la position de départ sur l'axe X et 0 sur l'axe Y. Dans ce cas seul le décalage I est nécessaire.

Le G-code de cet exemple serait:

```
G0 X0 Y0
G2 X1 Y1 I1 F10 (arc en sens horaire dans le plan XY)
```

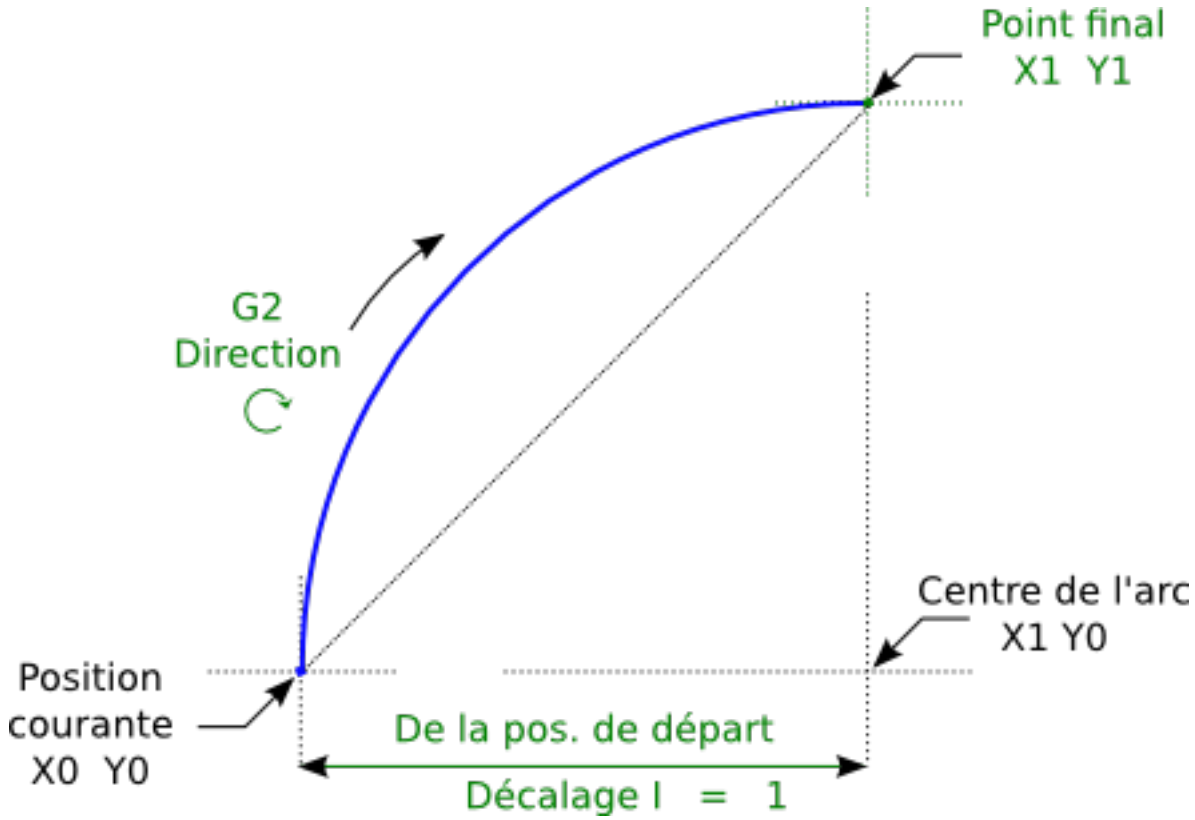


Figure 5.3: Exemple avec G2

Dans cet autre exemple, nous pouvons voir les différences de décalages pour Y selon que nous faisons un mouvement G2 ou un mouvement G3. Pour le mouvement G2 la position de départ est en X0 Y0, alors que pour le mouvement G3 elle est en X0 Y1. Le centre de l'arc est en X1 Y0.5 pour les deux. Le décalage J du mouvement G2 est 0.5 alors que celui du mouvement G3 est -0.5.

Le G-code de cet exemple serait:

```
G0 X0 Y0
G2 X0 Y1 I1 J0.5 F25 (arc en sens horaire dans le plan XY)
G3 X0 Y0 I1 J-0.5 F25 (arc en sens anti-horaire dans le plan XY)
```

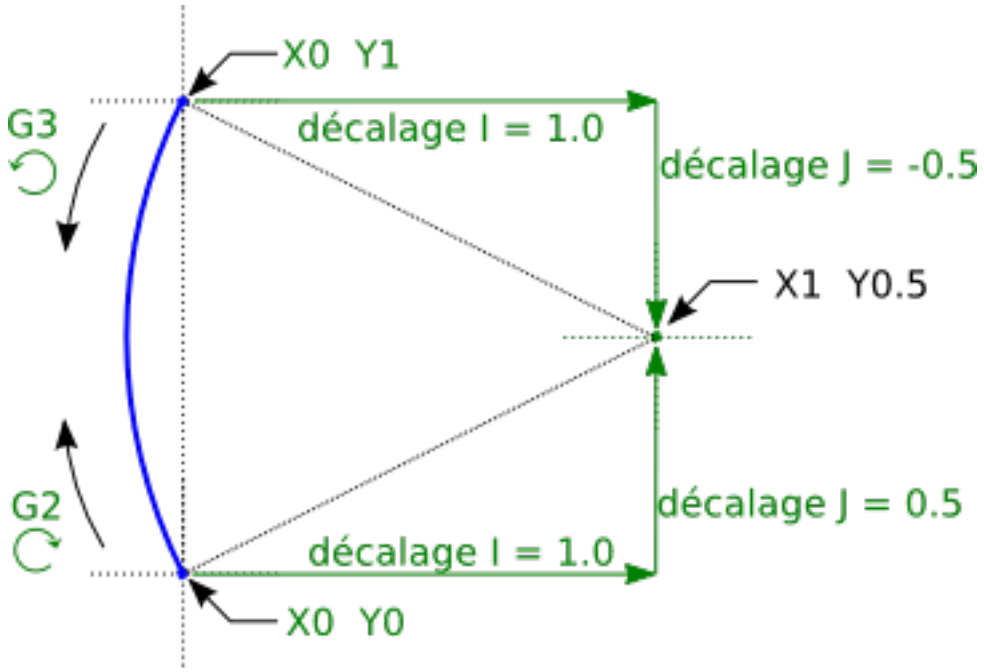


Figure 5.4: Exemple avec G2-G3

Voici un exemple au format centre pour usiner une hélice:

```
G0 X0 Y0 Z0
G17 G2 X10 Y16 I3 J4 Z-1 (Arc hélicoïdal avec ajout de Z)
```

exemple avec P

```
G0 X0 Y0 Z0
G2 X0 Y1 Z-1 I1 J0.5 P2 F25
```

Cet exemple signifie, faire un mouvement circulaire ou hélicoïdal en sens horaire (vu du côté positif sur l'axe Z), dont l'axe est parallèle à l'axe Z, se terminant en X10, Y16 et Z9, avec son centre décalé de 3 unités dans la direction X, par rapport à la position X courante. Son centre décalé dans la direction Y de 4 unités depuis la position Y courante. Si la position courante est X7, Y7 au départ, le centre sera en X10, Y11. Si la valeur de départ en Z est 9, ce sera un arc circulaire. Autrement, ce sera un arc hélicoïdal. Le rayon de cet arc serait de 5 unités.

Dans le format centre, le rayon de l'arc n'est pas spécifié, mais il peut facilement être trouvé puisque c'est la distance entre le point courant et le centre du cercle, ou le point final de l'arc et le centre.

5.3.5.3 Arcs au format rayon (format non recommandé)

```
G2 ou G3 axes R-
```

- R - rayon depuis la position courante

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, inférieurs à 165 degrés ou compris entre 195 et 345 degrés sera possible.

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.

Voici un exemple de commande pour usiner un arc au format rayon:

```
G17 G2 X10 Y15 R20 Z5 (arc au format rayon)
```

Cet exemple 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.

5.3.6 G4 Tempo

```
G4 P-
```

- P - durée de la temporisation en secondes (un flottant)

Les axes s'immobiliseront pour une durée de P secondes. Cette commande n'affecte pas la broche, les arrosages ni les entrées/sorties.

C'est une erreur si:

- Le nombre P est négatif ou n'est pas spécifié.

5.3.7 G5 Spline cubique

```
G5 X- Y- <I- J-> P- Q-
```

- I - offset incrémental en X, du point de départ au premier point de contrôle
- J - offset incrémental en Y, du point de départ au premier point de contrôle
- P - offset incrémental en X, du point de départ au second point de contrôle
- Q - offset incrémental en Y, du point de départ au second point de contrôle

G5 crée une B-spline cubique dans le plan XY avec les axes X et Y seuls. P et Q doivent être tous les deux spécifiés pour chaque commande G5.

Pour la première d'une série de commandes G5, I et J doivent être tous les deux spécifiés. Pour les commandes G5 suivantes de la série, soit I et J sont spécifiés tous les deux, soit aucun ne l'est. Si aucun n'est spécifié, la direction de départ de ce cube rejoindra automatiquement la direction de fin du cube précédent (comme si I et J étaient les négatifs des P et Q précédents).

Par exemple, pour programmer une courbe en forme de N:

G5 Simple spline cubique initiale

```
G90 G17
G0 X0 Y0
G5 I0 J3 P0 Q-3 X1 Y1
```

Une seconde courbe en N qui s'attache doucement à celle-ci peux maintenant être faite sans spécifier I et J:

G5 Simple spline cubique subséquente

```
G5 P0 Q-3 X2 Y2
```

C'est une erreur si:

- P et Q ne sont pas spécifiés tous les deux
- Un seul, de I ou J est spécifié
- Aucun de I ou J n'est spécifié à la première série de commandes G5
- Un axe autre que X ou Y est spécifié
- Le plan courant n'est pas G17

5.3.8 G5.1 Spline quadratique

```
G5.1 X- Y- I- J-
```

- I - Offset incrémental en X, du point de départ au point de contrôle
- J - Offset incrémental en Y, du point de départ au point de contrôle

G5.1 crée une B-spline quadratique dans le plan XY avec les seuls axes X et Y. Ne pas spécifier I ou J donne un offset nul pour l'axe non spécifié, un ou les deux doivent donc être donnés.

Par exemple, pour programmer une parabole, entre l'origine X-2 Y4 et X2 Y4:

G5.1 Simple spline quadratique

```
G90 G17
G0 X-2 Y4
G5.1 X2 I2 J-8
```

C'est une erreur si:

- Les offsets I et J ne sont pas spécifiés ou sont à zéro
- Un autre axe que X ou Y est spécifié
- Le plan actif n'est pas G17

5.3.9 G5.2 G5.3 Block NURBS

```
G5.2 <P-> <X- Y-> <L->
X- Y- <P->
...
G5.3
```

Warning
G5.2, G5.3 sont expérimentaux, il n'ont pas encore été testés totalement.

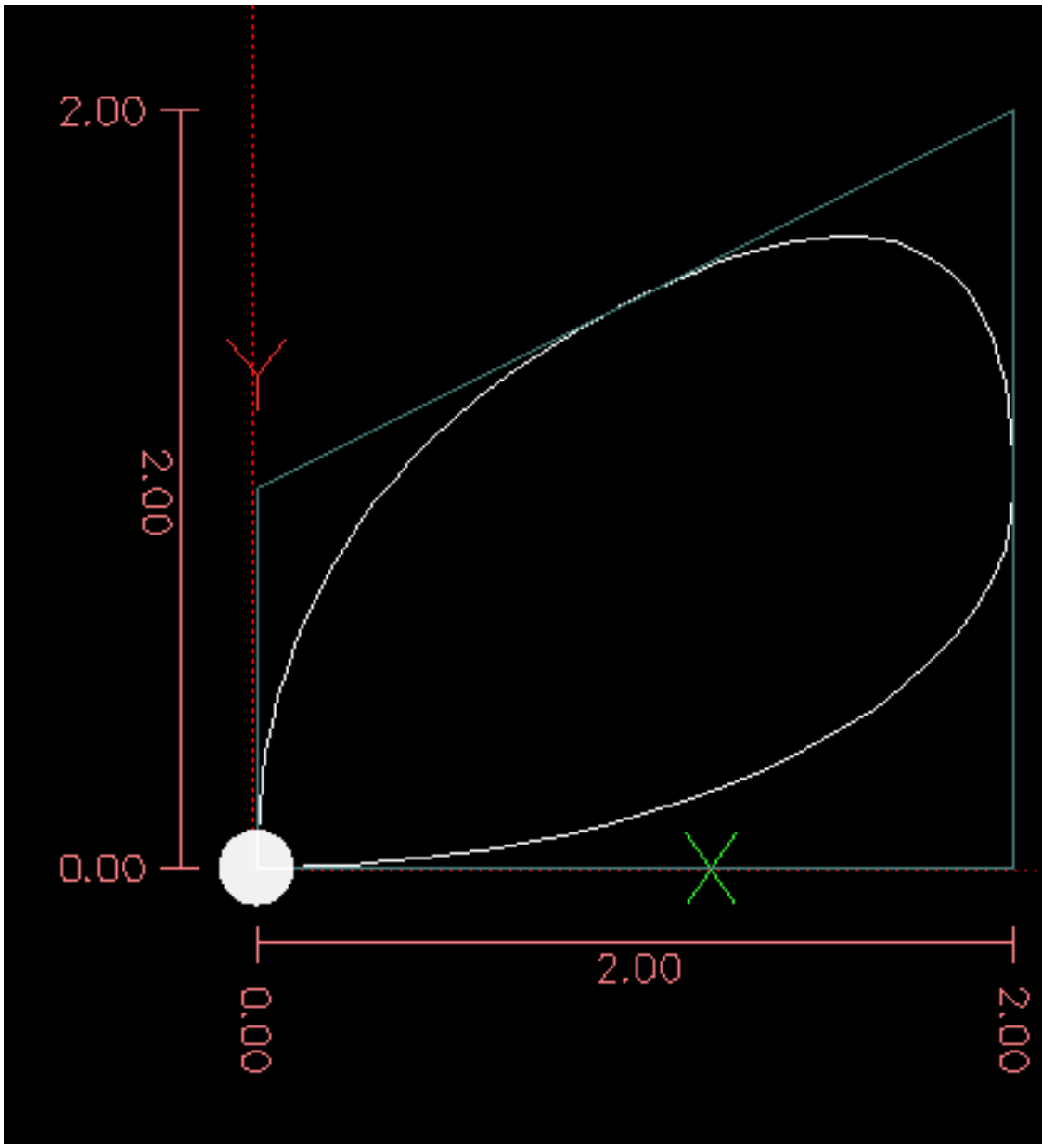
G5.2 est pour ouvrir un bloc de données définissant un NURBS et G5.3 pour fermer le bloc de données. Dans les lignes entre ces deux codes, les points de contrôle de la courbe sont définis avec deux éléments, leur poids relatif (P) et le paramètre (L) qui détermine l'ordre de la courbe.

Les coordonnées courantes, avant la premiére commande G5.2, est toujours prise comme premier point de contrôle du NURBS. Pour définir le poids pour le premier point de contrôle, premièrement programmer G5.2 P- sans donner X ni Y.

Le poids par défaut si P n'est pas spécifié est 1. L'ordre par défaut si L n'est pas spécifié est 3.

G5.2 Exemple

```
G0 X0 Y0 (mouvement en vitesse rapide)
F10 (set feed rate)
G5.2 P1 L3
     X0 Y1 P1
     X2 Y2 P1
     X2 Y0 P1
     X0 Y0 P2
G5.3
; Les mouvements en vitesse rapide montrent le même parcours sans le bloc NURBS
G0 X0 Y1
   X2 Y2
   X2 Y0
   X0 Y0
M2
```



Simple sortie NURBS

D’autres informations sur NURBS sont disponibles ici:

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

5.3.10 G7 Mode diamètre sur les tours

```
G7
```

Sur un tour, programmer G7 pour passer l’axe X en mode diamètre. En mode diamètre, les mouvements de l’axe X font la moitié de la cote programmée. Par exemple, X10 placera l’outil à 5 unités du centre, ce qui produira bien une pièce d’un diamètre de 10 unités.

5.3.11 G8 Mode rayon sur les tours

```
G8
```

Sur un tour, programmer G8 pour passer l'axe X en mode rayon. En mode rayon, les mouvements de l'axe X sont égaux à la cote programmée. Ce qui signifie que X10 placera l'outil à 10 unités du centre et aura pour résultat une pièce d'un diamètre de 20 unités. G8 est le mode par défaut à la mise sous tension.

5.3.12 G10 L1 Ajustements dans la table d'outils

```
G10 L1 P- axes <R- I- J- Q->
```

- P - numéro d'outil
- R - rayon de bec
- I - angle frontal (tour)
- J - angle arrière (tour)
- Q - orientation (tour)

G10 L1 ajuste les valeurs de la table d'outils pour l'outil N°P aux valeurs passées dans les paramètres. Les nouvelles valeurs peuvent être passées depuis un programme ou depuis la fenêtre d'entrées manuelles (MDI). Un G10 L1 valide, réécrit et recharge la table d'outils.

Exemples avec G10 L1:

```
G10 L1 P1 Z1.5 (fixe le décalage en Z de l'outil 1 à 1.5 de l'origine machine)
G10 L1 P2 R0.15 Q3 (fixe le rayon de bec de l'outil 2 à 0.15 avec une orientation 3)
```

C'est une erreur si:

- La compensation d'outil est active
- Le mot P n'est pas spécifié
- Le mot P ne correspond pas à un numéro d'outil valide de la table d'outils.

D'autres informations sur l'orientation des outils de tour sont disponibles ici.

5.3.13 G10 L2 Établissement de l'origine d'un système de coordonnées

```
G10 L2 P- <axes R->
```

- P - système de coordonnées (0 à 9)
- R - rotation autour de l'axe Z

G10 L2 décale l'origine des axes dans le système de coordonnées spécifié par la valeur du mot d'axe. Le décalage s'effectue à partir de l'origine machine établie par la prise d'origine machine (homing). Les valeurs de ce décalage vont remplacer toutes celles en effet sur le système de coordonnées spécifié. Les mots d'axe inutilisés resteront inchangés.

Programmer P0 à P9 pour spécifier le système de coordonnées à décaler.

Table 5.4: Systèmes de coordonnées

Valeur P	Système de coordonnées	G-code
0	Actif courant	n/a
1	1	G54
2	2	G55
3	3	G56
4	4	G57
5	5	G58
6	6	G59
7	7	G59.1
8	8	G59.2
9	9	G59.3

Facultativement, programmer R pour indiquer la rotation des axes XY autour de l'axe Z. La direction de rotation est anti-horaire comme vue depuis le côté positif de l'axe Z.

Tous les mots d'axe sont facultatifs.

Être en mode relatif (G91) est sans effet sur G10 L2.

Concepts importants:

- G10 L2 Pn ne change pas l'actuel système de coordonnées par celui spécifié par P, il est nécessaire d'utiliser G54 à 59.3 pour sélectionner le système de coordonnées.
- Quand un mouvement de rotation est en cours, jogger un axe, déplacera celui-ci seulement dans le sens négatif ou positif et non pas le long de l'axe de rotation.
- Si un décalage d'origine créé avec G92 ou G92 est actif avant la commande G10 L2, il reste actif 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. Si il est actif à ce moment là, les nouvelles coordonnées prennent effet immédiatement.

C'est une erreur si:

- Le nombre P n'est pas évalué comme étant un nombre entier compris entre 0 et 9.
- Un axe est programmé mais n'est pas défini dans la configuration.

Premier exemple avec G10 L2:

```
G10 L2 P1 X3.5 Y17.2
```

Place l'origine du premier système de coordonnées (celui sélectionné par G54) au points X3.5 et Y17.2 (en coordonnées absolues). La coordonnée Z de l'origine, ainsi que les coordonnées de tous les autres axes, restent inchangées puisque seuls X et Y étaient spécifiés.

Deuxième exemple avec G10 L2:

```
G10 L2 P1 X0 Y0 Z0 (révoque les décalages en X, Y et Z du système N°1)
```

L'exemple précédent fixe les origines XYZ du système de coordonnées G54, à l'origine machine.

Les systèmes de coordonnées sont décrits en détail ici.

5.3.14 G10 L10 modifie les offsets d'outil dans la table d'outils

```
G10 L10 P- axes <R- I- J- Q->
```

- P - numéro d'outil
- R - rotation autour de l'axe Z
- I - angle frontal (tour)
- J - angle arrière (tour)
- Q - orientation (tour)

G10 L10 modifie les valeurs de l'outil P dans la table d'outils, de sorte que si la compensation d'outil est rechargée, avec la machine à la position courante et avec les G5x et G52/G92 actifs, les coordonnées courantes pour l'axe spécifié deviendront les coordonnées spécifiées. Les axes non spécifiés dans la commande G10 L10 ne seront pas modifiés.

Exemple avec G10 L10:

```
M6 T1 G43 (appel l'outil 1 et active la correction de longueur d'outil)
G10 L10 P1 Z1.5 (fixe la position courante en Z à 1.5 dans la table d'outils)
G43 (recharge l'offset de longueur d'outil depuis la table d'outils modifiée)
M2 (fin de programme)
```

Pour d'autres détals voir les commandes M6, Tn et G43/G43.1.

C'est une erreur si:

- La compensation d'outil est activée.
- Le mot P n'est pas spécifié.
- Le mot P ne correspond pas à un numéro d'outil valide de la table d'outils.

5.3.15 G10 L11 modifie les offsets d'outil dans la table d'outils

```
G10 L11 P- axes <R- I- J- Q->
```

- P - numéro d'outil
- R - rotation autour de l'axe Z
- I - angle frontal (tour)
- J - angle arrière (tour)
- Q - orientation (tour)

G10 L11 est identique à G10 L10 excepté qu'au lieux de fixer les valeurs par rapport aux décalages de coordonnées courants, il les fixe de sorte que les coordonnées courantes deviennent celles spécifiées par les paramètres si la nouvelle compensation d'outil est rechargée et que la machine est placée dans le système de coordonnées G59.3, système sans aucun décalage G52/G92 actif.

Ceci permet à l'utilisateur de fixer le système de coordonnées G59.3 à un point fixe de la machine et d'utiliser cet emplacement pour mesurer l'outil sans s'occuper des autres décalages courants actifs.

C'est une erreur si:

- La compensation d'outil est activée
- Le mot P n'est pas spécifié.
- Le mot P ne correspond pas à un numéro d'outil valide de la table d'outils.

5.3.16 G10 L20 Établissement de l'origine d'un système de coordonnées

```
G10 L20 P- axes
```

- P - système de coordonnées (0-9)

G10 L20 est similaire à G10 L2 excepté qu'au lieu d'ajuster les offsets à des valeurs données, il les place à des valeurs calculées de sorte que les coordonnées courantes deviennent les valeurs données en paramètres.

Exemple avec G10 L20:

```
G10 L20 P1 X1.5 (fixe la position courante en X du système de coordonnées G54
à 1.5)
```

C'est une erreur si:

- Le nombre P n'est pas évalué comme une entier compris entre 0 et 9.
- Un axe non défini dans la configuration est programmé.

5.3.17 G17 à G19.1 Choix du plan de travail

Ces codes sélectionnent le plan de travail courant comme décrit ci-dessous:

- G17 - XY (par défaut)
- G18 - ZX
- G19 - YZ
- G17.1 - UV
- G18.1 - WU
- G19.1 - VW

Les plans UV, WU et VW ne supportent pas les arcs. Il est de bonne pratique d'inclure la sélection du plan de travail dans le préambule du programme G-code. Les effets de la sélection d'un plan de travail sont discutés dans la section sur les arcs.

5.3.18 G20, G21 Choix des unités machine

- G20 - pour utiliser le pouce comme unité de longueur.
- G21 - pour utiliser le millimètre comme unité de longueur.

C'est toujours une bonne pratique de programmer soit G20, soit G21, dans le préambule du programme, avant tout mouvement et de ne plus en changer ailleurs dans le programme.

5.3.19 G28, G28.1 Aller à une position prédéfinie

Warning
Pour une bonne répétabilité de la position et que la position soit correctement enregistrée avec G28.1, faire la prise d'origine générale avant d'utiliser G28.

G28 utilise les valeurs enregistrées dans les paramètres 5161 à 5166 comme points finaux des mouvements des axes X Y Z A B C U V W. Les valeurs des paramètres sont des coordonnées machine absolues, en unités machine natives, telles que fixées dans le fichier ini. Tous les axes définis dans le fichier ini seront déplacés lors d'un G28.

- G28 - effectue un mouvement en vitesse rapide de la position courante à la position absolue enregistrée dans les paramètres 5161 à 5166.
- G28 axes - effectue un déplacement en vitesse rapide à la position spécifiée par axes y compris les décalages, puis effectuera un mouvement en vitesse rapide aux coordonnées absolues stockées dans les paramètres 5161 à 5166 pour les axes spécifiés.
- G28.1 - enregistre la position absolue courante dans les paramètres 5161 à 5166.

Exemple avec G28

```
G28 Z2.5 (vitesse rapide vers Z2.5 puis emplacement spécifié dans les paramètres ↩
    enregistrés de G28)
```

C'est une erreur si:

- La compensation d'outil est active.

5.3.20 G30, G30.1 Aller à une position prédéfinie

Warning
Pour une bonne répétabilité de la position et que la position soit correctement enregistrée avec G30.1, faire la prise d'origine générale avant d'utiliser G30.

- G30 - effectue un mouvement en vitesse rapide de la position courante à la position absolue stockée dans les paramètres 5181 à 5186. Les valeurs stockées dans les paramètres font référence au système de coordonnées absolues qui est le système de coordonnées machine.
- G30 axes - effectue un déplacement en vitesse rapide depuis la position courante jusqu'à la position spécifiée par axes, y compris les décalages, suivi d'un mouvement rapide à la position absolue stockée dans les paramètres 5181 à 5186 pour les axes spécifiés. Les axes non spécifiés ne bougeront pas.
- G30.1 - enregistre la position absolue courante dans les paramètres 5181 à 5186.

Note
Les paramètres de G30 peuvent être utilisés pour déplacer l'outil quand un M6 est programmé avec la variable [TOOL_CHANGE_AT_G30]=1 dans la section [EMCIO] du fichier ini.

Exemple avec G30

```
G30 Z2.5 (mvt rapide à Z2.5 puis déplacement selon les paramètres de G30
stockés)
```

C'est une erreur si:

- La compensation de d'outil est active.

5.3.21 G33 Mouvement avec broche synchronisée

```
G33 X- Y- Z- K-
```

- K - distance par tour

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, si il commence à Z=0, G33 Z-1 K.0625 produira un mouvement d'un pouce de long en Z en même temps que 16 tours de broche. Cette commande peut 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.

Les mouvements avec broche synchronisée utilisent l'index de broche et les pins spindle at speed pour le filetage multi-passes. Un mouvement avec G33 se termine au point final programmé.

Note
K suit la ligne d'avance décrite par X- Y- Z-. K n'est pas parallèle à l'axe Z si les points d'arrivée des axes X et Y sont utilisés, par exemple pour réaliser un filetage conique.

Informations techniques Au début de chaque passe G33, LinuxCNC utilise la vitesse de broche et les limites d'accélération de la machine pour calculer combien de temps prendra Z pour accélérer après chaque impulsion d'index et détermine de combien de degrés la broche tournera pendant ce temps là. Il ajoute alors cet angle à la position de l'index puis calcule la position de Z utilisant l'angle de broche correct. Cela signifie que Z aura atteints la position correcte juste en fin d'accélération à la bonne vitesse, il peux immédiatement usiner le bon filetage.

Connections de hAL Les pins spindle.N.at-speed et l'index encoder.n.phase-Z pour la broche doivent être connectés dans le fichier HAL pour que G33 soit opérationnel. Voir le Manuel de l'intégrateur pour plus d'informations sur les mouvements synchronisés avec la broche.

Exemple avec G33:

```
G90 (mode distance absolue)
G0 X1 Z0.1 (positionnement en vitesse rapide)
S100 M3 (broche en rotation à 100tr/mn)
G33 Z-2 K0.125 (mouvement vers Z -2 avec une avance de 0.125 par tour)
G0 X1.25 (mouvement de dégagement en vitesse rapide)
Z0.1 (mouvement en vitesse rapide à Z0.1)
M2 (fin de programme)
```

- Voir les sections G90, G0 et M2 pour plus d'informations.

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.

5.3.22 G33.1 Taraudage Rigide

```
G33.1 X- Y- Z- K-
```

- K - distance par tour

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:

Warning
Si pour un taraudage rigide, les coordonnées X et Y spécifiées ne sont pas les coordonnées courantes lors de l'appel de G33.1, le mouvement ne s'effectuera pas le long de l'axe Z mais de la position courante jusqu'aux coordonnées X et Y spécifiées.

1. Un mouvement aux coordonnées spécifiées, synchronisé avec la rotation de la broche, avec le ratio donné et débutant à l'impulsion d'index du codeur de broche.
2. 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)
3. Le mouvement reste synchronisé en continu avec la broche, même au delà de la coordonnée du point final spécifié pendant l'arrêt de la broche et son inversion.
4. Le mouvement synchronisé se poursuit pour revenir aux coordonnées initiales.
5. Quand les coordonnées initiale sont atteintes, la commande inverse la broche une seconde fois (ex: de 300tr/mn sens anti-horaire à 300tr/mn en sens horaire)
6. Le mouvement reste synchronisé même au delà des coordonnées initiales pendant que la broche s'arrête, puis s'inverse.
7. Un mouvement non synchronisé ramène le mobile en arrière, aux coordonnées initiales.

Tous les mouvements avec broche synchronisée ont besoin d'un index de broche, pour conserver la trajectoire prévue et que les passes se chevauchent exactement. Un mouvement avec G33.1 se termine aux coordonnées initiales. Les mots d'axes sont facultatifs, sauf au moins un qui doit être utilisé.

Exemple avec G33.1:

```
G90 (mode distance absolue)
G0 X1.000 Y1.000 Z0.100 (mouvement rapide au point de départ taraudage rigide
en 20 filets par pouce)
G33.1 Z-0.750 K0.05 (et une profondeur de filet de 0.750)
M2 (fin de programme)
```

- Voir les sections G90, G0 et M2 pour plus d'informations.

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 d'une vitesse de broche trop élevée.

5.3.23 G38.x Mesure au palpeur

```
G38.x axes
```

- G38.2 - palpe vers la pièce, stoppe au toucher, signale une erreur en cas de défaut.
- G38.3 - palpe vers la pièce, stoppe au toucher.
- G38.4 - palpe en quittant la pièce, stoppe en perdant le contact, signal une erreur en cas de défaut.
- G38.5 - palpe en quittant la pièce, stoppe en perdant le contact.

Important
Cette commande n'est pas utilisable si la machine n'a pas été configurée pour exploiter un signal de sonde entre HAL et LinuxCNC. Le signal de la sonde doit être envoyé sur une broche d'entrée puis transmis à motion.probe-entrée (bit, In). G38.x utilise la valeur de cette broche pour déterminer quand la sonde a touché ou perdu le contact. TRUE si le contact de la sonde est fermé (Touché), FALSE si il est ouvert.

Programmer G38.x axes, pour effectuer une mesure au palpeur. Les mots d'axe sont facultatifs excepté au moins un. Les mots d'axe définissent ensemble, le point de destination, à partir de l'emplacement actuel, vers lequel la sonde se déplace. Si le palpeur n'a pas déclenché avant que la destination soit atteinte, G38.2 et G38.4 signaleront une erreur. L'outil dans la broche doit être un palpeur ou un actionneur de contact.

En réponse à cette commande, la machine déplace le point contrôlé (qui est le centre de la boule du stylet du palpeur) en ligne droite, à la vitesse travail courante, 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 (dans les limites d'accélération de la machine) lorsque le point programmé est atteint ou quand l'entrée du palpeur bascule dans l'état attendu selon la première éventualité.

Le tableau de signification des différents codes de mesure.

Table 5.5: Codes de mesure

Code	État ciblé	Sens de destination	Signal d'erreur
G38.2	Touché	Vers la pièce	Oui
G38.3	Touché	Vers la pièce	Non
G38.4	Quitté	Depuis la pièce	Oui
G38.5	Quitté	Depuis la pièce	Non

Après une mesure réussie, les paramètres 5061 à 5069 contiendront les coordonnées des axes XYZ-ABCUVW, pour l'emplacement du point contrôlé à l'instant du changement d'état du palpeur. Après une mesure manquée, ils contiendront les coordonnées du point programmé. Le paramètre 5070 est mis à 1 si la mesure est réussie et à 0 si elle est manquée. Si la mesure n'a pas réussi, G38.2 et G38.4 signaleront une erreur en affichant un message à l'écran si l'interface graphique choisie le permet.

Un commentaire de la forme (PROBEOPEN filename.txt) ouvrira le fichier filename.txt et y enregistrera les 9 coordonnées de XYZABCUVW pour chaque mesure réussie. Le fichier doit être fermé avec le commentaire (PROBECLOSE).

Dans le répertoire des exemples, le fichier smartprobe.ngc montre l'utilisation d'un palpeur et l'enregistreme des coordonnées de la pièce dans un fichier. Le fichier smartprobe.ngc peut être utilisé par ngcgui avec un minimum de modifications.

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 d'outil est activée.
- La vitesse travail est à zéro.
- Le palpeur est déjà au contact de la cible.

5.3.24 G40 Révocation de la compensation de rayon d'outil

- G40 - révoque la compensation de rayon d'outil. Le mouvement suivant, de sortie de compensation, doit être une droite au moins aussi longue que le diamètre de l'outil. Ce n'est pas une erreur de désactiver la compensation quand elle est déjà inactive.

Exemple avec G40

```
; la position courante est X1 après la fin du mvt avec compensation
G40 (révoque la compensation)
G0 X1.6 (mouvement linéaire aussi long que le diamètre d'outil)
M2 (fin de programme)
```

- Voir les sections G0 et M2 pour plus d'informations.

C'est une erreur si:

- Un mouvement en arc avec G2 ou G3 suit un G40.
- Le mouvement suivant la révocation de compensation est inférieur au diamètre de l'outil.

5.3.25 G41, G42 Compensation de rayon d'outil

```
G41 <D-> (compensation à gauche du profil)
G42 <D-> (compensation à droite du profil)
```

- D - Numéro d'outil

Le mot D est facultatif. En son absence ou si il est à zéro, le rayon de l'outil courant est utilisé. Si le mot D est présent, il devrait normalement correspondre au numéro de l'outil monté dans la broche, bien que cela ne soit pas indispensable, il doit par contre correspondre à un numéro d'outil valide.

Pour activer la compensation d'outil à gauche du profil, programmer G41. G41 applique la compensation d'outil à gauche de la ligne programmée vu de l'extrémité positive de l'axe perpendiculaire au plan.

Pour activer la compensation d'outil à droite du profil, programmer G42. G42 applique la correction d'outil à droite de la ligne programmée vu de l'extrémité positive de l'axe perpendiculaire au plan.

Le mouvement d'entrée doit être au moins aussi long que le rayon de l'outil. Le mouvement d'entrée peut être effectué en vitesse rapide.

La compensation d'outil ne peut être effectuée que si le plan XY ou le plan XZ est actif.

Les commandes définies par l'utilisateur, M100 à M199, sont autorisées lorsque la compensation d'outil est activée.

Le comportement de la machine, quand la compensation d'outil est activée, est décrit dans la section sur la compensation d'outil.

C'est une erreur si:

- Le nombre D ne correspond, ni à zéro, ni à un numéro d'outil valide.
- Le plan YZ est le plan de travail actif.
- La compensation d'outil est activée alors qu'elle est déjà active.

5.3.26 G41.1, G42.1 Compensation dynamique d'outil

```
G41.1 D- <L-> (à gauche du profil)
G42.1 D- <L-> (à droite du profil)
```

- Le mot D spécifie le diamètre de l'outil.
- Le mot L spécifie l'orientation de l'outil, est à 0 par défaut si non spécifié.

Pour activer la compensation dynamique d'outil à gauche du profil, programmer G41.1 D- L-.

Pour activer la compensation dynamique d'outil à droite du profil, programmer G42.1 D- L-.

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.
- Le nombre L est utilisée alors que le plan XZ n'est pas le plan actif.
- La compensation d'outil est activée alors qu'elle est déjà active.

Plus d'informations sur l'orientation des outils, sur les outils de tour en 1-2-3-4 et les outils de tour en 5-6-7-8.

5.3.27 G43 Activation de la compensation de longueur d'outil

- H - Numéro d'outil
- G43 - Utilise l'outil courant chargé par le dernier Tn M6. G43 modifie les mouvements ultérieurs en décalant les coordonnées de Z et/ou de X, de la longueur de l'outil. G43 ne provoque aucun mouvement. L'effet de la compensation ne se produira qu'au cours du prochain mouvement des axes compensés, de sorte que le point final de ce mouvement sera la position compensée.
- G43 H- - Utilise l'offset de l'outil correspondant fourni par la table d'outils. Ce n'est pas une erreur d'avoir la valeur de H à zéro, le numéro de l'outil courant sera utilisé.

Exemple de ligne avec G43 H-

```
G43 H1 (ajuste les offsets d'outil avec les valeurs de l'outil 1 fournies par
la table d'outils)
```

C'est une erreur si:

- La valeur de H n'est pas un entier, il est négatif, ou il ne correspond, ni à zéro, ni à un numéro d'outil valide.

5.3.28 G43.1 Compensation dynamique de longueur d'outil

```
G43.1 axes
```

- G43.1 axes - Modifie les mouvements ultérieurs en décalant les coordonnées de Z et/ou de X, selon les offsets stockés dans la table d'outils. G43.1 ne provoque aucun mouvement. L'effet de la compensation ne se produira qu'au cours du prochain mouvement des axes compensés de sorte que le point final de ce mouvement sera la position compensée.

Exemple avec G43.1

```
G90 (passe en mode absolu)
T1 M6 G43 (charge l'outil N°1 et son offset de longueur, Z est à la position
machine 0 et la visu affiche Z1.500)
G43.1 Z0.250 (décale l'outil courant de 0.250, la visu affiche maintenant
Z1.250)
M2 (fin de programme)
```

- Voir les sections G90 & T et M2 pour plus d'informations.

C'est une erreur si:

- Une commande de mouvement est sur la même ligne que G43.1

5.3.29 G49 Révocation de la compensation de longueur d'outil

Pour révoquer 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 de révoquer une compensation de longueur d'outil alors qu'aucune n'est couramment utilisée.

5.3.30 G52 Local Coordinate System Offset

```
G52 axes
```

G52 is used in a part program as a temporary "local coordinate system offset" within the workpiece coordinate system. More information on G52 is in the Local and Global Offsets section.

5.3.31 G53 Mouvement en coordonnées absolues

```
G53 axes
```

Pour un déplacement exprimé en coordonnées système, programmer G53 sur la même ligne qu'un mouvement linéaire. G53 n'est pas modal, il doit donc être programmé sur chaque ligne où il doit être actif. G0 ou G1 ne doivent pas se trouver sur la même ligne si un d'eux est déjà actif. Par exemple:

Exemple avec G53

```
G53 G0 X0 Y0 Z0 (mouvement linéaire rapide des axes à leur positions d'origine)
G53 X2 (mouvement linéaire rapide à la coordonnée absolue X=2)
```

C’est une erreur si:

- G53 est utilisé sans que G0 ou G1 ne soit actif.
- G53 est utilisé alors que la compensation d’outil est active.

Étudier le chapitre sur les systèmes de coordonnées et de leurs décalages, pour bien maîtriser ces concepts.

5.3.32 G54 à G59.3 Choix du système de coordonnées

- G54 - Système de coordonnées pièce 1
- G55 - Système de coordonnées pièce 2
- G56 - Système de coordonnées pièce 3
- G57 - Système de coordonnées pièce 4
- G58 - Système de coordonnées pièce 5
- G59 - Système de coordonnées pièce 6
- G59.1 - Système de coordonnées pièce 7
- G59.2 - Système de coordonnées pièce 8
- G59.3 - Système de coordonnées pièce 9

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 stockent les valeurs de chacun des axes dans les variables indiquées dans le tableau ci-dessous.

Table 5.6: Paramètres des systèmes de coordonnées pièce

Choix	CS	X	Y	Z	A	B	C	U	V	W	R
G54	1	5221	5222	5223	5224	5225	5226	5227	5228	5229	5230
G55	2	5241	5242	5243	5244	5245	5246	5247	5248	5249	5250
G56	3	5261	5262	5263	5264	5265	5266	5267	5268	5269	5270
G57	4	5281	5282	5283	5284	5285	5286	5287	5288	5289	5290
G58	5	5301	5302	5303	5304	5305	5306	5307	5308	5309	5310
G59	6	5321	5322	5323	5324	5325	5326	5327	5328	5329	5330
G59.1	7	5341	5342	5343	5344	5345	5346	5347	5348	5349	5350
G59.2	8	5361	5362	5363	5364	5365	5366	5367	5368	5369	5370
G59.3	9	5381	5382	5383	5384	5385	5386	5387	5388	5389	5390

C’est une erreur si:

- Un de ces G-codes est utilisé alors que la compensation d’outil est active.

Voir la section sur les systèmes de coordonnée pour une vue complète.

5.3.33 G61, G61.1 Contrôle de trajectoire exacte

- G61 - Met la machine en mode de trajectoire exacte. G61 suivra exactement la trajectoire programmée même si cela doit aboutir à un arrêt complet momentané du mobile.
- G61.1 - Met la machine en mode arrêts exacts.

5.3.34 G64 Contrôle de trajectoire continue avec tolérance

```
G64 <P- <Q->>
```

- P- - Déviation maximale tolérée par rapport à la trajectoire programmée.
- Q- - Tolérance naïve cam.
- G64 - Recherche de la meilleure vitesse possible.
- G64 P- - Mélange entre meilleure vitesse et tolérance de déviation.
- G64 P- Q- - Est le moyen d'affiner encore pour obtenir le meilleur compromis entre vitesse et précision de la trajectoire. La vitesse sera réduite si nécessaire pour maintenir la trajectoire, même si ça doit aboutir à un arrêt complet momentané. Le détecteur naïve cam est activé. Quand il y a une série de mouvements linéaires XYZ en vitesse travail, avec une même vitesse de déplacement, inférieure à Q-, ils sont regroupés en un seul segment linéaire, ainsi la vitesse s'en trouve améliorée puisqu'il n'y a plus de décélération/arrêt/accélération aux points de jonction des segments. Sur les mouvements G2/G3 dans le plan G17 (XY) lorsque le maximum d'écart entre un arc et une ligne droite est inférieur à la déviation maximale P-, la tolérance de l'arc est divisée en deux lignes (depuis le début de l'arc jusqu'au milieu et du milieu jusqu'à la fin). Ces deux lignes sont ensuite soumises à l'algorithme naïve cam. Ainsi, les cas ligne-arc, arc-arc et arc-ligne et le cas ligne-ligne, bénéficient de l'algorithme naïve cam, ce qui améliore les performances en simplifiant les trajectoires. Il est permis de programmer ce mode même si il est déjà actif.

Exemple de ligne de programme avec G64

```
G64 P0.015 (fixe la déviation d'usinage à 0.015 maximum de la trajectoire
programmée)
```

Il est de bonne pratique de spécifier un type de contrôle de trajectoire dans le préambule de chaque programme G-code.

5.3.35 G73 Cycle de perçage avec brise copeaux

```
G73 axes R- Q- <L->
```

- R- - Position du plan de retrait en Z
- Q- - Incrément delta parallèle à l'axe Z
- L- - Répétition

Le cycle G73 est destiné au perçage profond ou au fraisage avec brise-copeaux. Les retraits, au cours de ce cycle, fragmentent les copeaux longs (fréquents lors de l'usinage de l'aluminium). Ce cycle utilise la valeur Q- qui représente un incrément delta parallèle à l'axe Z. Le cycle se décompose de la manière suivante:

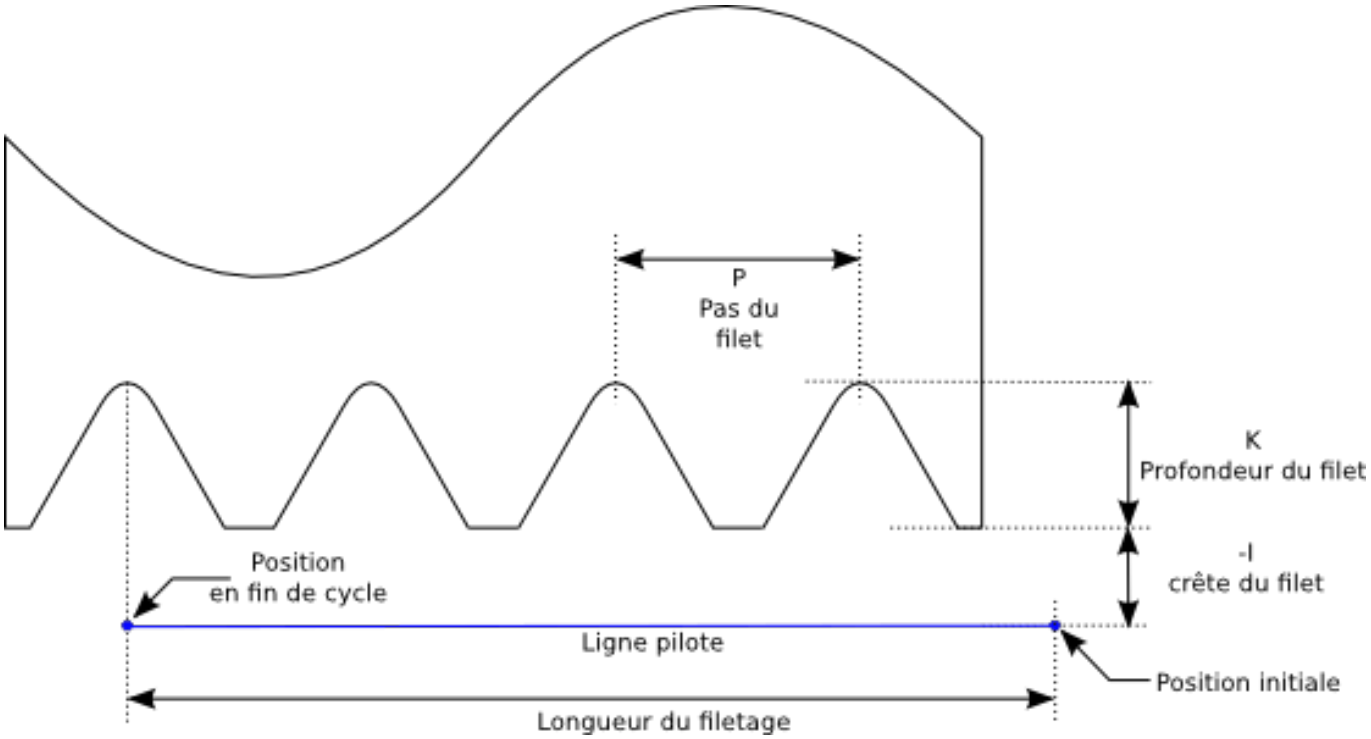
1. Un mouvement préliminaire. Comme décrit dans cet exposé sur le mouvement préliminaire
2. Un mouvement de l'axe Z seul, en vitesse travail, sur la position la moins profonde entre, l'incrément delta ou la position de Z programmée.
3. Une petite remontée en vitesse rapide.
4. Répétition des étapes 2 et 3 jusqu'à ce que la position programmée de Z soit atteinte à l'étape 2.
5. Un mouvement de l'axe Z en vitesse rapide jusqu'au plan de retrait.

C'est une erreur si:

- La valeur de Q est négative ou égale à zéro.
- Le nombre R n'est pas spécifié.

5.3.36 G76 Cycle de filetage préprogrammé

```
G76 P- Z- I- J- R- K- Q- H- E- L-
```



- Ligne pilote - La ligne pilote est une ligne imaginaire, parallèle à l'axe de la broche (Z), 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.
- P- - Le pas du filet en distance de déplacement par tour.
- Z- - La position finale du filetage. A la fin du cycle, l'outil sera à cette position Z.

Note
En mode diamètre G7, les valeurs I, J et K sont des mesures de diamètre. En mode rayon G8, les valeurs I, J et K sont des mesures de rayon.

- I- - La crête du filet est une distance 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.
- J- - Une valeur positive, spécifie la profondeur de la passe initiale. La première passe sera à J au delà de la crête du filet I.
- K- - Une valeur positive, spécifie la profondeur finale du filet. La dernière passe du filetage sera à K au delà de la crête du filet I.

Paramètres facultatifs:

- R- - La profondeur de dégressivité. R1.0 spécifie une profondeur de passe constante pour les passes successives du filetage. R2.0 spécifie une surface constante. Les valeurs comprises entre 1.0 et 2.0 spécifient une profondeur décroissante mais une surface croissante. Enfin, les valeurs supérieures à 2.0 sélectionnent une surface décroissante.

Warning
Les valeurs inutilement hautes de dégressivité, produiront un nombre inutilement important de passes. (dégressivité = plongée par paliers)

- Q- - L'angle de pénétration oblique. C'est l'angle (en degrés) décrivant de combien, les passes successives doivent être décalées le long de l'axe Z. 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 de l'outil. Typiquement, les valeurs sont 29, 29.5 ou 30 degrés.
- H- - Le nombre de passes de finition. 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 coniques avec les valeurs de E et L.

- E- - Spécifie la longueur des parties coniques le long de l'axe Z. L'angle du cône ira de la profondeur de la dernière passe à la crête du filet I. E2.0 donnera un cône d'entrée et de sortie d'une longueur de 2.0 unités dans le sens du filetage. Pour un cône à 45 degrés, programmer E identique à K.
- L- - Spécifie quelles extrémités du filetage doivent être coniques. Programmer L0 pour aucune (par défaut), L1 pour une entrée conique, L2 pour une sortie conique, ou L3 pour l'entrée et la sortie coniques.

L'outil fera une brève pause pour la synchronisation avec l'impulsion d'index avant chaque passe de filetage. 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 initial) 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 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.

La position finale de l'outil sera à la fin de la ligne pilote. Un mouvement de sécurité peut être nécessaire avec un filetage interne, pour sortir l'outil de la pièce.

C'est une erreur si:

- Le plan de travail actif n'est pas ZX.

- D'autres mots d'axes que X ou Y, sont spécifiés.
- La dégressivité R est inférieure à 1.0.
- Tous les mots requis ne sont pas spécifiés.
- P, J, K ou H est négatif.
- E- est supérieur à la moitié de la longueur de la ligne pilote.

Connections de HAL Les pins spindle.N.at-speed et l'index encoder.n.phase-Z doivent être connectées dans le fichier HAL pour que G76 soit opérationnel. Voir le Manuel de l'intégrateur pour plus d'informations sur les mouvements synchronisés avec la broche.

Informations techniques Le cycle préprogrammé G76 est basé sur le mouvement avec broche synchronisée G33, voir les informations technique relatives à G33.

Un programme de filetage, 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.

Exemple de G-Code avec G76

```
G0 Z-0.5 X0.2
G76 P0.05 Z-1 I-0.075 J0.008 K0.045 Q29.5 L2 E0.045
```

Sur l'image ci-dessous, l'outil est à la position finale après que le cycle G76 soit terminé. On voit que le parcours d'entrée de l'outil sur la droite, spécifié par Q29.5 et le parcours de sortie conique à gauche comme spécifié par L2 E0.045. Les lignes blanches sont les mouvements de coupe.

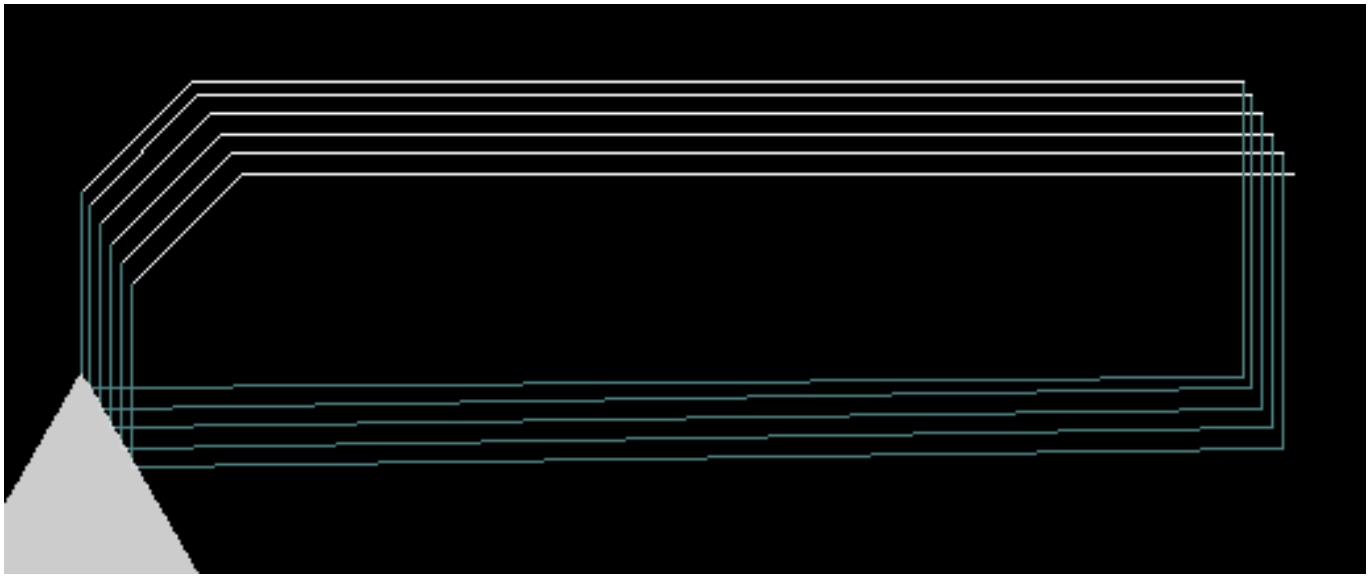


Figure 5.5: Parcours d'outil de l'exemple

5.3.37 Les cycles de perçage G81 à G89

Les cycles de perçage de G81 à G89 et la révocation de ces cycle G80, sont décrits dans cette section. Des exemples sont donnés plus bas avec les descriptions.

Tous les cycles de perçage sont effectués dans le respect du plan de travail courant. N'importe lequel des six plans de travail 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 autres plans

de travail et les mots corrects doivent être utilisés. Par exemple, dans le plan G17.1, l'action de retrait s'effectue parallèlement à l'axe W et les positions ou incréments sont donnés avec U et W. Dans ce cas, substituer U, V, W avec X, Y, Z dans les instructions suivantes.

Les mots d'axes rotatifs ne sont pas autorisés dans les cycles de perçage. Quand le plan actif est X, Y, Z, les mots d'axes U, V, W ne sont pas autorisés. De même, si le plan actif est U, V, W, les mots d'axes X, Y, Z ne sont pas autorisés.

5.3.37.1 Mots communs

Tous les cycles de perçage utilisent les groupes X, Y, Z ou U, V, W selon le plan sélectionné, ainsi que le mot R. La position de R- (signifiant retrait) 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, etc.). Quelques cycles de perçage utilisent des arguments supplémentaires.

5.3.37.2 Mots sticky

Dans les cycles de perçage, un nombre est qualifié de sticky (persistante, collant) si, quand le même cycle est répété sur plusieurs lignes de code en colonne, le nombre doit être indiqué la première fois, mais il devient facultatif 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), les valeurs X, Y, est R sont traitées comme des incréments depuis la position courante, Z est un incrément depuis la position de l'axe Z avant le mouvement impliquant l'axe Z. En mode de déplacements absolus, les valeurs de X, Y, R, et Z sont des positions absolues dans le système de coordonnées courant.

5.3.37.3 Répétition de cycle

Le mot L est facultatif et représente le nombre de répétitions. L=0 n'est pas permis. Si les fonctionnalités de répétition sont utilisées, elles le sont normalement en mode relatif, de sorte que la même séquence de mouvements se répète à plusieurs emplacements régulièrement espacés le long d'une ligne droite. Quand L>1 en mode relatif 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. Ainsi, si vous programmez `L10`, vous obtiendrez 10 cycles. Le premier cycle sera la distance X, Y depuis la position d'origine. Les positions de R- et Z- ne changent pas durant toutes les répétitions. En mode absolu, L>1 signifie `faire le même cycle à la même place plusieurs fois`, omis, le mot L est équivalent à L=1. La valeur de L n'est pas sticky.

5.3.37.4 Mode de retrait

La hauteur du mouvement de retrait à la fin de chaque répétition (appelée plan de retrait dans les descriptions suivantes) est déterminée par le mode de retrait: 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 sur les options du plan de retrait.

5.3.37.5 Erreurs des cycles de perçage

Il y a une erreur si:

- Tous les mots X, Y et Z sont manquants durant un cycle de perçage.

- Des mots d’axes de différents groupes (XYZ) (UVW) sont utilisés.
- 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 de perçage.
- Une vitesse inverse du temps est activée durant un cycle de perçage.
- La compensation d’outil est activée durant un cycle de perçage.

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 de perçage n’a pas encore été activé.
- La valeur de R est inférieure à celle de Z.

Si un autre plan est actif, les conditions d’erreur sont analogues à celles du plan XY décrites ci-dessus.

5.3.37.6 Mouvement préliminaire et Intermédiaire

Le mouvement préliminaire est un ensemble de mouvements commun à tous les cycles de perçage.

Tout au début de l’exécution d’un cycle de perçage, 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 autre plan est actif, le mouvement préliminaire et intermédiaire est analogue.

5.3.37.7 Pourquoi utiliser les cycles de perçage?

Il y a au moins deux raisons pour utiliser les cycles de perçage. La première est l’économie de code et la seconde la sécurité offerte par le mouvement préliminaire qui permet de ne pas s’occuper du point de départ du cycle.

5.3.38 G80 Révocation des codes modaux

- G80 - Révoque, tant qu’il est actif, tous les codes de mouvements modaux du groupe 1 auquel il appartient. Il est révoqué lui même par tout g-code du même groupe.

C’est une erreur si:

- Des mots d’axes sont programmés quand G80 est actif.

Exemple 1 avec G80:

```
G90 G81 X1 Y1 Z1.5 R2.8 (cycle de perçage en mode de déplacement absolu)
G80 (révoque G81)
G0 X0 Y0 Z0 (active les mouvements en vitesse rapide et déplace le
mobile en X0, Y0 et Z0)
```

L'exemple 1 produit les mêmes déplacements et le même état final de la machine que l'exemple suivant:

Exemple avec G0:

```
G90 G81 X1 Y1 Z1.5 R2.8 (cycle de perçage en mode de déplacement absolu)
G0 X0 Y0 Z0 (active les mouvements en vitesse rapide et déplace le
mobile en X0, Y0 et Z0)
```

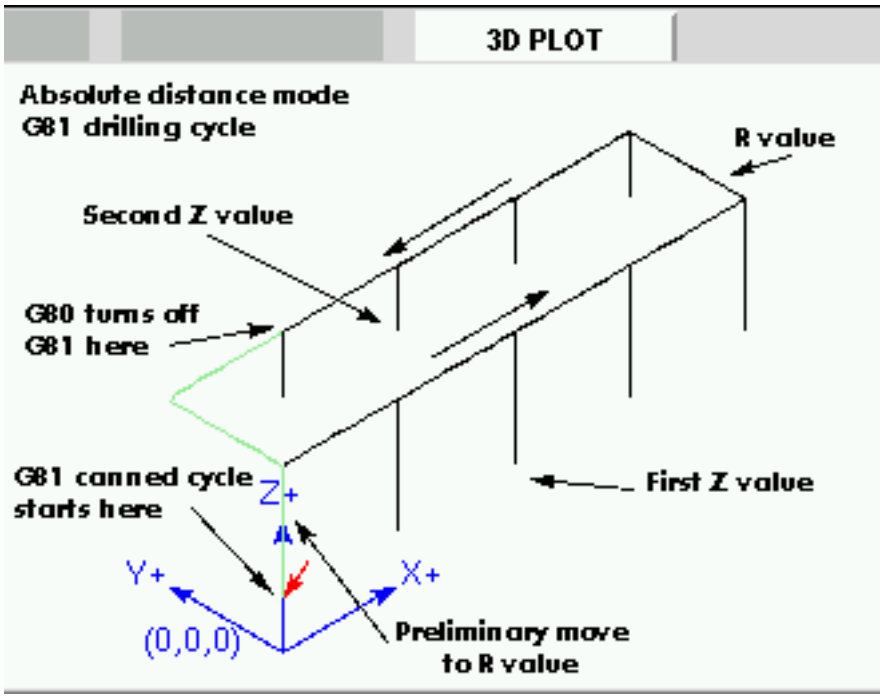
L'avantage du premier exemple est que la ligne du G80 révoque clairement le cycle G81. Avec ce premier programme, le programmeur doit revenir en mode mouvement avec G0, comme c'est fait sur la ligne suivante, ou tout autre mot G de mouvement.

Si un cycle de perçage n'est pas révoqué avec G80 ou un autre mot G de mouvement, le cycle de perçage attend de se répéter en utilisant la prochaine ligne de code contenant un ou plusieurs mots d'axe. Le fichier suivant perce (G81) un ensemble de huit trous, tel que montré sur l'image qui suit.

Exemple 2 avec G80:

```
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
N230 Z0
N240 M2 (fin du programme)
```

Note
Noter que la position de Z change après les quatre premiers trous. C'est également un des rares cas dans lesquels les numéros de lignes sont présents, permettant d'envoyer le lecteur sur une ligne de code spécifique.



L'utilisation du G80 de la ligne N200 est facultative puisqu'il y a un G0 sur la ligne suivante qui révoque le cycle G81. Mais utiliser G80, comme l'exemple 2 le montre, donne une meilleure lisibilité au programme. Sans ce G80, il ne serait pas aussi évident que tous les blocs compris entre N120 et N200 appartiennent au cycle de perçage.

5.3.39 G81 Cycle de perçage

```
G81 (X- Y- Z- ) ou (U- V- W- ) R- L-
```

Le cycle G81 est destiné au perçage.

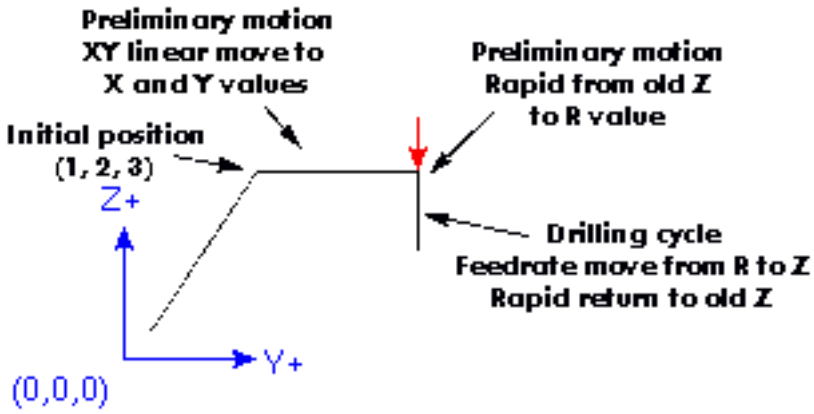
1. Un mouvement préliminaire, comme décrit sur cette page.
2. Un déplacement de l'axe Z seul à la vitesse programmée, vers la position Z programmée.
3. Retrait de l'axe Z en vitesse rapide jusqu'au plan de retrait R.

Exemple 1: G81 en position absolue

Supposons que la position courante soit, X1, Y2, Z3 dans le plan XY, la ligne de code 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.



- Un mouvement en vitesse rapide, parallèle au plan XY vers X4, Y5, Z3
- Un mouvement en vitesse rapide, parallèle à l'axe Z vers X4, Y5, Z2.8
- Un mouvement en vitesse travail, parallèle à l'axe Z vers X4, Y5, Z1.5
- 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

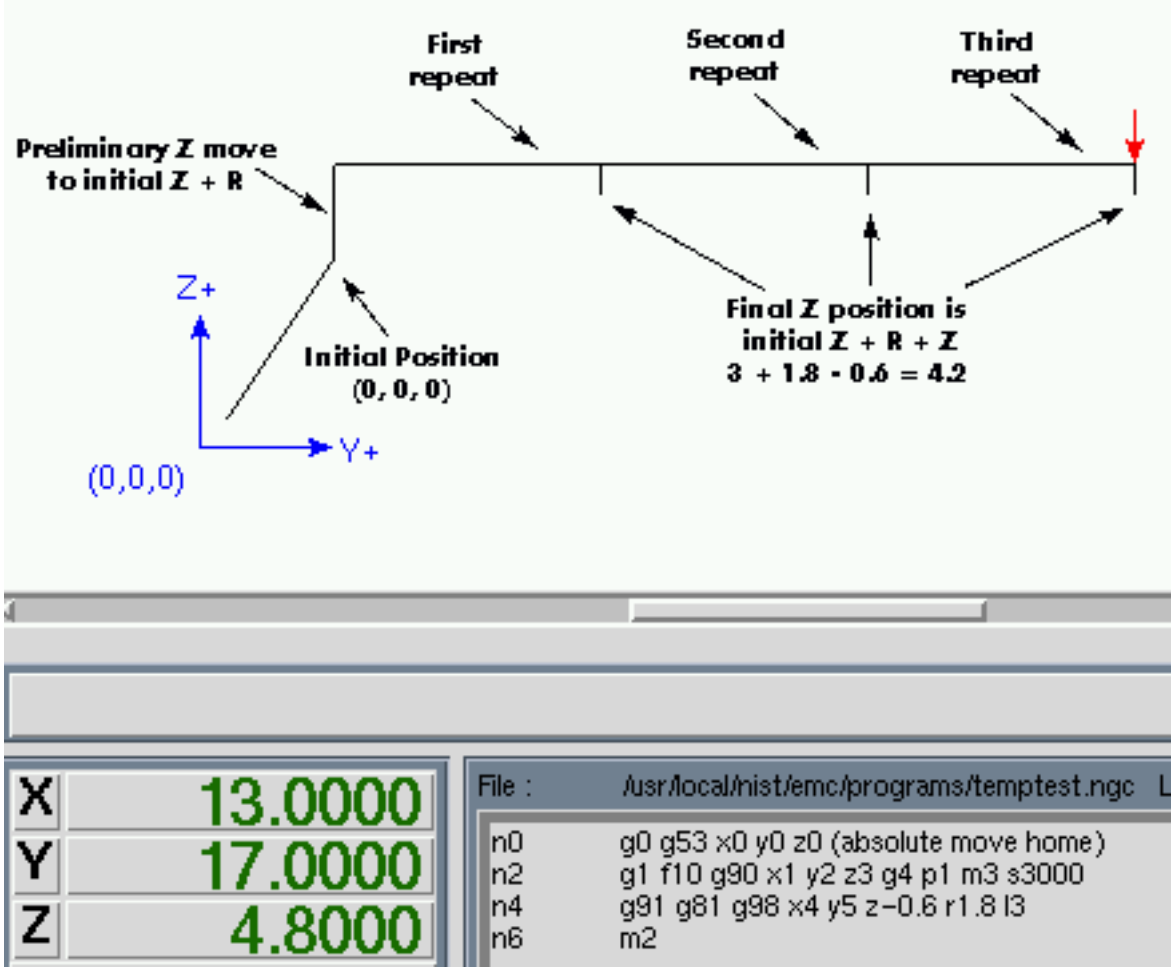
La deuxième répétition produira 3 mouvements. La position de X est augmentée de 4 et passe à 9, la position de Y est augmentée de 5 et passe à 12.

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.

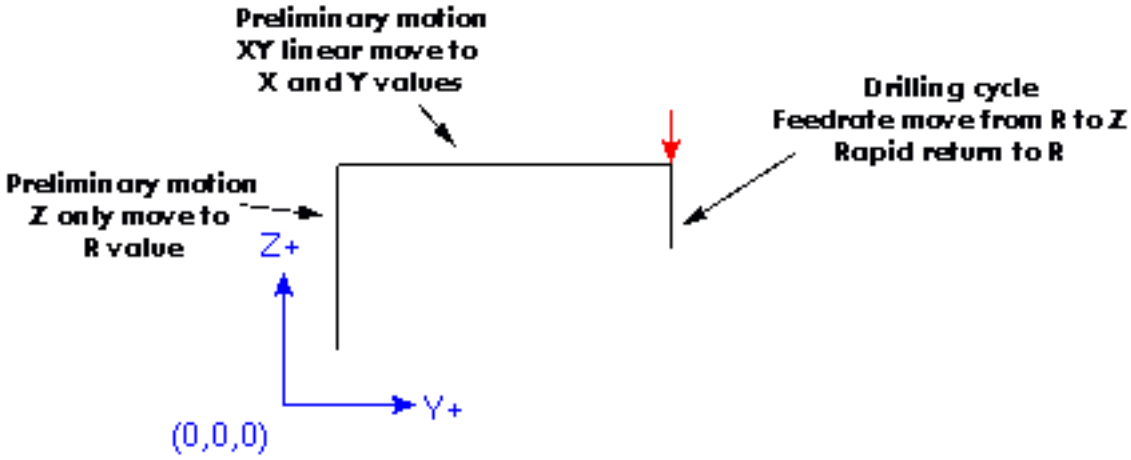
1. Un déplacement en vitesse rapide, parallèle au plan XY vers X13, Y17, Z4.8

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 le premier g81 de la ligne de code soit exécuté, 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.

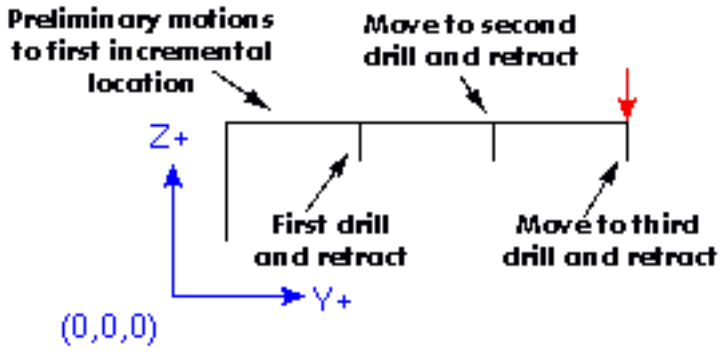


Exemple 4: G81 en absolu avec R > Z

Il s'agit de la trajectoire pour le second bloc 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: G81 en relatif avec R > Z

```
G90 G98 G81 X4 Y5 Z-0.6 R1.8
```

Puisque ce tracé commence en (X0, Y0, Z0), l'interpréteur ajoute R1.8 au Z0 initial et déplace le mobile en vitesse rapide à cet emplacement, comme dans l'exemple 4. Après ce mouvement initial à une hauteur Z0.6, le mouvement en vitesse rapide se terminera en X4 Y5. Alors la hauteur Z sera à 0.6 en dessous de la valeur de R. La fonction de répétition fera encore déplacer Z au même emplacement.

5.3.40 G82 Cycle de perçage avec temporisation

```
G82 (X- Y- Z- ) ou (U- V- W- ) R- L- P-
```

Le cycle G82 est destiné au perçage. 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.

1. Un mouvement préliminaire. Comme décrit sur cette page.
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.

```
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.

5.3.41 G83 Cycle de perçage avec débourrage

```
G83 (X- Y- Z-) or (U- V- W-) R- L- Q-
```

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'aluminium). Ce cycle utilise la valeur Q qui représente un incrément delta le long de l'axe Z.

donnera:

1. Un mouvement préliminaire, comme décrit sur cette page.
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.

5.3.42 G84 Cycle de taraudage à droite

Ce code n'est pas encore implémenté dans LinuxCNC. Il est accepté mais son comportement n'est pas défini. Voir le taraudage rigide.

5.3.43 G85 Cycle d'alésage, sans temporisation, retrait en vitesse travail

```
G85 (X- Y- Z-) or (U- V- W-) R- L-
```

Le cycle G85 est destiné à l'alésage, mais peut être utilisé pour le perçage ou le fraisage.

1. Un mouvement préliminaire, comme décrit sur cette page.
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.

5.3.44 G86 Cycle d'alésage, arrêt de broche, retrait en vitesse rapide

```
G86 (X- Y- Z-) or (U- V- W-) R- L- P-
```

Le cycle G86 est destiné à l'alésage. Ce cycle utilise la valeur P pour une temporisation en secondes.

1. Un mouvement préliminaire, comme décrit sur cette page.
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 exécuté.

5.3.45 G87 Alésage inverse

Ce code n'est pas encore implémenté dans LinuxCNC. Il est accepté mais son comportement n'est pas défini.

5.3.46 G88 Alésage, arrêt de broche, retrait en manuel

Ce code n'est pas encore implémenté dans LinuxCNC. Il est accepté mais son comportement n'est pas défini.

5.3.47 G89 Cycle d'alésage, temporisation, retrait en vitesse travail

```
G89 (X- Y- Z-) or (U- V- W-) R- L- P-
```

Le cycle G89 est destiné à l'alésage. Il utilise la valeur de P pour une temporisation en secondes.

1. Un mouvement préliminaire, comme décrit sur cette page.
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.

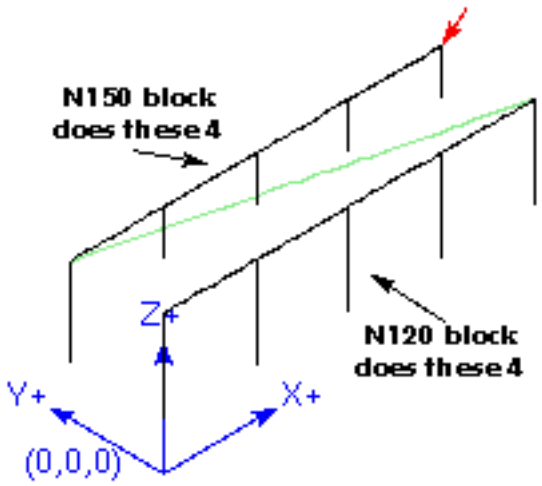
5.3.47.1 Pourquoi utiliser les cycles de perçage ?

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.

Nous avons montré plus haut, comment les cycles de perçage 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 de perçage. 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 de perçage.

Exemple 5: perçage de huit trous, réécrit.

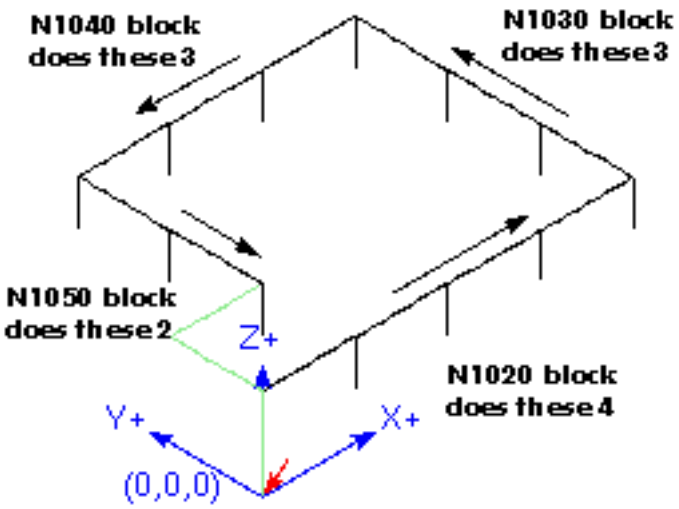
```
G90 G0 X0 Y0 Z0 (coordonnées d'origine)
G1 F10 X0 G4 P0.1
G91 G81 X1 Y0 Z-1 R1 L4 (cycle de perçage)
G90 G0 X0 Y1
Z0
G91 G81 X1 Y0 Z-.5 R1 L4 (cycle de perçage)
G80 (révocation du cycle G81)
M2 (fin de programme)
```



Exemple 6: Douze trous en carré

Cet exemple montre l'utilisation du mot L pour répéter une série incrémentale de cycles de perçage pour des blocs 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.

```
G90 G0 X0 Y0 Z0 (coordonnées d'origine)
G1 F50 X0 G4 P0.1
G91 G81 X1 Y0 Z-0.5 R1 L4 (cycle de perçage)
X0 Y1 R0 L3 (répétition)
X-1 Y0 L3 (répétition)
X0 Y-1 L2 (répétition)
G80 (révocation du cycle G81)
G90 G0 X0 (retour vers l'origine en vitesse rapide)
Y0
Z0
M2 (fin de programme)
```



La deuxième raison d'utiliser les cycles de perçages, 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.

5.3.48 G90, G91: Modes de déplacement

- G90 est le mode de déplacement absolu, les valeurs d'axes X, Y, Z, A, B, C, U, V, W représentent les positions dans le système de coordonnées courant. Les exceptions à cette règle sont décrites dans la section sur les cycles de perçage.
- G91 est le mode de déplacement relatif, en mode relatif, les valeurs d'axes représentent un incrément depuis la position courante.

Exemple avec G90

```
G90 (passe en mode de déplacement absolu)
G0 X2.5 (déplacement linéaire en vitesse rapide à la coordonnée X=2.5 en
incluant tous les offsets en cours)
```

Exemple avec G91

```
G91 (passe en mode de déplacement relatif)
G0 X2.5 (déplacement linéaire en vitesse rapide, à +2.5 en X de la position
courante)
```

- Voir G0 pour plus d'information.

5.3.49 G90.1, G91.1: Mode de déplacement en arc (I, J et K)

- G90.1 - Mode de déplacement absolu pour les offsets I, J et K. Quand G90.1 est actif, I et J doivent être tous les deux spécifiés avec G2/G3 pour le plan XY ou J et K pour le plan XZ, sinon c'est une erreur.
- G91.1 - Mode de déplacement relatif pour les offsets I, J et K. G91.1 replace I, J et K à leur fonctionnement normal.

5.3.50 G92 Décalage d'origine des systèmes de coordonnées

```
G92 axes
```

Voir ce chapitre pour une vision générale des systèmes de coordonnées.

G92 fixera de nouvelles valeurs de coordonnées au point actuel (sans faire de mouvement). Les mots d'axes contiennent les valeurs souhaitées. Au moins un mot d'axe est obligatoire, les autres sont facultatifs. Si il n'y a pas de mot d'axe pour un axe donné, les coordonnées de cet axe resteront inchangées.

Quand G92 est exécuté, les origines de tous les systèmes de coordonnées sont déplacées. Elles seront déplacées de sorte que les valeurs du point contrôlé courant, dans le système de coordonnées courant, deviendront celles spécifiées dans la ligne du G92. Les origines de tous les systèmes de coordonnées sont décalées de la même distance.

Par exemple, supposons que le point courant soit à X=4 et qu'aucun décalage G92 ne soit actif. La ligne G92 X7 est programmée, toutes les origines seront décalées de -3 en X, ce qui fera que le point courant deviendra X=7. Ce -3 est enregistré dans le paramètre 5211.

Être en mode de déplacement relatif est sans effet sur l'action de G92.

Des décalages G92 peuvent déjà être actifs quand G92 est appelé. Si c'est le cas, ils seront remplacés par le nouveau décalage, de sorte que le point courant devienne la valeur spécifiée.

C'est une erreur si:

- Tous les mots d'axes sont omis.

LinuxCNC conserve les décalages G92 et les réutilise au prochain démarrage du logiciel. Pour éviter cela, programmer un G92.1 qui les effacera, ou un G92.2 qui supprimera les valeurs enregistrées.

Note
The G52 command can also be used to change this offset; see the Offsets Section for more details about G92 and G52 and how they interact.

Voir le chapitre sur les systèmes de coordonnées.

Voir la section sur les décalages G92.

Voir la section sur les paramètres.

5.3.51 G92.1, G92.2 Remise à zéro des décalages des systèmes de coordonnées

- G92.1 - Positionne les décalages d'axes à 0 et passe les paramètres 5211 à 5219 à zéro.
- G92.2 - Positionne les décalages d'axes à 0, laisse les valeurs des paramètres inchangées, elles ne seront pas utilisées.

Note
G92.1 only clears G92 offsets, to change G53-G59.3 coordinate system offsets in G-code use either G10 L2 or G10 L20.

5.3.52 G92.3 Restauration des décalages d'axe

- G92.3 - Positionne les décalages d'axes aux valeurs enregistrées dans les paramètres 5211 à 5219.

Il est possible de positionner les décalages d'axes dans un programme puis de ré-utiliser les mêmes dans un autre programme. Pour cela, programmer G92 dans le premier programme, ce qui positionnera les paramètres 5211 à 5219. 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 par le premier.

5.3.53 G93, G94, G95: Choix des modes de vitesse

- G93 - Passe en mode inverse du temps. Dans le mode vitesse inverse du temps, le mot F signifie que le mouvement doit être terminé en [1/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).
- G94 - Passe en mode unités par minute. Dans le mode vitesse en unités par minute, le mot F indique le déplacement du point contrôlé en millimètres par minute, en pouces par minute, ou en degrés par minute, selon l'unité utilisée.
- G95 - Passe en mode unités par tour. Dans le mode vitesse en unités par tour, le mot F donne le déplacement du point contrôlé à effectuer sur l'axe Z, en millimètres par tour de broche ou en pouces, selon l'unité utilisée.

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.

5.3.54 G96, G97: Modes de contrôle de la broche

```
G96 <D-> S- (vitesse de coupe constante)
G97         (mode tr/mn)
```

- D- - Vitesse de broche maximale en tours par minute.
- S- - Vitesse de coupe constante.
- G96 D- S- - Passe à une vitesse de coupe constante de S pieds par minute, si G20 est actif, ou S mètres par minute, si G21 est actif. D- est facultatif.

  Lorsque G96 est utilisé, s'assurer que X0 dans le système de coordonnées en cours (y compris les compensations d'outils) est bien le centre de rotation, sinon LinuxCNC ne donnera pas la vitesse de broche désirée. G96 n'est pas affecté par les mode rayon ou diamètre.
- G97 - Vitesse de coupe en tr/mn.

Exemple avec G96

```
G96 D2500 S250 (passe à une vitesse de coupe constante de 250 m/mn maximum pour
une vitesse de broche maximale de 2500tr/mn).
```

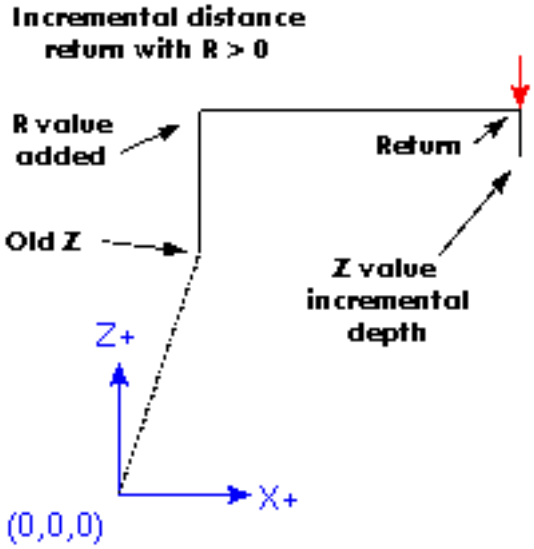
C’est une erreur si:

- S n’est pas spécifié avec G96.
- Une vitesse est spécifiée en mode G96 et la broche ne tourne pas.

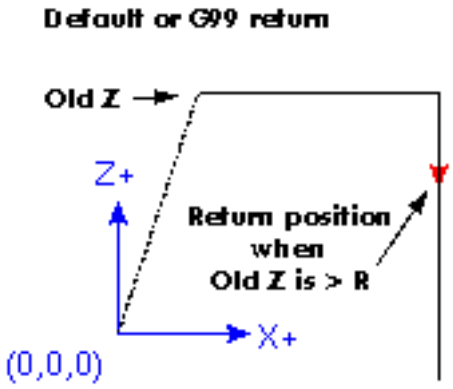
5.3.55 G98, G99: Options du plan de retrait

Quand la broche se rétracte pendant les cycles de perçage, il existe deux options pour indiquer comment elle doit se rétracter:

1. G98 Retrait perpendiculaire au plan de travail courant jusqu’à la position qui était celle de cet axe juste avant le début du cycle de perçage. (à moins que cette position ne soit inférieure à celle indiquée par le mot R, auquel cas, c’est cette dernière qui serait utilisée).



1. G99 Retrait perpendiculaire au plan de travail courant jusqu’à la position indiquée par le mot R.



Ne pas oublier que la signification du mot R change selon que le mode de déplacement est absolu ou relatif.

Le plan de retrait initial (G98) est annulé chaque fois que le mode de mouvement est abandonné, que ce soit explicitement avec G80 ou implicitement (tout code de mouvement qui n'est pas un cycle). Basculer d'un mode de cycle à un autre, par exemple entre G81 et G83 n'annule pas le plan de retrait initial. Il est permis de basculer entre G98 et G99 durant une série de cycles de perçage.

5.4 Les M-codes

5.4.1 Table des M-codes

Code	Description
M0 M1	Pause dans le programme
M2 M30	Fin de programme
M60	Pause pour déchargement pièce
M3 M4 M5	Contrôle de la broche
M6 Tn	Appel d'outil n=numéro d'outil
M7 M8 M9	Contrôle des arrosages
M19	Orientation de la broche
M48 M49	Contrôle des correcteurs de vitesse
M50	Contrôle du correcteur de vitesse travail
M51	Contrôle du correcteur de vitesse de broche
M52	Correcteur dynamique de vitesse d'avance
M53	Contrôle de la coupure de vitesse
M61	Correction du numéro de l'outil courant
M62 à M65	Contrôle de sortie numérique
M66	Contrôle d'entrée numérique et analogique
M67	Contrôle sortie analogique synchronisée
M68	Contrôle sortie analogique directe
M70	Enregistre l'état modal
M71	Invalide l'état modal enregistré
M72	Restaure l'état modal
M73	Enregistrement/auto-restauration de l'état modal
M100 à M199	M-codes définis par l'utilisateur

5.4.2 M0, M1, pause dans le programme

- M0 - Effectue une pause temporaire dans le programme en cours (quelle que soit la position du bouton d'arrêt facultatif). LinuxCNC reste en mode automatique afin que le MDI ou d'autres actions manuelles ne puissent pas être activés. Presser le départ cycle après cette commande relance le programme à la ligne suivante.
- M1 - Stoppe temporairement le programme en cours (mais seulement si le bouton d'arrêt optionnel est activé). LinuxCNC reste en mode automatique afin que le MDI ou d'autres actions manuelles ne puissent pas être activés. Presser le départ cycle après cette commande relance le programme à la ligne suivante.

Note
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.

5.4.3 M2, M30, fin de programme

- M2 - Indique la fin du programme. Presser le départ cycle après cette commande relance le programme au début du fichier.
- M30 - Décharge le porte-pièce du chargeur et termine le programme. Presser le départ cycle après cette commande relance le programme au début du fichier.

Les deux commandes précédentes produisent les effets suivants:

1. Changement du mode automatique au mode MDI.
2. Les décalages d'axes sont mis aux valeurs par défaut (comme avec G54).
3. Le plan de travail actif devient XY (comme avec G17).
4. Le mode de déplacement devient absolu (comme avec G90).
5. La vitesse travail passe en unités par minute (comme avec G94).
6. Les correcteurs de vitesse sont activés (comme avec M48).
7. Les compensations d'outil sont désactivées (comme avec G40).
8. La broche est arrêtée (comme avec M5).
9. Le mode mouvement courant devient G1 (comme avec G1).
10. L'arrosage est arrêté (comme avec M9).

Note
Les lignes de code placées après un M2 ou un M30 ne sont pas exécutées.

5.4.4 M60, pause pour déchargement pièce

- M60 - Procède au changement de porte-pièce avec le chargeur de pièces et effectue une pause dans le programme en cours (quel que soit le réglage du bouton d'arrêt facultatif). Presser ensuite le bouton de départ cycle pour relancer le programme à la ligne suivante.

5.4.5 M3, M4, M5 Contrôle de la broche

- M3 Snnnnn - Démarre la broche en sens horaire à la vitesse **nnnnn**.
- M4 Snnnnn - Démarre la broche en sens anti-horaire à la vitesse **nnnnn**.
- M5 - Arrête la rotation de la broche.

Il est permis d'utiliser M3 ou M4 si la vitesse de broche est à zéro. Si cela est fait (ou si le bouton du correcteur 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.

5.4.6 M6 Appel d'outil

5.4.6.1 Changement d'outil manuel

Si le composant de HAL, hal_manualtoolchange est chargé, M6 va arrêter la broche et inviter l'utilisateur à changer l'outil. Pour plus d'informations sur hal_manualtoolchange voir la section sur le changement manuel d'outil.

5.4.6.2 Changement d'outil

Pour changer l'outil, actuellement dans la broche, par un autre, nouvellement sélectionné en utilisant le mot T, voir la section sur le choix de l'outil, 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 de poche 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.

Warning
La longueur d'outil n'est pas modifié par M6, utilisez un G43 après le M6 pour changer la longueur d'outil.

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.

5.4.7 M7, M8, M9 Contrôle de l'arrosage

- M7 - Active l'arrosage par gouttelettes.
- M8 - Active l'arrosage fluide.
- M9 - Arrête tous les arrosages.

Il est toujours permis d'utiliser une de ces commandes, que les arrosages soient arrêtés ou non.

5.4.8 M19 Orientation de la broche

- M19 R- Q- [P-]
- R - Position à atteindre à partir de 0, cette valeur doit être comprise entre 0 et 360 degrés.
- Q - Durée d'attente en secondes pour compléter l'orientation. Si spindle.N.is-oriented n'est pas devenue vraie dans le temps imparti par Q, une erreur de timeout se produira.
- P - Direction de rotation vers la position cible.
  - 0 - rotation pour petit mouvement angulaire (défaut)
  - 1 - rotation toujours en sens horaire (même direction qu'avec M3)
  - 2 - rotation toujours en sens anti-horaire (même direction qu'avec M4)

M19 est révoqué par M3,M4 ou M5.

L'orientation de la broche nécessite un codeur de position avec index, indiquant la position de la broche ainsi que sa direction de rotation.

Paramètres de réglage de la section [RS274NGC].

ORIENT_OFFSET = 0 à 360 (offset fixe en degrés, ajouté au mot R de M19)

Broches de HAL

- spindle.N.orient-angle (sortie float) Orientation souhaitée pour M19. Valeur du paramètre R de M19 plus la valeur du paramètre d'ini [RS274NGC]ORIENT_OFFSET.

M19 est une commande du groupe modal 7, comme M3, M4 et M5.

- spindle.N.orient-mode (sortie s32) Mode de rotation de la broche souhaité. Reflète le mot P de M19, Défaut = 0
- spindle.N.orient (sortie bit) Indique le début du cycle d'orientation de la broche. Positionné par M19. Remis à zéro par M3,M4 ou M5. Si spindle-orient-fault n'est pas à zéro alors que spindle-orient est vraie la commande M19 échoue avec un message d'erreur.
- spindle.N.is-oriented (entrée bit) Pin de confirmation de l'orientation de la broche. Termine le cycle d'orientation. Si spindle-orient est vraie quand spindle-is-oriented est activée, la pin spindle-orient est mise à zéro et la pin spindle-locked est activée. La pin spindle-brake est également activée.
- spindle.N.orient-fault (entrée s32) Entrée de code d'erreur pour le cycle d'orientation. Toute valeur, autre que zéro, provoquera l'abandon du cycle d'orientation.
- spindle.N.locked (sortie bit) Pin indiquant que le cycle de rotation est terminé. Désactivée par M3,M4 ou M5.

5.4.9 M48, M49 Contrôle des correcteurs de vitesse

- M48 - Autorise les curseurs de corrections de vitesses de broche et celui de vitesse d'avance travail.
- M49 - Inhibe les deux curseurs.

Il est permis d'autoriser ou d'inhiber ces curseurs quand ils sont déjà autorisés ou inhibés. Ils peuvent aussi être activés individuellement en utilisant les commandes M50 et M51, voir ci-dessous.

5.4.10 M50 Contrôle du correcteur de vitesse travail

- M50 <P1> - Autorise le curseur de correction de vitesse d'avance travail. Le paramètre P1 est optionnel.
- M50 P0 - Inhibe le curseur de correction d'avance travail.

Quand il est inhibé, le curseur de correction de vitesse n'a plus aucune influence et les mouvements seront exécutés à la vitesse d'avance travail programmée. (à moins que ne soit actif un correcteur de vitesse adaptative).

5.4.11 M51 Contrôle du correcteur de vitesse broche

- M51 <P1> - Autorise le curseur de correction de vitesse de la broche. Le paramètre P1 est optionnel.
- M51 P0 - Inhibe le curseur de correction de vitesse de broche.

Quand il est inhibé, le curseur 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 sur le réglage de la vitesse de broche.

5.4.12 M52 Contrôle de vitesse adaptative

- M52 P1 - Utilise une vitesse adaptative. Le paramètre P1 est optionnel.
- M52 P0 - Cesse l'utilisation d'une vitesse adaptative.

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 LinuxCNC, la HAL pin motion.adaptive-feed est utilisée dans ce but. Les valeurs de motion.adaptive-feed doivent être dans comprises entre -1 (pleine vitesse arrière) et 1 (pleine vitesse). Une valeur du nulle correspond à l'arrêt du mouvement.

Note
L'utilisation de vitess adaptative négative (destinée notamment aux machines à plasma et à électroérosion) est une nouveauté et n'a pas encore été testée de manière approfondie sur des machines réelles.

5.4.13 M53 Contrôle de la coupure de vitesse

- M53 P1 - Autorise le bouton de coupure de vitesse. Le paramètre P1 est optionnel. Autoriser la coupure de vitesse permet d'interrompre les mouvements par le biais d'une coupure de vitesse. Dans LinuxCNC, la HAL pin motion.feed-hold est utilisée pour cette fonctionnalité. Une valeur de 1 provoque un arrêt des mouvements quand M53 est actif.
- M53 P0 - Inhibe le bouton de coupure de vitesse. L'état de motion.feed-hold est sans effet sur la vitesse quand M53 est inhibé.

5.4.14 M61 Correction du numéro de l'outil courant

- 'M61 Q ' - Corrige le numéro de l'outil courant, en mode MDI ou après un changement manuel d'outil dans la fenêtre de données manuelles. Au démarrage de LinuxCNC avec un outil dans la broche, il est possible ainsi d'ajuster le numéro de l'outil courant sans faire de changement d'outil.

C'est une erreur si:

- Q n'est pas égal où supérieur à 0

5.4.15 M62 à M65 Contrôle de bits de sortie numérique

- M62 P - Active un bit de sortie numérique en synchronisme avec un mouvement.
- M63 P - Désactive un bit de sortie numérique en synchronisme avec un mouvement.
- M64 P - Active immédiatement un bit de sortie numérique.
- M65 P - Désactive immédiatement un bit de sortie numérique.

Le mot P spécifie le numéro du bit de sortie numérique. Le mot P doit être compris entre 0 et une valeur par défaut de 3. Si nécessaire, le nombre des entrées/sorties peut être augmenté en utilisant le paramètre num_dio lors du chargement du contrôleur de mouvement. Voir le manuel de l'intégrateur et section "LinuxCNC et HAL", pour plus d'informations.

Les commandes M62 et M63 seront mises en file d'attente. Toute nouvelle commande, destinée à un bit de sortie écrasera l'ancien réglage de ce bit. Plusieurs bits peuvent changer d'état simultanément par l'envoi de plusieurs commandes M62/M63.

Les nouveaux changements d'état des bits de sortie spécifiés, seront effectifs au début du prochain mouvement commandé. S'il n'y a pas de commande de mouvement ultérieur, les changements en attente n'auront pas lieu. Il est préférable de toujours programmer un G-code de mouvement (G0, G1, etc) juste après les M62/63.

M64 et M65 produisent leur effet immédiatement après être reçus par le contrôleur de mouvement. Ils ne sont pas synchronisés avec un mouvement.

Note
M62 à M66 ne seront opérationnels que si les pins motion.digital-out-nn appropriées sont connectées aux sorties dans le fichier HAL.

5.4.16 M66 Contrôle d'entrée numérique et analogique

```
M66 P- | E- <L-> <Q->
```

- P - Spécifie le numéro d'un bit d'entrée numérique entre 0 et 3.
- E - Spécifie le numéro d'un bit d'entrée analogique entre 0 et 3.
- L - Spécifie le mode d'attente.
  - Mode 0: IMMEDIATE - pas d'attente, retour immédiat, la valeur courante de l'entrée est stockée dans le paramètre #5399
  - Mode 1: RISE attente d'un front montant sur l'entrée.

- Mode 2: FALL attente d'un front descendant sur l'entrée.
- Mode 3: HIGH attente d'un état logique HAUT sur l'entrée.
- Mode 4: LOW attente d'un état logique BAS sur l'entrée.

- Q - Spécifie le timeout pour l'attente, en secondes. Si le timeout est dépassé, l'attente est interrompue et la variable #5399 positionnée à -1.
- Le mode 0 est le seul autorisé pour une entrée analogique.

Exemple de ligne avec M66

```
M66 P0 L3 Q5 (attend jusqu'à 5 secondes la montée de l'entrée numérique 0)
```

- M66 attend un nouvel événement sur une entrée ou la fin de l'exécution du programme, jusqu'à ce que l'événement sélectionné (ou le timeout programmé) ne survienne. 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 numérique).

Si nécessaire, le nombre des entrées/sorties peut être augmenté en utilisant les paramètres num_dio ou num_aio lors du chargement du contrôleur de mouvement. Voir le Manuel de l'intégrateur pour plus d'informations, section des configurations, paragraphes "LinuxCNC et HAL".

Note
M66 ne sera opérationnel que si les pins motion.digital-in-nn ou motion.analog-in-nn appropriées sont connectées aux entrées dans le fichier HAL.

5.4.17 M67 Contrôle de sortie analogique

```
M67 E- Q-
```

- M67 - Contrôle une sortie analogique synchronisée avec un mouvement.
- E - Spécifie le numéro de la sortie, doit être compris entre 0 et 3.
- Q - Spécifie la valeur à appliquer sur la sortie.

Les changements de valeur spécifiés, seront effectifs au début du prochain mouvement commandé. S'il n'y a pas de commande de mouvement ultérieur, les changements en attente n'auront pas lieu. Il est préférable de toujours programmer un G-code de mouvement (G0, G1, etc) juste après les M67. M67 fonctionne comme M62 à M63.

Le nombre d'entrées/sorties peut être augmenté en utilisant le paramètre num_aio au chargement du contrôleur de mouvement. Voir les chapitres "LinuxCNC et HAL" dans la section configuration du Manuel de l'intégrateur pour plus d'informations sur le contrôleur de mouvement.

Note
M67 ne sera opérationnel que si les pins motion.analog-out-nn appropriées sont connectées aux sorties dans le fichier HAL.

5.4.18 M68 Contrôle de sortie analogique directe

```
M68 E- Q-
```

- M68 - Contrôle directement une sortie analogique.
- E - Spécifie le numéro de la sortie, doit être compris entre 0 et 3.
- Q - Spécifie la valeur à appliquer sur la sortie.

M68 produit son effet immédiatement après être reçu par le contrôleur de mouvement. Il n'est pas synchronisé avec un mouvement. M68 fonctionne comme M64 à M65.

Le nombre d'entrées/sorties peut être augmenté en utilisant le paramètre num_aio au chargement du contrôleur de mouvement. Voir le chapitre "LinuxCNC et HAL" dans le Manuel de l'intégrateur pour plus d'informations sur le contrôleur de mouvement.

Note
M68 ne sera opérationnel que si les pins motion.analog-out-nn appropriées sont connectées aux sorties dans le fichier HAL.

5.4.19 M70 Enregistrement de l'état modal

Pour enregistrer explicitement l'état modal au niveau de l'appel courant, programmer M70. Une fois l'état modal enregistré avec M70, il peut être restauré exactement dans le même état en exécutant un M72.

Une paire d'instructions M70 et M72 est typiquement utilisée pour protéger un programme contre d'éventuels changements modaux pouvant se produire dans les sous-programmes.

Les états enregistrés sont les suivants:

- unités machine courantes G20/G21 (po/mm)
- plan de travail courant (G17/G18/G19 G17.1,G18.1,G19.1)
- statut de la compensation de rayon d'outil (G40,G41,G42,G41.1,G42,1)
- mode de déplacement - relatif/absolu (G90/G91)
- mode de vitesse (G93/G94,G95)
- coordonnées système courantes (G54-G59.3)
- statut de la compensation de longueur d'outil (G43,G43.1,G49)
- options du plan de retrait (G98,G99)
- mode de contrôle de broche (G96-css ou G97-RPM)
- mode de déplacement en arc (G90.1, G91.1)
- mode diamètre/rayon des tours (G7,G8)
- mode de contrôle de trajectoire (G61, G61.1, G64)
- avance et vitesse broche courantes (valeurs F et S)
- statut de la broche (M3,M4,M5) - on/off et direction

- statut de l'arrosage (M7) et (M8)
- réglages des correcteurs de vitesse broche (M51) et du correcteur de vitesse travail (M50)
- réglage du contrôle de vitesse adaptative (M52)
- réglage du contrôle de la coupure de vitesse (M53)

Noter qu'en particulier, les modes de mouvement (G1 etc) ne sont PAS restaurés.

Le niveau de l'appel courant signifie:

- Exécution dans le programme principal. Il n'y a qu'un seul emplacement de stockage pour l'état modal au niveau du programme principal; si plusieurs instructions M70 sont exécutées tour à tour, seul l'état enregistré le plus récent est restauré quand un M72 est exécuté.
- Exécution dans un sous-programme G-code. L'état enregistré par M70 dans un sous-programme se comporte exactement comme un paramètre nommé local - on ne peut s'y référer qu'à l'intérieur du sous-programme en invoquant un M72, à la sortie du sous-programme, le paramètre disparaît.

Une invocation récursive d'un sous-programme introduit un nouveau niveau d'appel.

5.4.20 M71 Invalidation de l'état modal enregistré

L'état modal enregistré par M70 ou par M73 au niveau de l'appel courant est invalidé (ne peut plus être restauré nulle part).

Un appel ultérieur à M72 sur le même niveau d'appel, échouera.

Si il est exécuté dans un sous-programme qui protège l'état modal par un M73, un return ou endsub ultérieur ne restaurera PAS l'état modal.

L'utilité de ce dispositif est douteuse. Il ne devrait pas être invoqué quand il peut disparaître.

5.4.21 M72 Restauration de l'état modal

L'état modal enregistré par un M70 peut être restauré en exécutant un M72.

La gestion de G20/G21 reçoit un traitement particulier car les avances sont interprétées différemment selon G20/G21: si les unités de longueur (mm/po) doivent être modifiées par une opération de restauration, M72 va restaurer le mode distance en premier, puis ensuite tous les autres états, y compris les avances pour être sure que les valeurs d'avance soient interprétées selon un réglage d'unités correct.

C'est une erreur d'exécuter M72 sans enregistrement précédent avec M70 à ce niveau.

L'exemple suivant montre l'enregistrement puis la restauration de l'état modal autour de l'appel d'un sous-programme utilisant M70 et M72. Noter que le sous-programme imperialsub n'est pas "au courant" des caractéristiques de M7x et peut être utilisé non modifié:

```
O<showstate> sub
(DEBUG, imperial=#<_imperial> absolute=#<_absolute> feed=#<_feed> rpm=#<_rpm>)
O<showstate> endsub

O<imperialsub> sub
g20 (imperial)
g91 (relative mode)
F5 (low feed)
S300 (low rpm)
(debug, in subroutine, state now:)
o<showstate> call
```

```
O<imperialsub> endsub

; programme principal
g21 (metric)
g90 (absolute)
f200 (fast speed)
S2500 (high rpm)

(debug, in main, state now:)
o<showstate> call

M70 (save caller state in at global level)
O<imperialsub> call
M72 (explicitly restore state)

(debug, back in main, state now:)
o<showstate> call
m2
```

5.4.22 M73 Enregistrement et auto-restauration de l'état modal

Pour enregistrer l'état modal à l'intérieur d'un sous-programme et restaurer cet état lors d'un endsub ou autre return, programmer M73.

En cas d'abandon d'un programme en cours d'exécution dans un sous-programme traitant un M73, l'état ne sera **PAS** restauré.

En outre, la fin normale (M2) d'un programme principal contenant un M73 ne restaurera **pas** l'état.

L'utilisation suggérée consiste à placer au début d'un sous-programme, un O-code de sous-programme comme dans l'exemple ci-dessous. En utilisant M73, cette manière valide le design des sous-programmes qui doivent modifier l'état modal mais qui protège le programme appelant contre tout changement inopiné de l'état modal. Noter l'usage de paramètres nommés dans le sous-programme showstate.

```
O<showstate> sub
(DEBUG, imperial=#<_imperial> absolute=#<_absolute> feed=#<_feed> rpm=#<_rpm>)
O<showstate> endsub

O<imperialsub> sub
M73 (save caller state in current call context, restore on return or endsub)
g20 (imperial)
g91 (relative mode)
F5 (low feed)
S300 (low rpm)
(debug, in subroutine, state now:)
o<showstate> call

; note - M72 n'est pas utilisé ici - le endsub suivant ou un
; 'return' explicite restaurera l'état de l'appelant
O<imperialsub> endsub

; programme principal
g21 (metric)
g90 (absolute)
f200 (fast speed)
S2500 (high rpm)
(debug, in main, state now:)
o<showstate> call
o<imperialsub> call
(debug, back in main, state now:)
```

```
o<showstate> call
m2
```

5.4.23 Restauration sélective de l'état modal par le test de paramètres prédéfinis

Exécuter un M72 ou au retour d'un sous-programme contenant un 'M73_ pour restaurer tout l'état modal enregistré.

Si seulement certains aspects de l'état modal doivent être préservés, une alternative consiste a utiliser les paramètres nommés prédéfinis, paramètres locaux et états conditionnels. L'idée est de rappeler les modes à restaurer au début du sous-programme et de restaurer ceux-ci avant de quitter. Voici un exemple, basé sur le programme nc_files/tool-length-probe.ngc:

```
O<measure> sub   (measure reference tool)
;
#<absolute> = #<_absolute>  (remember in local variable if G90 was set)
;
g30 (above switch)
g38.2 z0 f15 (measure)
g91 g0z.2 (off the switch)
#1000=#5063 (save reference tool length)
(print,reference length is #1000)
;
O<restore_abs> if [#<absolute>]
    g90 (restore G90 only if it was set on entry:)
O<restore_abs> endif
;
O<measure> endsub
```

5.4.24 M100 à M199 Commandes définies par l'utilisateur

```
M1-- <P- Q->
```

- M1 -- - Un entier compris entre 100 et 199.
- P - Un nombre passé comme premier argument au programme externe.
- Q - Un nombre passé comme second argument au programme externe.

Le programme externe, nommé M100 à M199, (avec un M majuscule et aucune extension) qui doit se trouver dans le répertoire pointé par la variable [DISPLAY] PROGRAM_PREFIX du fichier ini, sera exécuté avec les valeurs P- et Q- comme étant ses deux arguments. L'exécution du fichier G-code courant passera en pause jusqu'à ce que le programme invoqué soit terminé. Tout fichier exécutable valide peut être utilisé. Le fichier doit se trouver dans le chemin spécifié dans le fichier ini de configuration. Voir la section sur le fichier de configuration dans le manuel de l'intégrateur.

Après la création d'un nouveau programme M1nn, l'interface graphique doit être redémarrée pour que le nouveau programme soit pris en compte, autrement une erreur M-code inconnu surviendra.

Warning
Ne pas utiliser un traitement de texte pour créer ou éditer ces fichiers. Un traitement de texte ajoute des caractères invisibles qui causent des problèmes et empêchent les scripts bash ou Python de fonctionner. Pour ces raisons, utiliser un éditeur de texte tel que Gedit dans Ubuntu ou le Notepad++ dans un autre OS.

Le message d'erreur M-code inconnu signifie que:

- La commande utilisateur spécifiée n'existe pas.
- Le fichier n'a pas été rendu exécutable.
- Le nom du fichier comporte une extension.
- Le nom du fichier ne suis pas le format suivant: M1nn où nn = 00 à 99.
- Le nom de fichier utilise un m minuscule.

Exemple d'utilisation, dans un programme G-code, on doit ouvrir et fermer un mandrin automatique via une broche du port parallèle, on appellera respectivement M101 pour ouvrir le mandrin et M102 pour le fermer. Les deux scripts bash correspondants, appelés M101 et M102 seront créés avant le lancement de LinuxCNC puis rendus exécutables, par exemple par un clic droit puis propriétés → permissions → Exécution. S'assurer que cette broche du port parallèle n'est pas déjà utilisée dans un fichier de HAL.

Exemple de fichier pour M101

```
#!/bin/bash
# ce fichier met la broche 14 du port à 1 pour ouvrir le mandrin automatique
halcmd setp parport.0.pin-14-out True
exit 0
```

Exemple de fichier pour M102

```
#!/bin/bash
# ce fichier met la broche 14 du port à 0 pour fermer le mandrin automatique
halcmd setp parport.0.pin-14-out False
exit 0
```

Pour passer des variables à un fichier M1nn, utiliser les mots facultatifs P et Q de cette façon:

```
M100 P123.456 Q321.654
```

Exemple pour M100

```
#!/bin/bash
tension=$1
vitesse=$2
halcmd setp thc.voltage $tension
halcmd setp thc.feedrate $vitesse
exit 0
```

Pour ouvrir un message graphique et passer en pause jusqu'à ce que la fenêtre du message soit fermée, utiliser un programme comme Eye of Gnome pour afficher le fichier graphique. Quand la fenêtre sera fermée, le programme reprendra.

Exemple pour M110, affichage d'un graphique avec passage en pause

```
#!/bin/bash
eog /home/robert/linuxcnc/nc_files/message.png
exit 0
```

Pour afficher un message graphique en continuant le traitement du fichier G-code, ajouter un caractère esperluette à la commande.

Exemple pour M110, affichage d'un graphique sans passer en pause

```
#!/bin/bash
eog /home/robert/linuxcnc/nc_files/message.png &
exit 0
```

5.5 Les O-codes

5.5.1 Utilisation des O-codes

Les O-codes permettent le contrôle de flux dans les programmes NGC. Ils commencent par une lettre **O**, qu'il ne faut pas confondre avec le chiffre **0**. Chaque bloc est associé à une adresse, qui est la valeur utilisée après la lettre **O**. Il faut prendre soin de bien faire correspondre les adresses des O-codes.

Exemple de numérotation

```
o100 sub
(noter que les blocs if - endif utilisent des numéros différents)
  o110 if [#2 GT 5]
    (du code ici)
  o110 endif
  (encore du code ici)
o100 endsub
```

Le comportement est indéfini si:

- Le même nombre est utilisé pour plusieurs blocs
- D'autres mots sont utilisés sur une ligne contenant un mot O-.
- Un commentaire est utilisé sur une ligne contenant un mot O-.

Tip
L'utilisation de la lettre **o** minuscule facilite la distinction avec le chiffre **0** qui peut être tapé par erreur. Par exemple:
o100 est plus facile à distinguer de **0100** que **O100**.

5.5.2 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.

Exemple de sous-programme

```
O100 sub (sous-programme de mouvement rapide à l'origine)
  G53 X0 Y0 Z0
O100 endsub
  (autres lignes)
O100 call (ici, appel du sous-programme)
M2
```

Pour plus de détails sur ces instructions voir:

- mouvement G53,
- mouvement rapide G0,
- fin de programme M2.

O- return 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.

Exemple avec O- return

```
o100 sub
  o110 if [#2 GT 5] (teste si le paramètre #2 est supérieur à 5)
    o100 return (si le test est vrai, retourne au début du sous-programme)
  o110 endif
  (autre code ici, qui sera exécuté si le paramètre #2 est inférieur à 5)
o100 endsub
```

Voir également les sections:

- les opérateurs binaires,
- les paramètres.

O- call 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:

Exemple d'appel O-

```
O200 call [1] [2] [3]
```

Les corps de sous-programme ne peuvent pas être imbriqués. Ils ne peuvent être appelés qu'après avoir 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.

5.5.3 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. La différence se trouve en fin de test de la condition. La boucle do - while exécute le code dans la boucle puis test la condition. La boucle while - endwhile effectue le test d'abord.

Exemple avec while - endwhile

```
(dessine la forme d'une dent de scie)
G0 X1 Y0 (déplacement en position de départ)
#1 = 1 (assigne la valeur 1 au paramètre #1)
F25 (fixe la vitesse d'avance travail)
o101 while [#1 LT 10]
  G1 X0
  G1 Y[#1/10] X1
  #1 = [#1+1] (incrémente le compteur de test)
o101 endwhile
M2 (fin de programme)
```

Exemple avec do - while

```
#1 = 0 (assigne la valeur 0 au paramètre #1)
o100 do
  (debug, paramètre 1 = #1)
  o110 if [#1 EQ 2]
```

```
    #1 = 3 (assigne la valeur 3 au paramètre #1)
    (msg, #1 s'est vu assigné la valeur 3)
    o100 continue (saute au début de la boucle)
  o110 endif
  (le code d'usinage ici)
  #1 = [#1 + 1] (incrémente le compteur de test)
o100 while [#1 LT 3]
(msg, boucle terminée)
M2
```

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.

5.5.4 Conditionnel: if, elseif, else, endif

Le if conditionnel exécute un groupe d'instructions avec le même nombre O qui commence avec if et se termine avec endif. Les conditions optionnelles elseif et else peuvent se trouver entre le if et le endif.

Si la condition du if est vraie, les instructions qui suivent le if seront exécutées jusqu'à, au maximum, l'instruction conditionnelle suivante.

Si la condition du if est fausse, alors les instructions conditionnelles elseif suivantes seront évaluées l'une après l'autre. Si la condition du elseif est vraie alors les instructions suivant ce elseif seront exécutées jusqu'à l'instruction conditionnelle suivante. Si aucune des conditions du if ou du elseif n'est vraie, alors les instructions suivant le else seront exécutées. Quand une condition est vraie, les autres instructions conditionnelles du groupe ne sont plus évaluées.

Exemple avec if - endif

```
O102 if [#31 EQ 3] (si le paramètre #31 est égal à 3 alors S2000)
    S2000
O102 endif
```

Exemple avec if - elseif - else - endif

```
o102 if [#2 GT 5] (si le paramètre #2 est supérieur à 5 alors F100)
  F100
o102 elseif [#2 LT 2] (sinon si le paramètre #2 est inférieur à 2 alors F200)
  F200
o102 else (sinon le paramètre #2 vaut entre 2 et 5 alors F150)
  F150
o102 endif
```

5.5.5 Répétition: Repeat

La répétition repeat, exécutera les blocs contenus entre repeat et endrepeat le nombre de fois spécifié entre crochets. L'exemple suivant montre comment usiner une séries de 5 formes diagonales commençant à la position courante.

Exemple avec repeat

```
(Usine 5 formes diagonales)
G91 (Mode incrémental)
O103 repeat [5]
    (insérer le code d'usinage ici)
    G0 X1 Y1 (Mouvement en diagonale vers la position suivante)
O103 endrepeat
G90 (Mode absolu)
```

5.5.6 Indirection

L'adresse de O- peut être donnée par un paramètre ou un calcul.

Exemple d'indirection

```
O[#101+2] call
```

Calcul des valeurs dans les O-codes Voici un condensé des sections utiles aux calculs des O-codes:

- les paramètres,
- les expressions,
- les opérateurs binaires,
- les fonctions.

5.5.7 Appel de fichier

Pour appeler un sous-programme par son nom, ce sous-programme doit contenir un sub et un endsub. Le fichier appelé doit se trouver dans le répertoire pointé par la variable PROGRAM_PREFIX ou SUBROUTINE_PATH du fichier ini. Les noms de fichiers ne peuvent inclure que des lettres **minuscules**, des chiffres, des points et des tirets bas. Un fichier de sous-programme nommé ne peut contenir qu'une seule définition de sous-programme.

Exemple: l'appel d'un fichier nommé

```
o<monfichier> call (appel un fichier nommé)
```

Exemple: l'appel d'un fichier numéroté

```
o123 call (appel un fichier numéroté)
```

Dans le fichier appelé doit se touver le sub et le endsub correspondant à l'appel. Le fichier doit être un fichier valide.

Exemple: le fichier monfichier.ngc appelé

```
o<monfichier> sub
  (du code ici)
o<monfichier> endsub
M2
```

Note
Les noms de fichiers doivent être en lettres minuscules, ainsi o<MonFichier> sera transformé en o<monfichier> par l'interpréteur.

5.6 Les autres codes

5.6.1 F: Réglage de la vitesse d'avance travail

Pour régler la vitesse d'avance, programmer F-. L'application de la vitesse est telle que décrite dans l'aperçu global d'une machine numérique, section vitesse d'avance, à moins que le mode vitesse inverse du temps ne soit activé, dans ce cas, la vitesse est telle que décrite dans la section sur le choix des modes de vitesse.

5.6.2 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 sur le cycle de taraudage à droite, si un cycle de perçage 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%.

5.6.3 T: Choix de l'outil

Pour sélectionner un outil, programmer T-, où la valeur de T correspond au numéro de la poche d'outil dans le carrousel. L'outil ne sera appelé et changé que quand un M6 sera programmé voir la section sur l'appel d'outil. 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 poches 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 Tn, 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.

Les mouvements rapides qui suivent un T<n> n'apparaissent pas sur l'écran de parcours d'outil d'Axis, et ce jusqu'au prochain mouvement en vitesse travail Cela se remarque surtout sur les machines ayant de longues distances de déplacement lors du changement d'outil, comme les tours. Cela peut prêter à confusion au début. Pour contourner ce dysfonctionnement pour l'outil courant, ajouter un G1 sans mouvement juste après le T<n>.

5.7 Exemples de fichiers G-Code

Après l'installation de LinuxCNC, plusieurs exemples de fichiers G-code se trouveront dans le répertoire /nc_files du dossier d'installation. Seuls les fichiers adaptés au type de la machine sont utilisable.

5.7.1 Exemples pour une fraiseuse

5.7.1.1 Fraisage hélicoïdal d'un orifice

- Nom du fichier: useful-subroutines.ngc
- Description: Programme pour usiner une poche ou un alésage avec utilisation des paramètres.

5.7.1.2 Rainurage

- Nom du fichier: useful-subroutines.ngc
- Decription: Programme pour fraiser une rainure avec utilisation des paramètres.

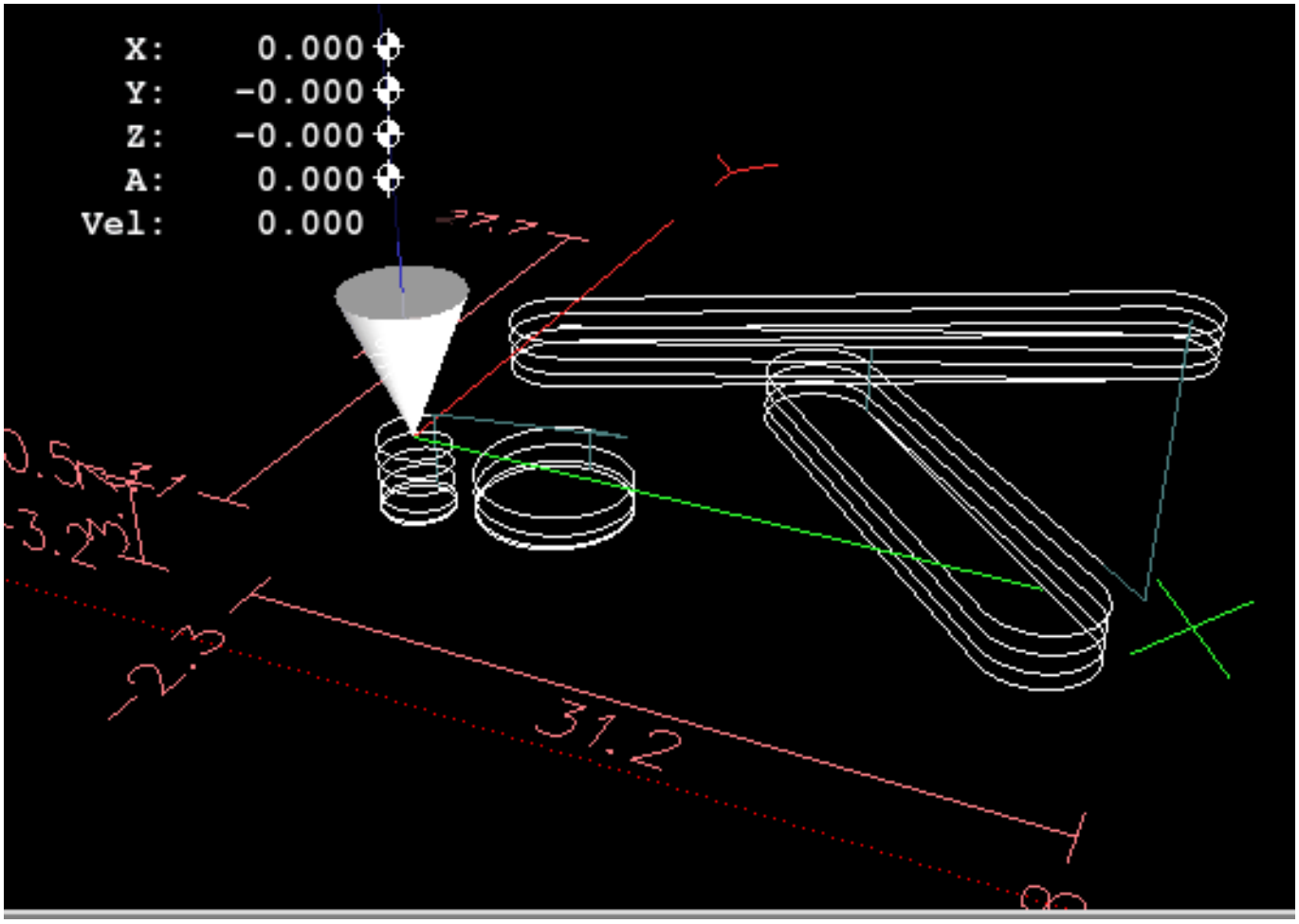


Figure 5.6: Les différents usinages de l'exemple

5.7.1.3 Palpage d'une grille rectangulaire de points

- Nom du fichier: gridprobe.ngc
- Description: Relevé d'une grille rectangulaire de points au palpeur.

Ce programme palpe de manière répétitive, en suivant une grille régulière en XY et écrit les points mesurés dans le fichier probe-results.txt situé dans le même répertoire que le fichier .ini.

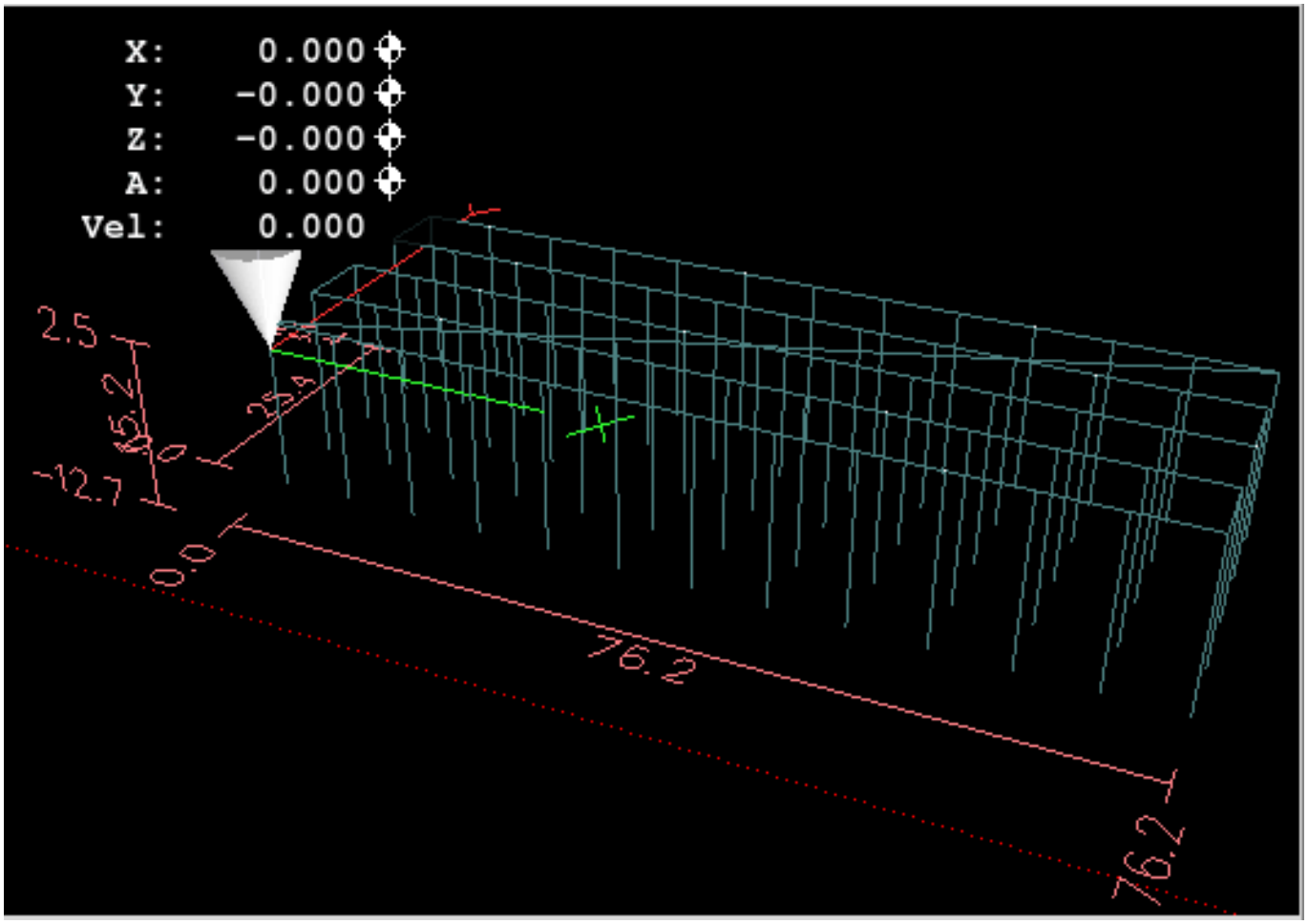


Figure 5.7: La grille de palpage

5.7.1.4 Amélioration du palpage d'une grille rectangulaire de points

- Nom du fichier: smartprobe.ngc
- Description: Relevé d'une grille rectangulaire de points au palpeur.

Ce programme est une amélioration du précédent. Il palpe de manière répétitive, en suivant une grille régulière en XY et écrit les points mesurés dans le fichier probe-results.txt situé dans le même répertoire que le fichier .ini.

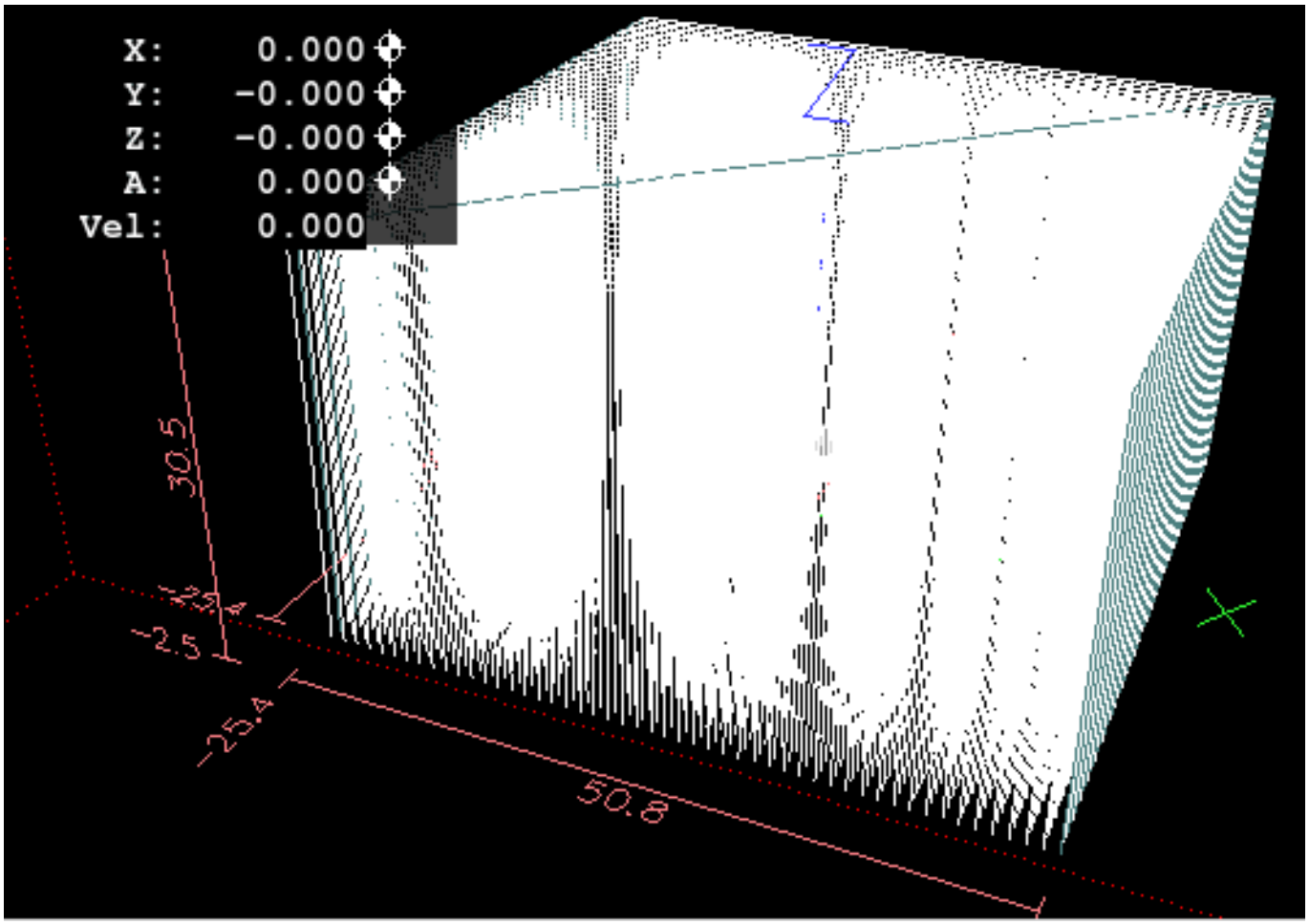


Figure 5.8: La grille de palpage plus fine

5.7.1.5 Mesure de longueur d'outil

- Nom du fichier: tool-lenght-probe.ngc
- Description: Mesure automatique de la longueur de l'outil.

Ce programme donne un exemple de la mesure automatique de longueur d'outil en utilisant un contact raccordé à l'entrée sonde. C'est très pratique pour une machine sur laquelle la longueur des outils est différente à chaque montage.

5.7.1.6 Mesure d'un alésage au palpeur

- Nom du fichier: probe-hole.ngc
- Description: Mesure le centre et le diamètre d'un alésage.

Ce programme montre comment trouver le centre d'un alésage, comment calculer son diamètre et enregistrer les mesures dans un fichier.

5.7.1.7 Compensation de rayon d'outil

- Nom du fichier: comp-g1.ngc
- Description: Mouvements d'entrée et de sortie avec la compensation de rayon d'outil.

Ce programme démontre la particularité du chemin d'outil sans et avec la compensation de rayon d'outil. Le rayon d'outil est pris dans la table d'outils.

5.7.2 Exemples pour un tour

5.7.2.1 Filetage

- Nom du fichier: lathe-g76.ngc
- Description: Dressage, filetage et tronçonnage.

Ce programme donne un exemple de filetage automatique sur un tour avec utilisation des paramètres. Il demande quelques adaptations pour fonctionner, ces adaptations seront un excellent exercice.

5.8 Différences avec RS274/NGC

5.8.1 Changements entre RS274/NGC et LinuxCNC

5.8.1.1 Position après un changement d'outil

Avec LinuxCNC, le mobile ne retourne pas sur la position de départ après un changement d'outil. Ce mode de fonctionnement est nécessaire, car un outil peut être plus long que l'outil précédent et dans ce cas un mouvement sur la position précédente placerait l'outil trop bas.

5.8.1.2 Les paramètres de décalage sont dans l'unité du fichier ini

Dans LinuxCNC, les valeurs mémorisées dans les paramètres pour les positions d'origine des commandes G28 et G30, les systèmes de coordonnées P1 à P9 et le décalage G92 sont dans l'unité du fichier ini. Ce changement a été fait car la position d'un point change selon que G20 ou G21 était actif lors de la programmation d'un G28, G30, G10 L2 ou G92.3.

5.8.1.3 Table d'outils, longueur et diamètre sont dans l'unité du fichier ini

Dans LinuxCNC, les longueurs d'outil (compensation) et les diamètres sont spécifiés seulement dans l'unité du fichier ini. Cela est nécessaire, car la longueur et le diamètre de l'outil changent selon que G20 ou G21 étaient actifs à l'initialisation des modes G43, G41, G42. Il était donc impossible de lancer un G-code avec des unités machines non natives, ceci même lorsque le G-code est simple et bien formé (débutant par G20 ou G21 et sans changement d'unité tout au long du programme) sans changer la table d'outils.

5.8.1.4 G84, G87 ne sont pas implémentés

G84 et G87 ne sont pour le moment pas implémentés. Ils le seront dans une version futur de LinuxCNC.

5.8.1.5 G28, G30 avec des mots d'axe

Lorsqu'un G28 ou un G30 est programmé avec seulement quelques mots d'axe présents, LinuxCNC déplace seulement les axes nommés. Ce comportement est commun aux contrôleurs de machine. Pour déplacer certains axes à un point intermédiaire, puis déplacer tous les axes à un point prédéfini, écrire deux lignes de G-code:

```
G0 X- Y-   (déplace les axes au point intermédiaire)
G28        (déplace tous les axes au point prédéfini)
```

5.8.2 Ajouts à RS274/NGC

Différences qui ne changent pas le déroulement des programmes en RS274/NGC.

5.8.2.1 Codes de filetage G33 et G76

Ces codes ne sont pas définis dans RS274/NGC.

5.8.2.2 G38.2

La pointe de touche n'est pas rétractée après un mouvement G38.2. Ce mouvement de retrait sera ajouté dans une version futur de LinuxCNC.

5.8.2.3 G38.3 à G38.5

Ces codes ne sont pas définis dans RS274/NGC.

5.8.2.4 Les O-codes

Ces codes ne sont pas définis dans RS274/NGC

5.8.2.5 M50 à M53 Correcteurs de vitesse

Ces codes ne sont pas définis dans RS274/NGC.

5.8.2.6 M61 à M66

Ces codes ne sont pas définis dans RS274/NGC.

5.8.2.7 G43, G43.1

Longueurs d'outil négatives
: Les spécifications RS274/NGC précisent "il est prévu que" toutes les longueurs d'outils soient positives. Cependant, G43 fonctionne avec des longueurs d'outil négatives.

Outils de tournage
: La compensation de longueur d'outil G43 peut compenser l'outil à la fois en X et en Z. Cette fonctionnalité est surtout utile sur les tours.

Longueurs d'Outil dynamiques
: LinuxCNC permet la spécification d'une longueur d'outil calculée par G43.1 I K.

5.8.2.8 G41.1, G42.1 Compensation dynamique

LinuxCNC permet dans le G-code, la spécification d'un diamètre d'outil et en mode tour, l'orientation est également spécifiée. 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 tournage.

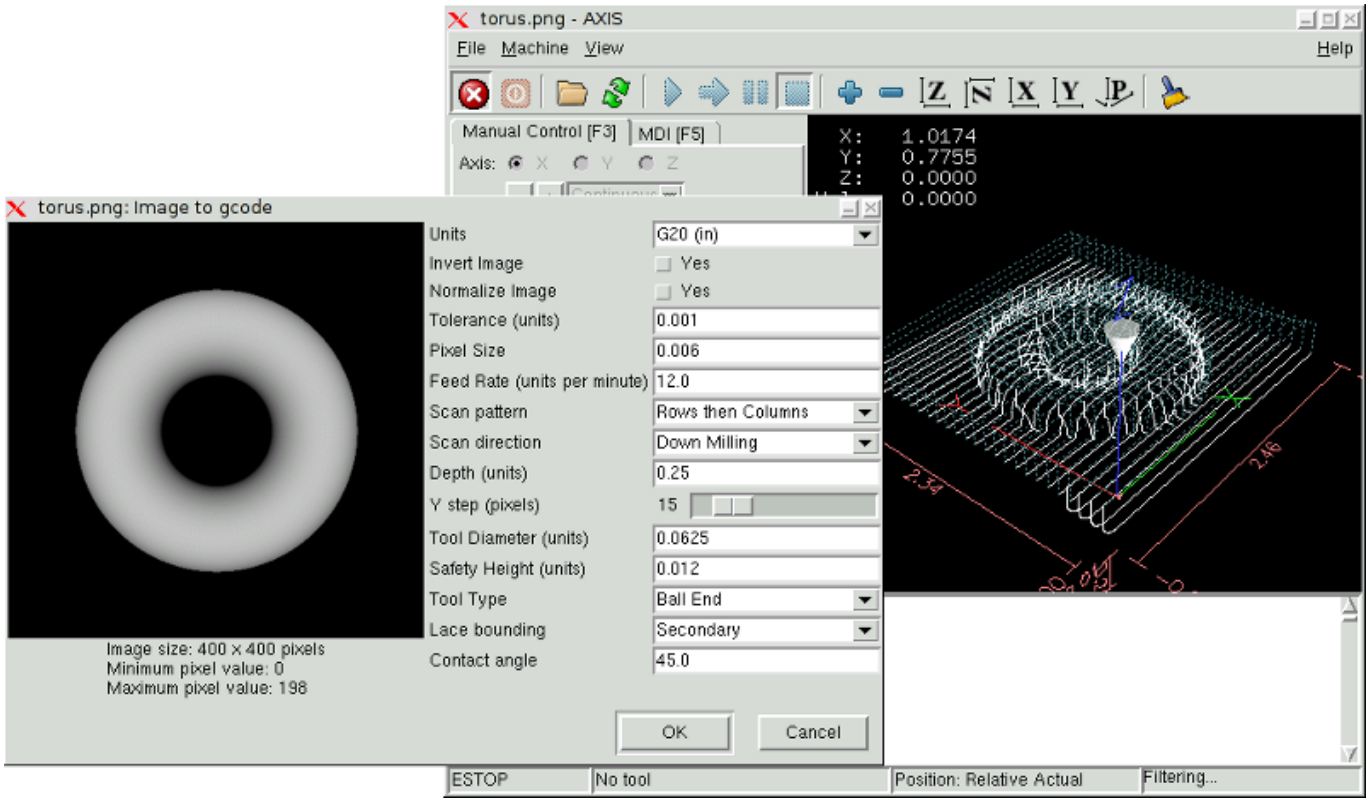
5.8.2.9 G43 sans le mot H

Ce code n'est pas permis en NGC. Dans LinuxCNC, il fixe la compensation de longueur pour l'outil actuellement chargé. Si aucun outil n'est actuellement chargé, c'est une erreur. Ceci a été fait afin que l'utilisateur n'ait pas à spécifier, pour chaque changement d'outil, le numéro d'outil à deux endroits et c'est cohérent avec la manière de fonctionner de G41/G42 quand le mot D n'est pas spécifié.

5.8.2.10 U, V et W axes

LinuxCNC peut admettre des machines ayant jusqu'à 9 axes en définissant un ensemble supplémentaire de 3 axes linéaires, connus comme U, V et W.

5.9 Image-to-gcode: Usiner un depth maps



5.9.1 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.

5.9.2 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
```

Le fichier de configuration: sim/axis.ini est déjà configuré de cette façon.

5.9.3 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 G-code 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.

5.9.4 Les différentes options

5.9.4.1 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).

5.9.4.2 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.

5.9.4.3 Normalize Image

Si yes, le pixel le plus sombre est ramené au noir, le pixel le plus lumineux est ramené au blanc.

5.9.4.4 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.

5.9.4.5 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 LinuxCNC, mais peut aussi estomper ou gommer les détails les plus fins de l'image.

5.9.4.6 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.

5.9.4.7 Plunge Feed Rate (unités par minute)

Vitesse du mouvement de plongée initial.

5.9.4.8 Feed Rate (unités par minute)

Vitesse d'avance pour le reste de l'usinage.

5.9.4.9 Spindle Speed (RPM)

Vitesse de rotation de la broche, en tours/mn

5.9.4.10 Scan Pattern

Modèles de balayage possibles:

- Rangées
- Colonnes
- Rangées puis colonnes
- Colonnes puis rangées

5.9.4.11 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.

5.9.4.12 Depth (unités)

Le dessus du bloc est toujours à Z=0. La profondeur d'usinage dans le matériau est de Z=-depth.

5.9.4.13 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'.'

5.9.4.14 Tool Diameter

Le diamètre du taillant de l'outil.

5.9.4.15 Safety Height

La hauteur à laquelle les mouvements de traversée. image-to-gcode considère toujours le dessus du matériau comme étant: Z=0.

5.9.4.16 Tool Type

La forme du taillant de l'outil. Les formes possibles sont:

- Hémisphèrique
- Plate
- Vé à 45 degrés
- Vé à 60 degrés

5.9.4.17 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.

5.9.4.18 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.

5.9.4.19 Offset d’ébauche et profondeur par passe d’ébauche

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 ci-dessous 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.

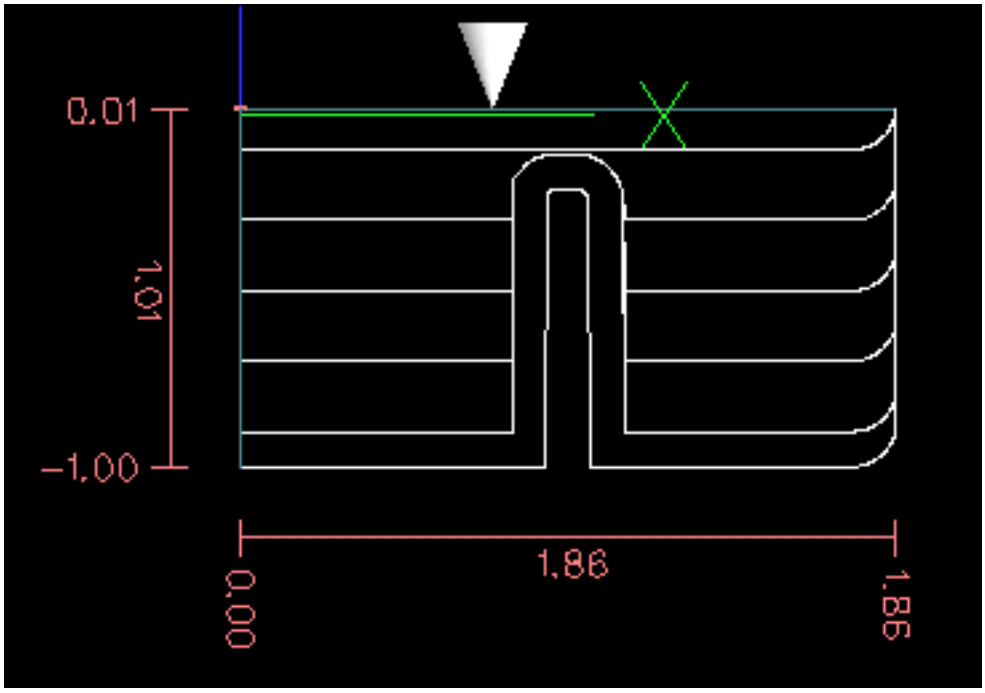


Figure 5.9: Passes d’ébauche

Chapter 6

Tool Compensation

6.1 Les compensations d'outil

6.1.1 Compensation de longueur d'outil

6.1.1.1 Toucher

Dans la boîte de dialogue du bouton Toucher de l'interface AXIS, il est possible de mettre à jour automatiquement la table d'outils.

Séquence typique pour mise à jour de la table d'outils:

- Après la prise d'origine, charger un outil Tn M6 dans lequel n est le numéro de l'outil.
- Déplacer l'outil pour établir le zéro pièce, en utilisant une cale d'épaisseur ou en faisant une petite passe puis une mesure.
- Cliquer sur le bouton Toucher de l'onglet Controle manuel (ou presser la touche Fin du clavier).
- Sélectionner Table d'outils dans la liste déroulante des systèmes de coordonnées.
- Entrer l'épaisseur de la cale ou la cote mesurée.
- Presser OK.

La table d'outil sera alors modifiée avec la longueur correcte en Z de l'outil. La visu affichera la position en Z correcte et une commande G43 sera passée pour que la nouvelle longueur Z de l'outil soit effective. Le choix Table d'outils n'apparaitra dans la liste déroulante du Toucher, que si l'outil à été chargé avec Tn M6.

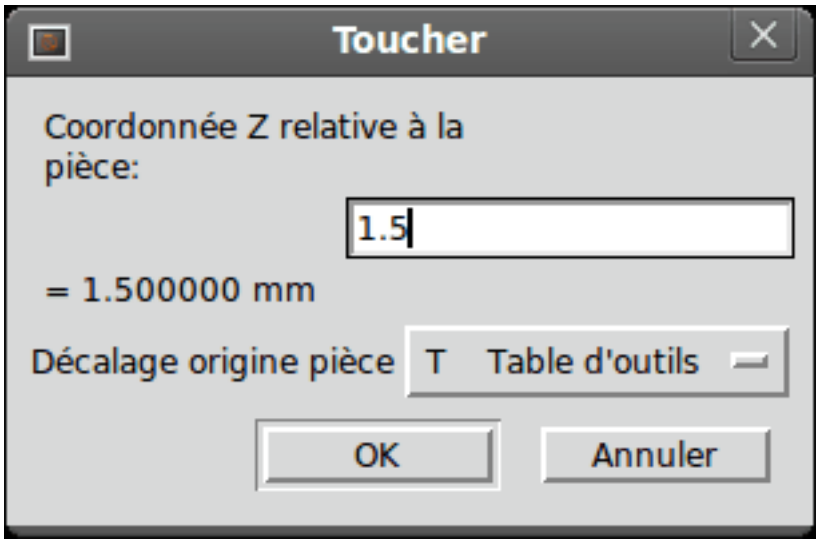


Figure 6.1: Toucher et table d'outils

6.1.1.2 Utilisation de G10 L1/L10/L11

Les commandes G10 L1/L10/L11 peuvent être utilisées pour ajuster les compensations dans la table d'outils: (Ce sera juste une brève présentation, se reporter au guide de références du G-code pour des explications plus détaillées)

G10 L1 Pn - (n est le N° d'outil) Fixe les offsets de l'outil. La position courante n'est pas significative. Tous les détails ici.

G10 L10 Pn - (n est le N° d'outil) Fixe l'offset à la position courante, met les valeurs dans un système de 1 à 8. Tous les détails ici.

G10 L11 Pn - (n est le N° d'outil) Fixe l'offset à la position courante, met les valeurs dans le système 9. Tous les détails ici.

6.1.2 Table d'outils

La table d'outils est un fichier texte qui contient les informations de chaque outil. Ce fichier est placé dans le même répertoire que le fichier de configuration. Il est nommé tool.tbl. Les outils peuvent être dans un changeur d'outils ou changés manuellement. Le fichier peut être édité avec un éditeur de texte ou être mis à jour avec la commande G10 L1. Voir la section spécifique au tour pour ce qui concerne les outils de tour, avec un exemple de table. Le nombre d'outils est limité à 1000 dans une table d'outils même si la numérotation des outils et des poches peut aller jusqu'à 99999.

6.1.2.1 Format de la table d'outils

Table 6.1: Format de la table d'outils

T#	P#	X	Y	Z	A	B	C	U	V	W	Dia	FA	BA	Ori	Rem
;	(aucune donnée après le point-virgule)														
T1	P17	X0	Y0	Z0	A0	B0	C0	U0	V0	W0	D0	I0	J0	Q0	;rem
T2	P5	X0	Y0	Z0	A0	B0	C0	U0	V0	W0	D0	I0	J0	Q0	;rem
T3	P12	X0	Y0	Z0	A0	B0	C0	U0	V0	W0	D0	I0	J0	Q0	;rem

En général, le format d'une ligne de table d'outils est le suivant:

- ; Un point-virgule comme premier caractère, aucune donnée
- T Numéro d'outil, 0-99999 (chaque numéro d'outil doit être unique)
- P Numéro de poche, 1-99999 (chaque numéro de poche doit être unique)
- X..W Offset d'outil sur les axes spécifiés - nombre à virgule flottante
- D diamètre d'outil - nombre à virgule flottante, valeur absolue
- I angle frontal (tour seulement) - nombre à virgule flottante
- J angle de dos (tour seulement) - nombre à virgule flottante
- Q orientation de l'outil (tour seulement) - entier de 0 à 9
- ; début de commentaire ou remarque - texte

Le fichier commence par un point-virgule en première ligne, suivi par les caractéristiques de 1000 outils au maximum. [1]

Les versions antérieures de LinuxCNC avaient deux différents formats de table d'outils un pour les fraiseuses et un pour les tours, mais depuis la version 2.4.x, un format unique est utilisé pour toutes les machines. Il suffi d'ignorer les parties de la table d'outils qui ne se rapportent pas la machine actuelle, ou que vous n'avez pas besoin d'utiliser.

Chaque ligne du fichier de la table d'outils après le point-virgule ouvrant, contient les données pour un seul outil. Une ligne peut contenir jusqu'à 16 entrées, mais peut aussi en contenir beaucoup moins.

Les unités utilisées pour les longueurs, diamètres, etc. sont en unités machine.

Vous voudrez probablement maintenir les entrées d'outils dans l'ordre croissant, surtout si vous utilisez un changeur d'outils aléatoire. Bien que la table d'outils permettent des numéros d'outils dans n'importe quel ordre.

Chaque ligne peut avoir jusqu'à 16 valeurs. Les deux premières valeurs sont requises. La dernière valeur (un point-virgule suivi d'un commentaire) est optionnelle. La lecture sera rendue plus facile si les valeurs sont disposées en colonnes, comme indiqué dans le tableau, mais la seule exigence sur le format est qu'il y ait au moins un espace ou une tabulation après chacune des valeurs sur une ligne et un saut de ligne à la fin de chaque ligne.

La signification des valeurs et le type de données qu'elles contiennent sont les suivantes:

Numéro d'outil (requis) La colonne T contient un nombre entier non signé, qui représente le code de l'outil. L'opérateur peut utiliser n'importe quel code pour n'importe quel outil, tant que les codes sont des entiers non signés.

Numéro de poche (requis) La colonne P contient un nombre entier non signé, qui représente le numéro de poche (numéro de slot) du changeur d'outils, poche dans laquelle l'outil se trouve. Les entrées de cette colonne doivent être toutes différentes. Le numéro de poche commence typiquement à 1 et va au maximum de poches disponibles sur le changeur d'outils. Mais tous les changeurs d'outils ne suivent pas ce modèle. Votre numéro de poche sera déterminé, par le numéro que votre changeur d'outils utilisera pour se référer à ses poches. Tout cela pour dire que les numéros de poche que vous

[1] Bien que les numéros d'outils puissent aller jusqu'à 99999, le nombre d'outils dans la table, en ce moment, est limité à un maximum de 1000 outils pour des raisons techniques. Les développeurs de LinuxCNC envisagent la possibilité de faire sauter cette limitation. Si vous avez un très gros changeur d'outils, merci d'être patient.

utiliserez seront déterminés par le schéma de numérotation de votre changeur d'outils. Les numéros de poche doivent suivre la même logique que la machine.

Données d'offset des outils (optionnelles) Les colonnes de données d'offset (XYZABCUVW) contiennent des nombres réels qui représentent les offsets d'outil pour chacun des axes. Ce nombre sera utilisé si, en usinage, les offsets de longueur d'outil sont utilisés et que l'outil concerné est sélectionné. Ces nombres peuvent être positif, égaux à zéro ou négatif, ils sont en fait, complètement optionnels. Bien qu'il vaudrait mieux qu'il y ait au moins une valeur ici, sinon il n'y aurait aucun intérêt à se servir d'une entrée complétement vide dans la table d'outils.

Sur une fraiseuse classique, on trouvera probablement une entrée en Z (offset de longueur d'outil). Sur un tour classique, on trouvera certainement un entrée en X (offset d'outil en X) et une en Z (offset d'outil en Z). Sur une fraiseuse classique utilisant la compensation de rayon d'outil, on trouvera une valeur en D pour l'offset de diamètre. Sur un tour classique utilisant la compensation de diamètre de bec d'outil, une valeur sera entrée en D (diamètre de bec).

Un tour demande encore d'autres information additionnelles pour décrire la forme et l'orientation de l'outil. Ainsi, sans tenir compte des angles ni des faces de l'outil, qui sont de la compétence du tourneur, on trouvera une valeur en I (angle avant) et en J (angle de dos) ainsi qu'une valeur en Q (orientation).

Une description complète des outils de tour ce trouve ici.

La colonne Diamètre contient un nombre réel. Ce nombre est utilisé seulement si la compensation est activée lors de l'usage de cet outil. Si la trajectoire programmée avec la compensation active, est un des bords de la matière à usiner, cette valeur doit être un nombre réel positif, représentant le diamètre mesuré de l'outil. Si la trajectoire programmée, toujours avec la compensation active, est prévue pour un diamètre nominal d'outil, ce nombre doit être très petit (négatif ou positif, mais proche de zéro), il représente seulement la différence entre le diamètre nominal et le diamètre mesuré de l'outil. Si la compensation n'est pas utilisée avec un outil, cette valeur est sans importance.

La colonne des commentaires peut optionnellement être utilisée pour décrire l'outil. Elle commence par un point-virgule, elle peut contenir n'importe quel texte pour le seul bénéfice de l'opérateur.

6.2 Fichier d'outils et compensations

6.2.1 Fichier d'outils

Les longueurs et diamètres d'outils peuvent être lus dans une table d'outils ou provenir d'un mot spécifié pour activer la compensation d'outil.

6.2.2 Compensation d'outil

La compensation d'outil peut causer beaucoup de problèmes aux meilleurs programmeurs. Mais elle peut aussi être une aide puissante quand elle est utilisée pour aider l'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 LinuxCNC.

6.2.3 Compensation de longueur d'outil

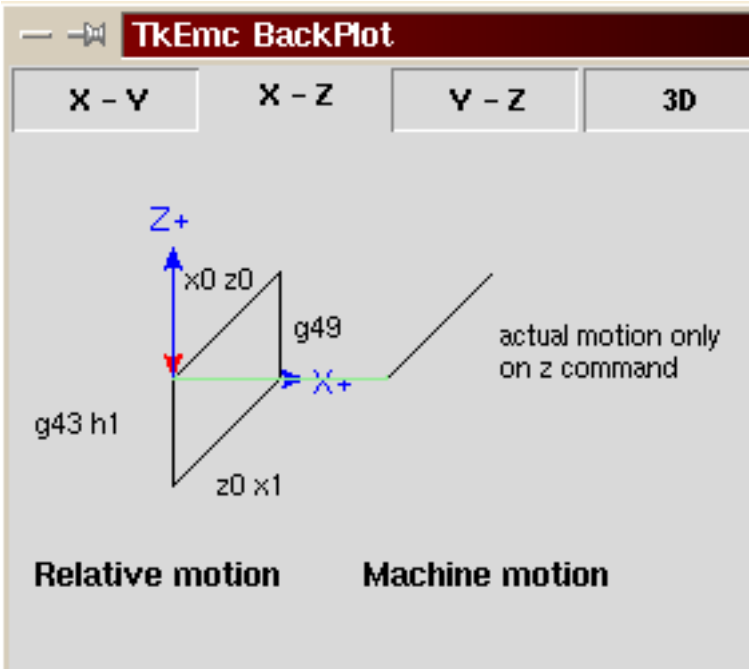
Les compensations de longueur d'outil sont données comme des nombres positifs dans la table d'outils. Une compensation d'outil est programmée en utilisant G43 Hn, où n est le numéro d'index de l'outil souhaité dans la table d'outil. Il est prévu que toutes les entrées dans la table d'outils soit positives. La valeur de H est vérifiée, elle doit être un entier non négatif quand elle est lue. L'interpréteur 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 réglages 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.

L'outil #1 fait un pouce de 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.

6.2.4 Compensation de rayon d'outil

La compensation de rayon d'outil permet de suivre un parcours sans se préoccuper du diamètre de l'outil. La seule restriction, c'est que les mouvements d'entrée doivent être au moins aussi long que le rayon de l'outil utilisé.

Il y a deux parcours que l'outil peut prendre pour usiner un profil quand la compensation de rayon est activée, un parcours à gauche du profil et un à droite du profil. Pour les visualiser, il faut s'imaginer être debout sur la pièce, marchant en suivant l'outil pendant que celui-ci progresse dans la matière. G41 fait passer l'outil à gauche du profil et G42 le fait passer à droite du profil.

Le point final de chaque mouvement, dépends du mouvement suivant. Si le mouvement suivant crée un angle extérieur, le mouvement se terminera à l'extrémité de la ligne de coupe compensée. Si le mouvement suivant crée un angle intérieur, l'outil s'arrêtera avant d'interférer avec la matière de la pièce. La figure suivante montre comment le mouvement se termine à différents endroits, dépendants du mouvement suivant.

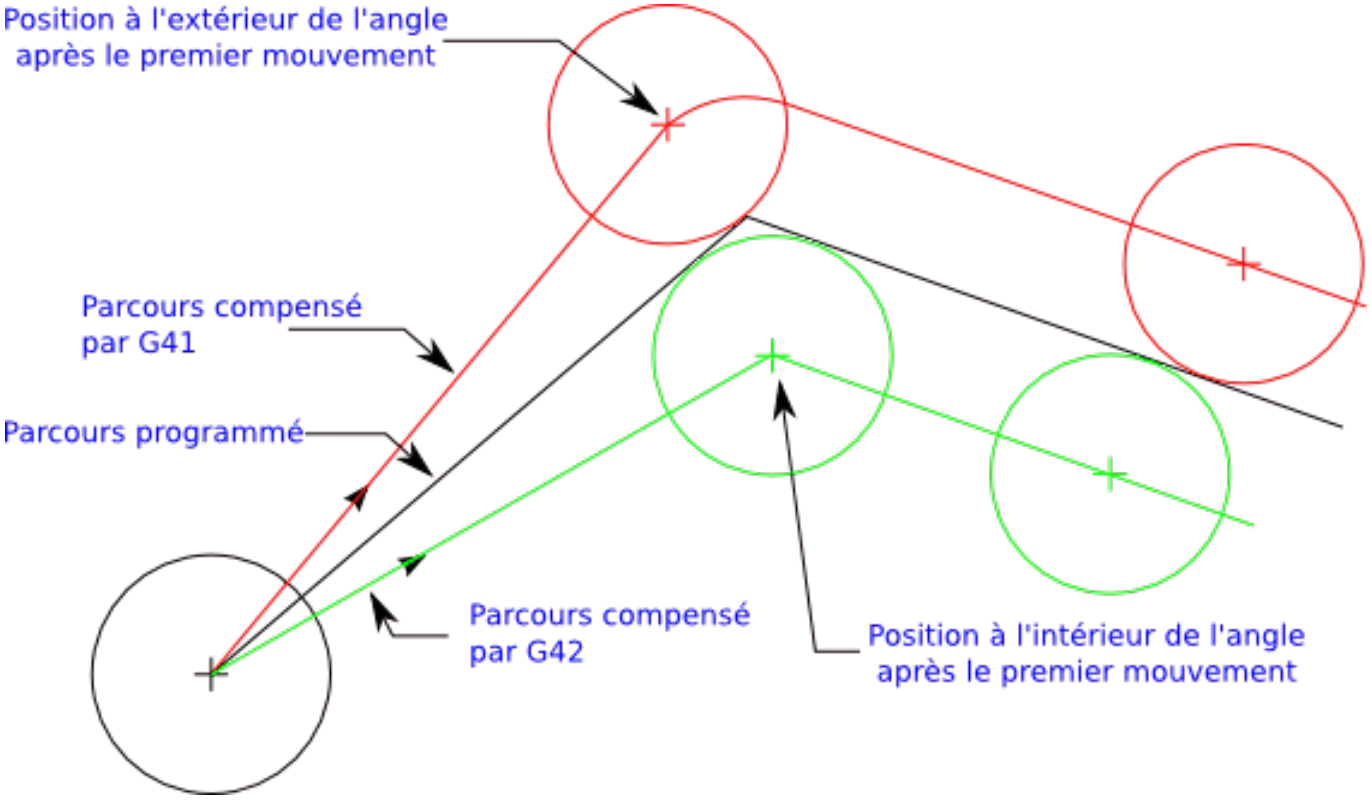


Figure 6.2: Point final de la compensation

6.2.4.1 Vue générale

La compensation de rayon d'outil utilise les données de la table d'outils pour déterminer le décalage nécessaire. Les données peuvent être introduites à la volée, avec G10 L1.

Tout mouvement suffisamment long pour arriver en position compensée, sera un mouvement d'entrée valide. La longueur minimale équivaut au rayon de l'outil. Ça peut être un mouvement en vitesse rapide au dessus de la pièce. Si plusieurs mouvements en vitesse rapide sont prévus après un G41/G42, seul le dernier placera l'outil en position compensée.

Dans la figure suivante, on voit que le mouvement d'entrée est compensé à droite du profil. Ce qui aura pour effet, lors du mouvement d'entrée, de déplacer le centre de l'outil, d'un rayon d'outil à droite

de X0. Dans ce cas, le mouvement d'entrée laissera un petit plot de matière en raison du décalage de compensation et de l'arrondi de l'outil.

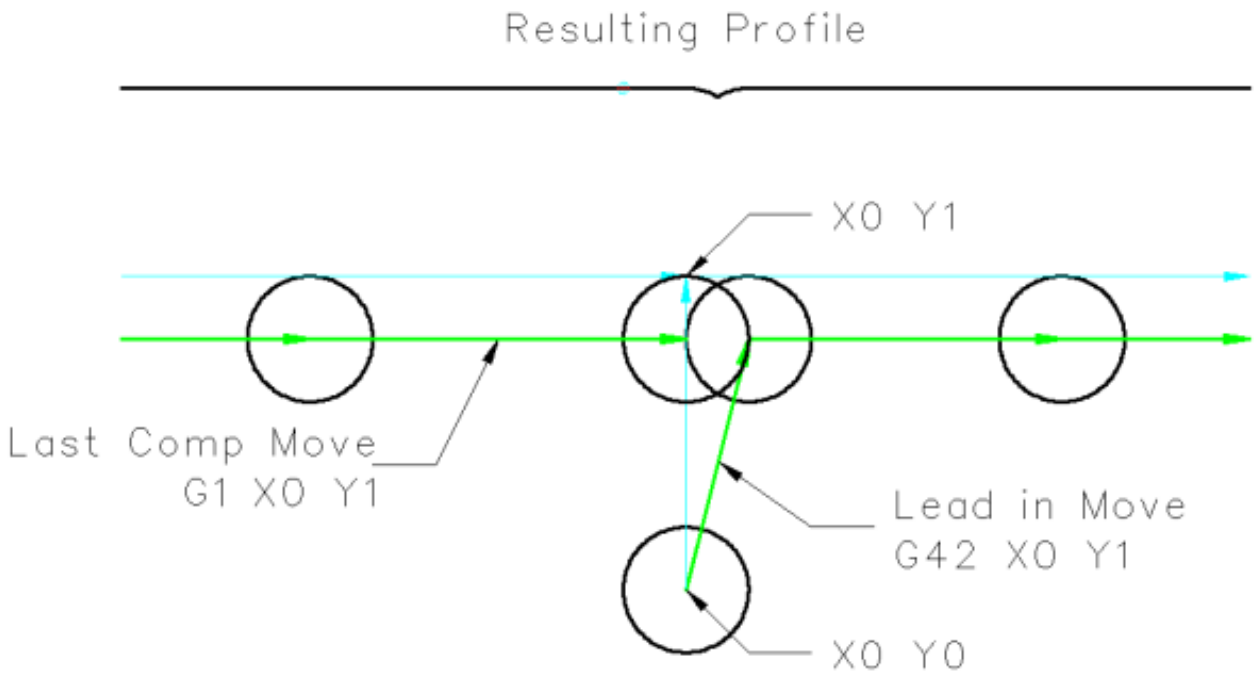


Figure 6.3: Mouvement d'entrée

Un mouvement en Z est possible pendant que le contour est suivi dans le plan XY. Des portions du contour peuvent être sautées en rétractant l'axe Z au dessus du bloc et en amenant Z au dessus du prochain point de départ.

Des mouvements en vitesse rapide peuvent être programmé avec les compensations d'outil actives.

- Débuter tout programme avec un G40 pour être sûr que la compensation est désactivée.

6.2.4.2 Exemples de profils

```
G-Code
F25  (Vitesse d'avance travail)
G40  (Révocation des compensations)
G10 L1 P1 R.25 Z1  (Réglages en table d'outils)
T1 M6  (Appel de l'outil 1)
G42  (Compensation d'outil à droite du profil)
G1 X1 Y1  (Mouvement d'entrée)
X3  (Parcours trajectoire de coupe)
Y3
X1
Y1
G40  (Révocation de la compensation)
G0 Y0 Y0  (Dégagement de l'outil)
M2  (Fin de programme)
```

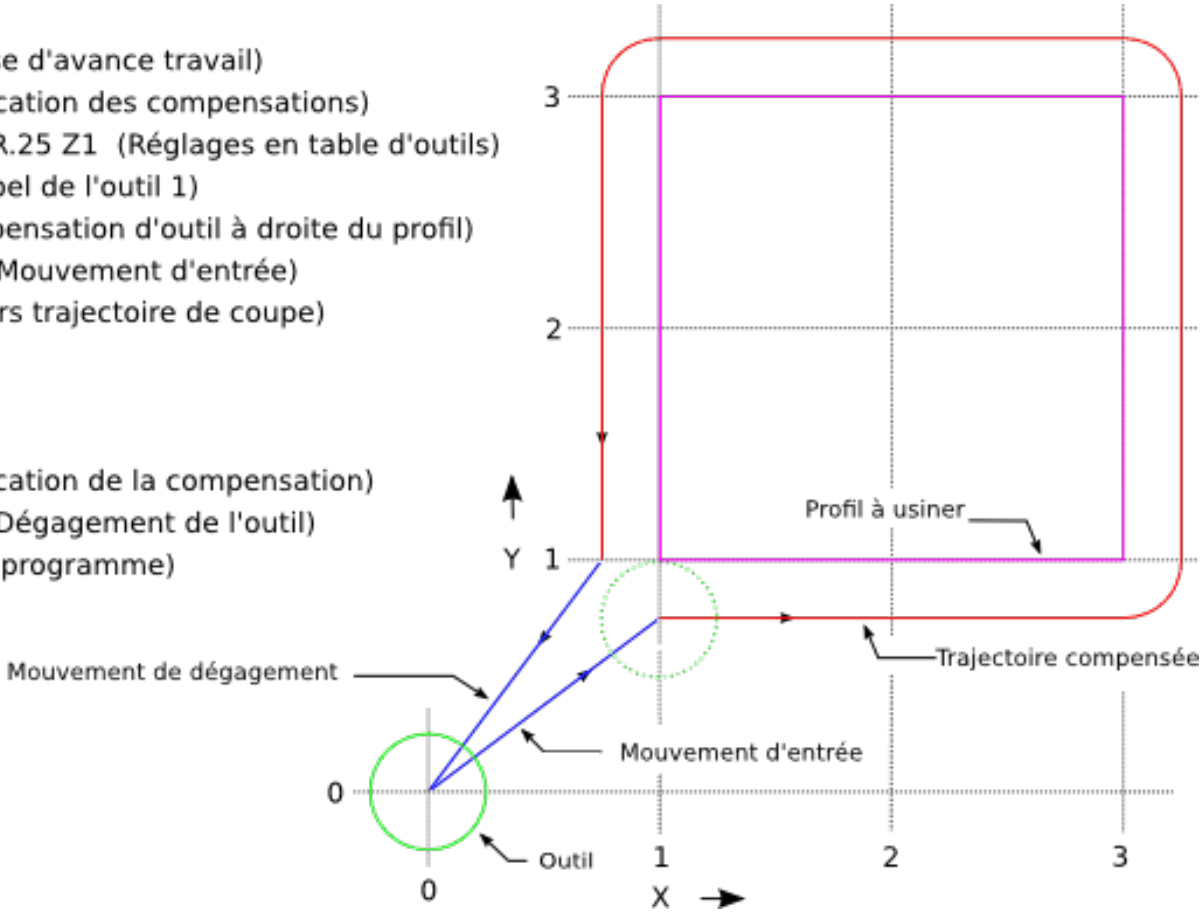


Figure 6.4: Profil extérieur

```
G-Code
F30  (Sélection de la vitesse)
G10 L1 P1 R.25 Z1  (Réglages en table d'outils)
T1 M6  (Chargement de l'outil 1)
G0 Z0  (Dégagement de l'outil sur l'axe Z)
G41  (Compensation d'outil à gauche du profil)
X3 Y2  (Déplacement rapide sur point de départ)
G1 X4 Z-1  (Plongée vers la profondeur de coupe)
G3 X5 Y3 J1  (Arc d'entrée)
G1 Y5  (Parcours sur la trajectoire de coupe)
X1
Y1
X5
Y3
G3 X4 Y4 I-1  (Arc de sortie)
G0 Z0  (Dégagement de l'outil sur axe Z)
G40  (Désactivation de la compensation d'outil)
G0 X1 Y1  (Déplacement sur position sécurisée)
T0 M6  (Déchargement de l'outil)
M2  (Fin de programme)
```

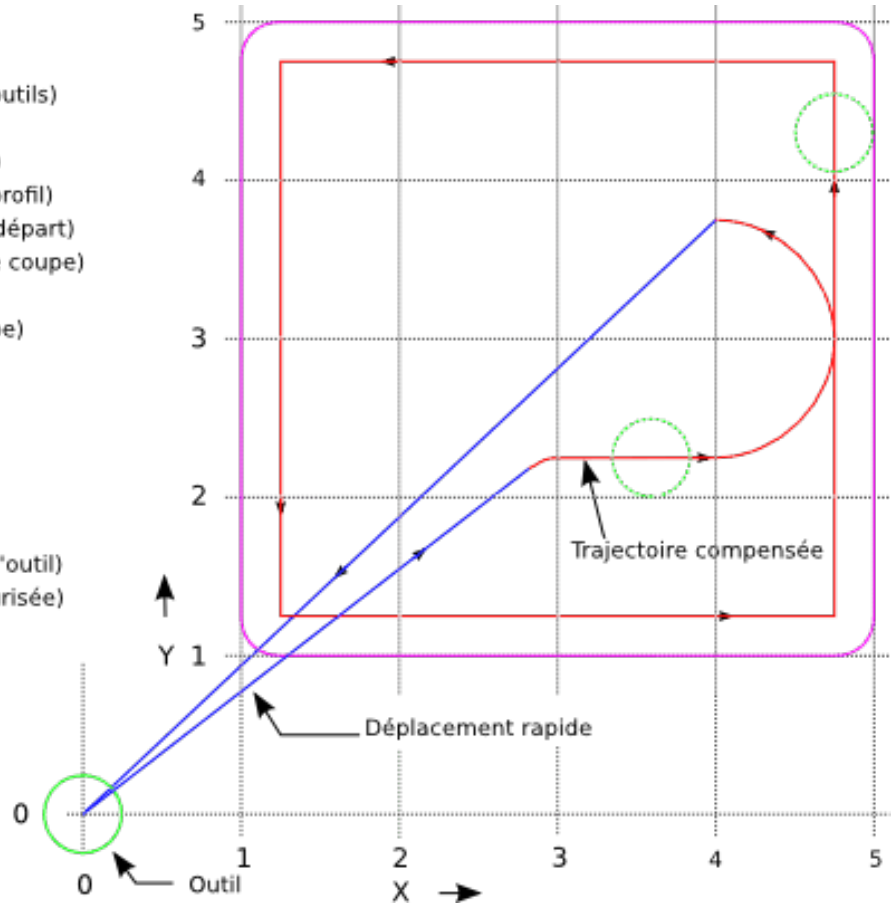


Figure 6.5: Profil intérieur

6.2.5 Deux exposés sur les compensations d'outil

Ces deux exposés ont été écrits par des experts de la CNC, nous pensons que leur lecture sera très utile.

By Jon Elson

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 le 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 légèrement 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 parcours 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.

6.2.5.1 Compensation de rayon d'outil, détails

By Tom Kramer and Fred Proctor

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 parcours suivi par le centre de l'outil. Dans les deux cas le triangle gris est conservé. Le parcours de gauche (avec les coins arrondis) est le parcours 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.

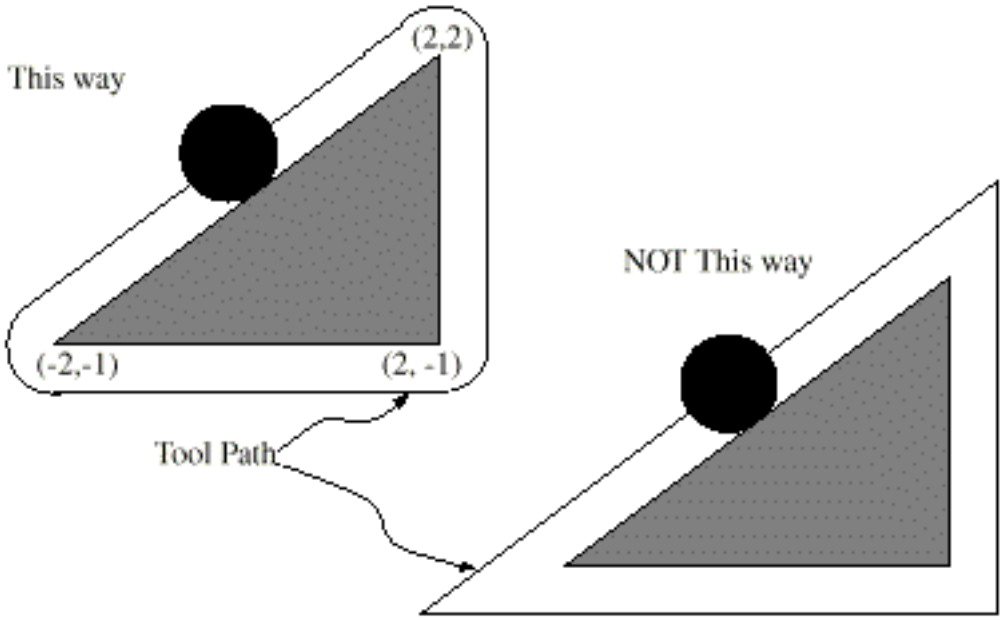


Figure 6.6:

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 parcours et jusqu'au point où l'usinage doit reprendre, l'axe Z plongera de nouveau en position. Ces dégagements 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.

- 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 figure:7 précédente, 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 un 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.

L'interpréteur actuel requiert une valeur D sur chaque ligne contenant un mot G41 ou G42. Le 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. Toutes les poches de la table d'outils peuvent être sélectionnés de cette façon. Le nombre D n'est pas nécessairement le même que le numéro de poche de l'outil monté dans la broche.

La compensation de rayon d'outil utilise les données fournies par la table d'outils de la machine. Pour chaque poche 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 cette poche et son diamètre actuel). La table d'outils est indexée par les numéros de poche. Reportez vous à la page des Fichiers d'outils pour savoir comment remplir ces fichiers.

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 parcours 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.

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'exception 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.

Lorsque le contour est sur le parcours d'outil, la trajectoire décrite dans le programme correspond au parcours 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 parcours 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 parcours 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 légèrement sur-dimensionné. La valeur du diamètre sera un petit nombre négatif si l'outil est légèrement 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 parcours 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 parcours 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 G-code 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 parcours d’outil)
N0050 G1 Y-1 (usinage le long du côté droit du parcours 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 parcours 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 légèrement intérieure au parcours 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.

Les messages en rapport avec la compensation de rayon d’outil, délivrés par l’interpréteur sont les suivants:

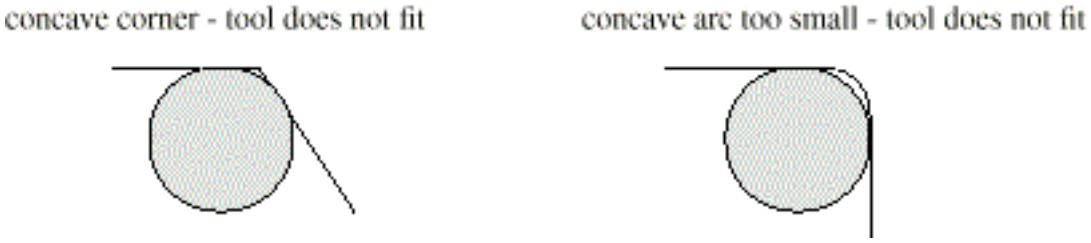
- Impossible de changer les décalages d’axes avec la compensation de rayon d’outil
- Impossible de changer d’unité avec la compensation de rayon d’outil
- Impossible d’activer la compensation de rayon d’outil en dehors du plan XY
- Action impossible, la compensation de rayon d’outil est déjà active
- Impossible d’utiliser G28 ou G30 avec la compensation de rayon d’outil
- Impossible d’utiliser G53 avec la compensation de rayon d’outil
- Impossible d’utiliser le plan XZ avec la compensation de rayon d’outil
- Impossible d’utiliser le plan YZ avec la compensation de rayon d’outil
- Coin concave avec la compensation de rayon d’outil
- Interférence de l’outil avec une partie finie de la pièce avec la compensation de rayon d’outil [2]
- Mot D sur une ligne sans mot de commande G41 ni G42
- Index de rayon d’outil trop grand
- Le rayon de l’outil n’est pas inférieur au rayon de l’arc avec la compensation de rayon
- Deux G-codes du même groupe modal sont utilisés.

[2] 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.

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 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.



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 parcours d'outil qui doit être suivi) et non une approximation. A cet égard, l'interpréteur utilisé par LinuxCNC 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.

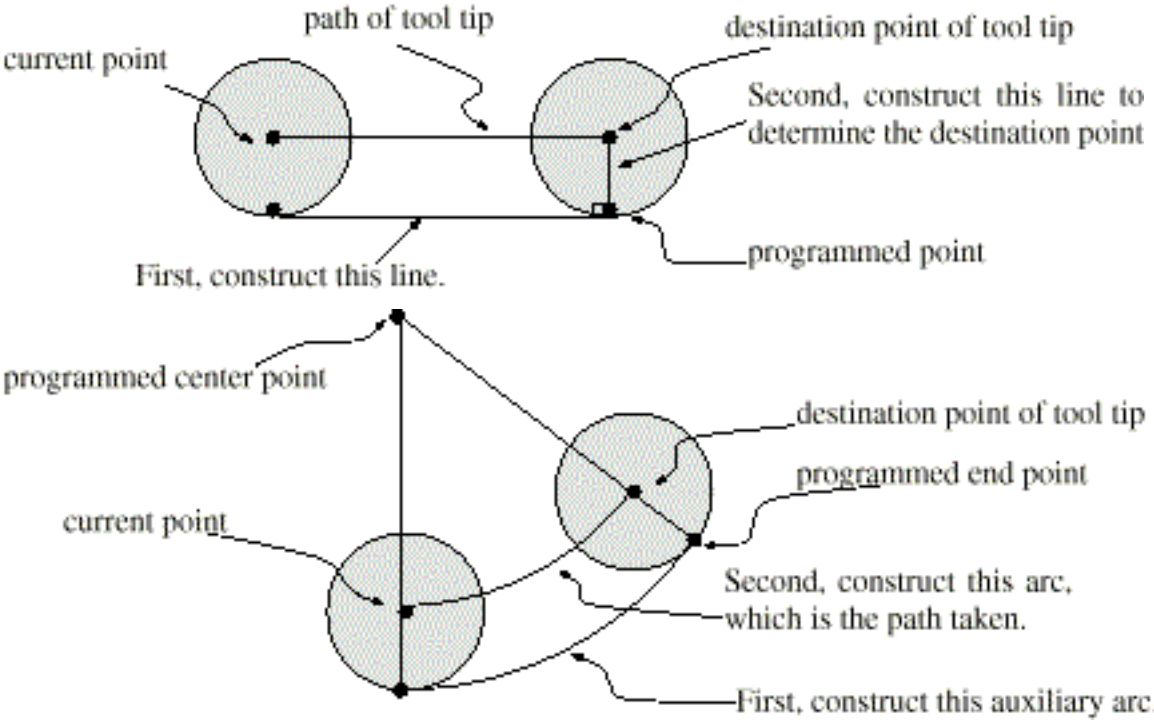
Il n'est pas possible de passer d'un index de rayon d'outil à un autre alors que la compensation est active. Il est également impossible de basculer la compensation d'un côté à l'autre avec la compensation active. Si le prochain point de destination XY est déjà dans le périmètre d'action de l'outil quand 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 programme NC, ce qui n'est pas permis.

6.2.5.2 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, tangent 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.



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 tangent à l'arc auxiliaire pendant tout le mouvement. C'est ce que montre 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 aigu se trouve dans le parcours, un arc est inséré pour tourner autour de l'angle. Le rayon de cet arc sera de la moitié 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.

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 aigus, un seul mouvement d'entrée (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.

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.

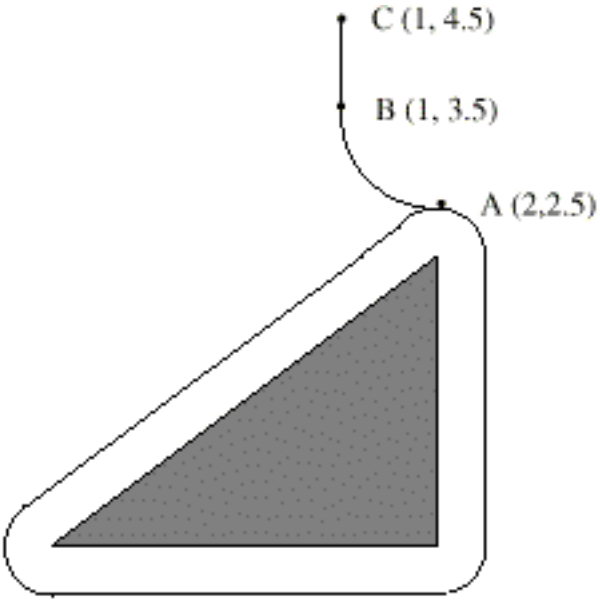


Figure 11. Cutter Radius Compensation Entry Moves

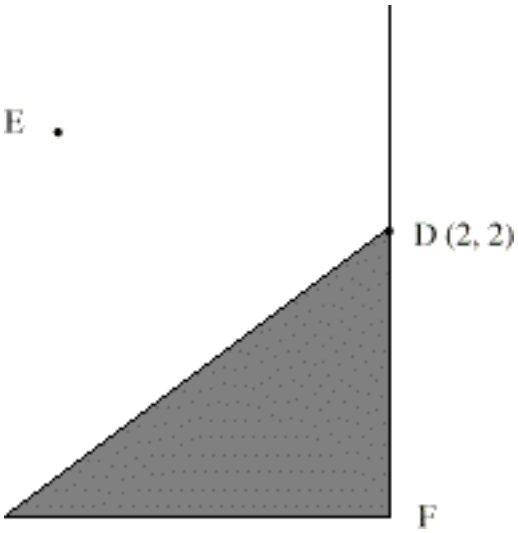
Dans cet exemple, l'arc AB et la ligne BC sont très larges, ce n'est pas nécessaire. Pour un contour sur parcours d'outil, le rayon de l'arc AB demande juste à être légèrement plus grand que la variation maximale du rayon de l'outil par rapport à son rayon nominal. Également, pour un contour sur parcours d'outil, le côté choisi pour la compensation doit être celui utilisé si l'outil est sur-dimensionné. Comme mentionné précédemment, si l'outil est sous-dimensionné, l'interpréteur basculera de l'autre côté.

Si le contour est sur le profil du matériau et qu'il comprends des angles aigus quelque part sur le contour, une méthode simple pour faire l'entrée est possible. Voir la figure 12.

Premièrement, choisir un angle aigu, 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 un angle aigu sur un contour sur parcours d'outil, si l'outil est sur-dimensionné, mais elle échouera si il est sous-dimensionné.



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.

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 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 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 tout moment.

6.2.5.3 Exemples de Jon Elson

Toutes les informations spécifiques au système se réfèrent au programme LinuxCNC 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 corps de 12mm 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 malgré le ressort, mais qui la laisse centrée à quelques dixièmes 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 LinuxCNC 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, LinuxCNC va s'arrêter là, avec un message d'erreur. Même si le gougeage est de .001mm 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 que LinuxCNC ne râle 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:

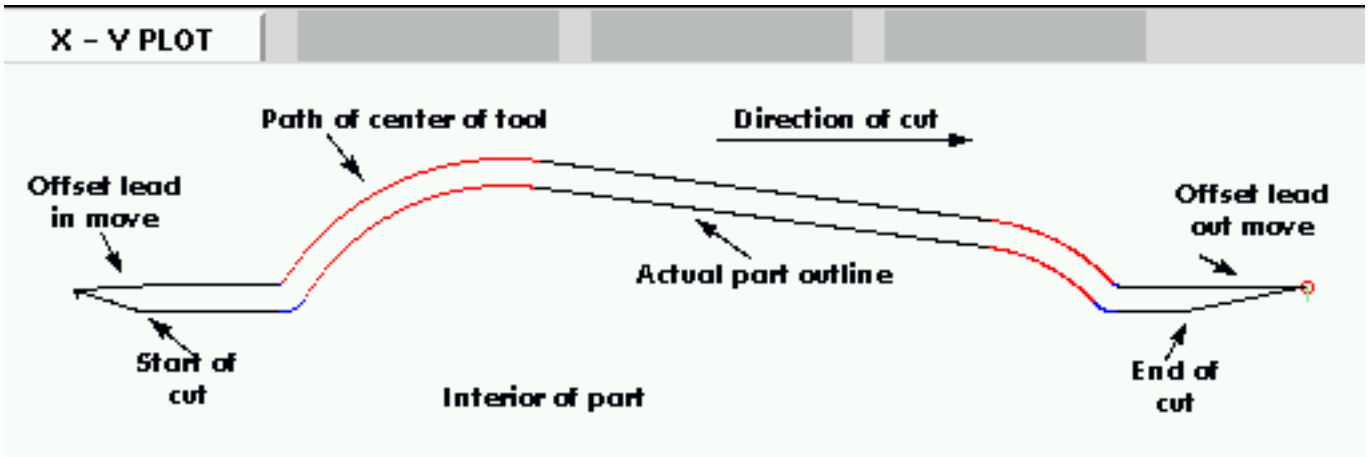
- G40 Annuler la compensation de rayon d'outil
- G41 Activer la compensation, outil à gauche du profil
- G42 Activer la compensation, outil à droite du profil

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 DIAMÈTRES 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.

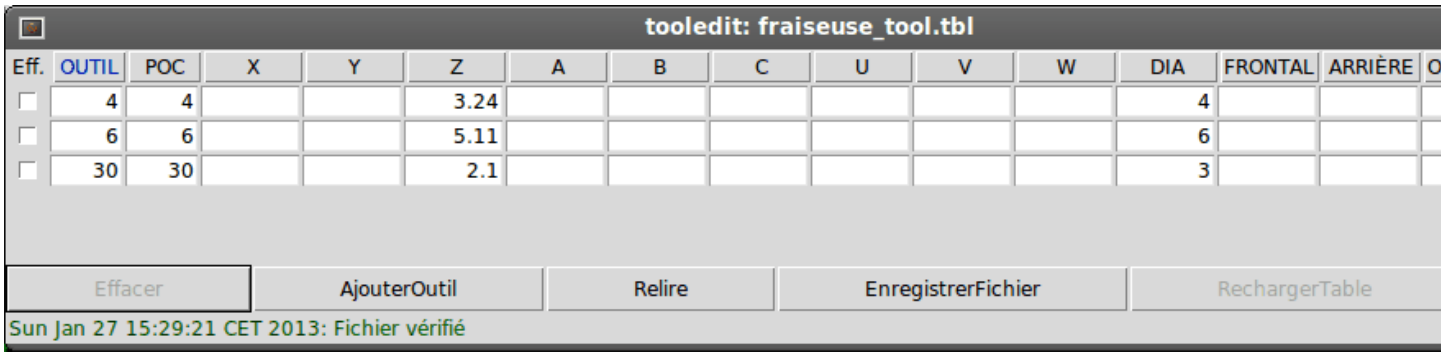


Certaines autres choses sont à noter, le programme commence avec G40, pour désactiver les compensations éventuellement actives. Cela évite un tas d'ennuis quand le programme s'arrête à cause d'une erreur de concavité, mais laisse la compensation désactivée. Noter aussi, en ligne 15, G17 est utilisé pour 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. LinuxCNC est très 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 degrés, alors les problèmes commencent avec LinuxCNC. 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.

6.3 Éditeur graphique de table d'outils

6.3.1 Versions requises pour l'éditeur d'outils

Note
Les éléments de tooledit modifiables décrits ici sont disponibles dans les versions 2.5.1 et suivantes. Dans la version 2.5.0, l'interface graphique ne permet pas ces ajustements.



Le programme tooledit rafraichi le fichier de la table d'outils avec les changements édités puis enregistrés avec le bouton EnregistrerFichier. Le bouton EnregistrerFichier met à jour le fichier système mais une action manuelle dans le menu Fichier -> Recharger la table d'outils est requise pour mettre à jour la table d'outils en cours d'utilisation par l'instance courante de LinuxCNC. Avec l'interface graphique Axis, il est toutefois possible, grâce au bouton RechargerTable, de rafraichir le fichier tout en mettant à jour la table d'outils courante. Ce dernier bouton est disponible uniquement quand la machine est En marche et inactive.

6.3.2 Choix des colonnes

Par défaut, le programme tooledit affiche toutes les colonnes utilisables dans une table d'outils. Très peu de machines utilisent tous les paramètres, les colonnes affichées peuvent être limitées aux paramètres utilisés. Pour cela, passer une ligne de paramètre dans le fichier ini, comme ci-dessous:

Syntaxe

```
[DISPLAY]
TOOL_EDITOR = tooledit nom_colonne nom_colonne ...
```

Exemple pour les colonnes Z et DIAM

```
[DISPLAY]
TOOL_EDITOR = tooledit Z DIAM
```

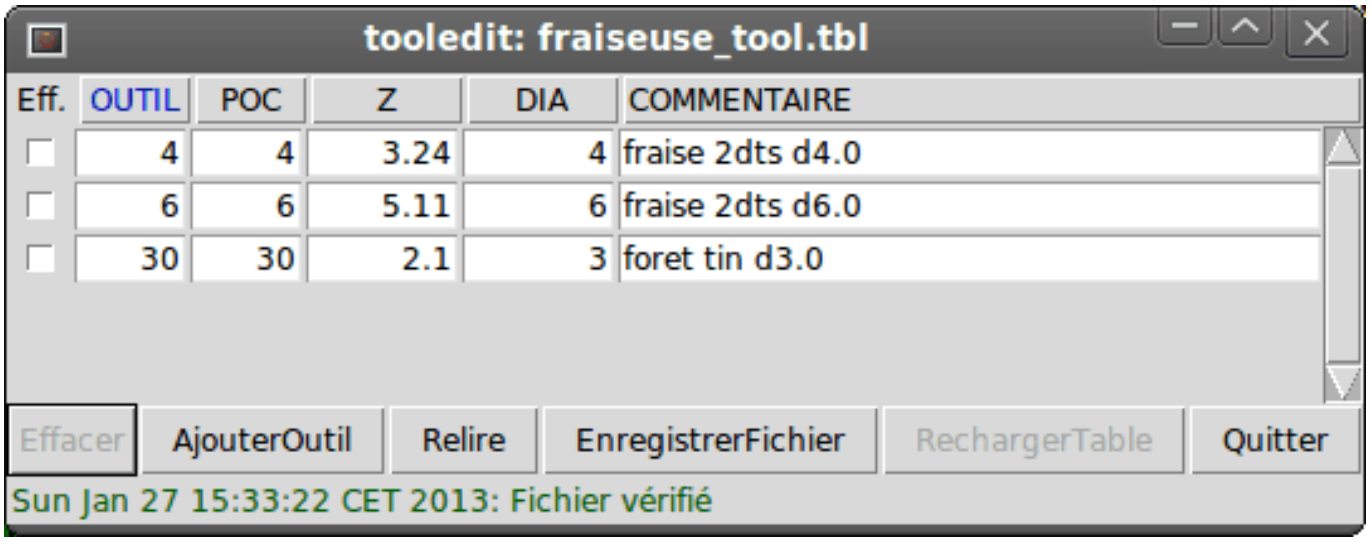
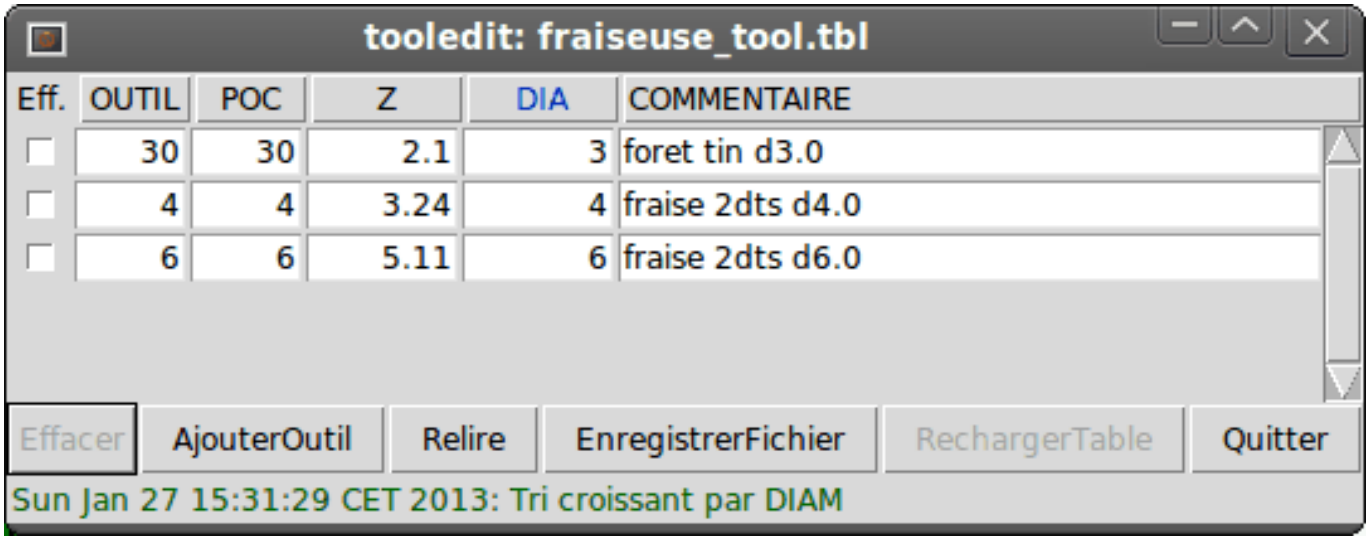


Figure 6.7: Résultat obtenu

6.3.3 Tri par colonne

Les outils de la table affichée peuvent être triés selon n'importe quelle colonne, par ordre croissant ou décroissant, en cliquant sur l'entête de la colonne. Un second clic inverse l'ordre de tri. Le tri par colonne nécessite une version de TCL/TK >= 8.5



Sous Ubuntu Lucid 10.04 tcl/tk8.4 est installé par défaut. Il est possible d'ajouter tcl/tk8.5 avec la commande suivante:

```
sudo apt-get install tcl8.5 tk8.5
```

Si le tri ne fonctionne pas il peut être nécessaire d'activer tcl/tk8.5 avec les commandes:

```
sudo update-alternatives --config tclsh  ;# choisir l'option pour tclsh8.5
sudo update-alternatives --config wish   ;# choisir l'option pour wish8.5
```

6.3.4 Utilisation autonome

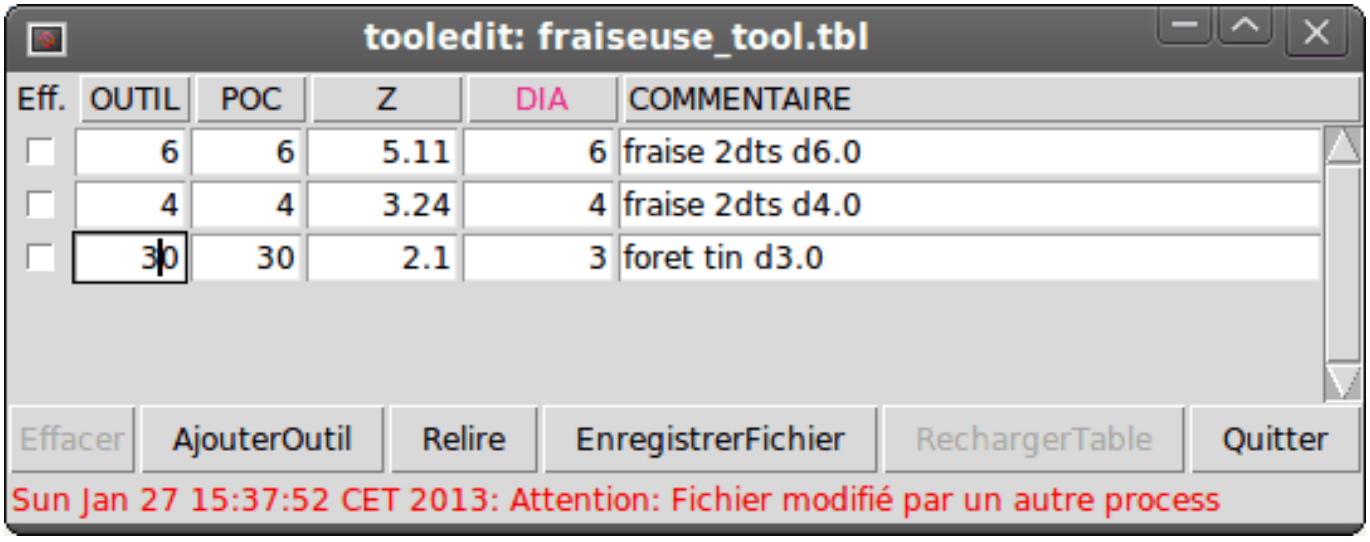
Le programme tooledit peut aussi être invoqué comme un programme autonome. Par exemple si le programme est dans le PATH de l'utilisateur, taper tooledit en suivant une des syntaxes suivantes selon le besoin:

Tooledit autonome

```
tooledit
Usage:
tooledit nomfichier
tooledit [colonne_1 ... colonne_n] nomfichier
```

Pour synchroniser un tooledit autonome avec l'application LinuxCNC courante, le nom de fichier doit être le même que celui spécifié par le paramètre **[EMCIO]TOOL_TABLE** dans le fichier ini.

Lors de l'utilisation de tooledit pendant la marche de LinuxCNC, l'exécution d'un g-code ou d'un autre programme peut altérer les données de la table d'outils. Toute modification est alors détectée par tooledit qui affiche le message: Attention: Fichier modifié par un autre process



L'affichage de la table d'outils dans tooledit peut être rafraichi par le bouton Relire, pour relire le fichier modifié.

Le nom de la table d'outils est spécifié dans le fichier ini par l'entrée:

```
[EMCIO]TOOL_TABLE = nom_fichier_table_
```

Le fichier de table d'outils est éditable avec n'importe quel simple éditeur de texte. (et non avec un traitement de texte)

L'interface graphique Axis peut optionnellement utiliser un paramètre du fichier ini pour spécifier quel éditeur doit être utilisé:

```
[DISPLAY]TOOL_EDITOR = path_pour_editeur
```

Par défaut, le programme tooledit est utilisé. Cet éditeur supporte tous les paramètres de table d'outils, permet l'ajout ou l'effacement d'outils et fourni un certain nombre de contrôles de validité des paramètres introduits.

Part IV

Configuration

Chapter 7

General Info

7.1 Test des capacités temps réel

7.1.1 Test de latence

Ce test est le premier test qui doit être effectué sur un PC pour savoir si celui-ci est capable de piloter une machine CNC.

La latence correspond au temps pris par le PC pour stopper ce qui est en cours et répondre à une requête externe. Dans notre cas, la requête est l'horloge qui sert au cadencement des impulsions de pas. Plus basse est la latence, plus rapide pourra être l'horloge, plus rapides et plus douces seront les impulsions de pas.

La latence est de loin plus importante que la vitesse du CPU. Un vieux Pentium III qui répond aux interruptions avec 10 microsecondes entre chacune, peut donner de meilleurs résultats que le dernier modèle de µP ultra rapide.

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 si le matériel envisagé est apte, c'est d'exécuter un test de latence.

La seule façon de découvrir ce qu'il en est sur un PC est d'exécuter le test de latence de HAL. Pour exécuter ce test, ouvrir une fenêtre de terminal à partir de Applications → Accessoires → Terminal et exécuter la commande suivante:

```
latency-test
```

Une fenêtre comme ci-dessous devrait s'ouvrir:

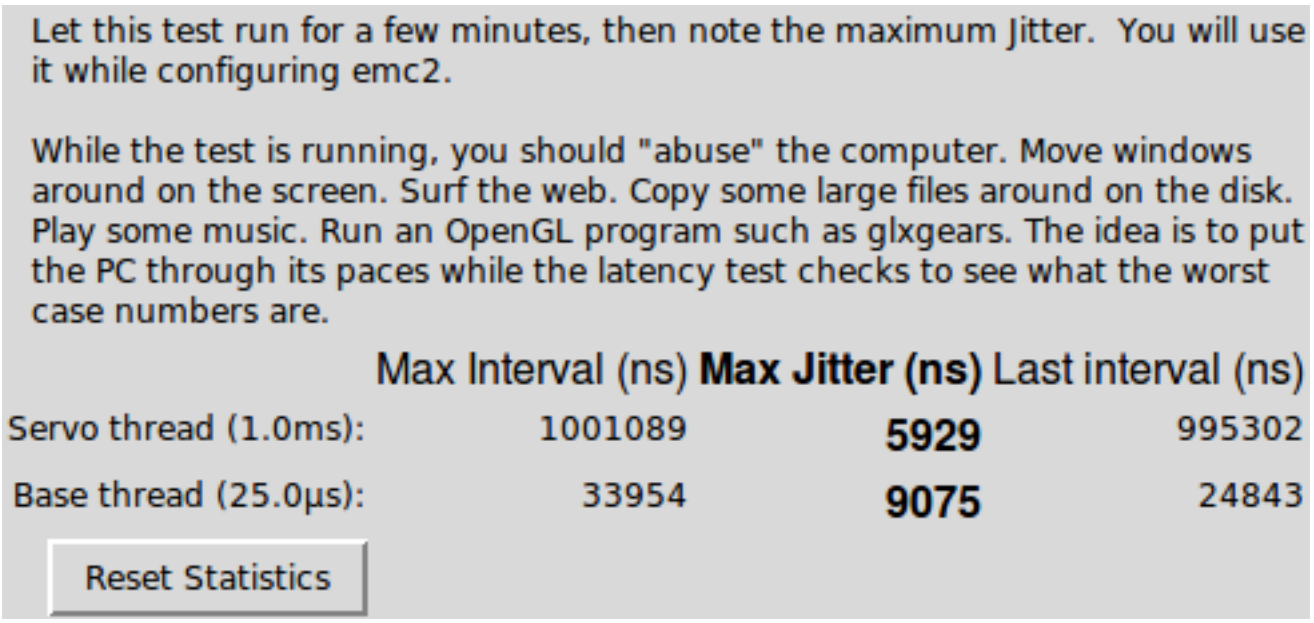


Figure 7.1: Test de latence de HAL

Alors que le test est en cours d'exécution, il faut charger l'ordinateur au maximum. Déplacer les fenêtres sur l'écran. Surfer sur le Web. Écouter de la musique. Exécuter un programme OpenGL comme glxgears. L'idée est de charger le PC au maximum pour que le temps de latence soit mesuré dans le cas le plus défavorable et donc, connaitre la latence maximale.

Note
Ne pas exécuter LinuxCNC ou Stepconf pendant que latency-test est en cours d'exécution.

La colonne max jitter et la ligne Base Thread de l'exemple ci-dessus donne 9075. Ce qui représente 9075 nanosecondes, soit 9.075 microsecondes. Noter ce nombre et l'entrer dans Stepconf quand il sera demandé.

Dans cet exemple de test de latence il n'a fallu que quelques secondes pour afficher cette valeur. Il est toutefois préférable de le laisser tourner 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.

Alors, comment interpréter les résultats? Si le résultat de Max Jitter est en dessous d'environ 15-20 µs (15000-20000 nanosecondes), l'ordinateur pourra donner d'excellents résultats pour la génération logicielle des pas. Si le temps de latence est à plus de 30-50 microsecondes, de bons résultats seront obtenus, mais la vitesse maximum sera un peu faible, spécialement si des micropas sont utilisés ou si le pas de la vis est fin. Si les résultats sont de 100µS 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 LinuxCNC, en utilisant des micropas ou pas.

Note
Si une latence élevée est obtenue, 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 d'occasion à $5US a résolu le problème. LinuxCNC n'exige pas de matériel de pointe.

7.1.2 Adresses des ports

Pour ceux qui construisent leur matériel, il est facile et économique d'augmenter le nombre d'entrées sorties d'un PC en lui ajoutant une carte PCI fournissant un ou deux ports parallèles supplémentaires. Faire suivre ces ports d'une couche d'opto-isolation est utile pour éviter les courts circuits pouvant détruire la carte, voir même toute la carte mère. LinuxCNC supporte un maximum de 8 ports parallèles.

Certaines parmi les bonnes cartes parallèles sont à base de chipset Netmos. Elles fournissent un signal +5V bien propre, elles fournissent un ou deux ports parallèles.

Pour trouver les adresses d'entrées/sorties de ces cartes, ouvrir une console et utiliser la commande en ligne:

```
lspci -v
```

Rechercher ensuite dans la liste de matériel fourni, le nom du chipset de la nouvelle carte, dans cette exemple c'est l'entrée NetMos Technology pour une carte à deux ports:

```
0000:01:0a.0 Communication controller: \
      Netmos Technology PCI 9815 Multi-I/O Controller (rev 01)
      Subsystem: LSI Losgic / Symbios Logic 2POS (2 port parallel adapter)
      Flags: medium devsel, IRQ 5
      I/O ports at b800 [size=8]
      I/O ports at bc00 [size=8]
      I/O ports at c000 [size=8]
      I/O ports at c400 [size=8]
      I/O ports at c800 [size=8]
      I/O ports at cc00 [size=16]
```

Après expérimentation, il se trouve que le premier port (incorporé à la carte) utilise la troisième adresse de la liste (c000) et le deuxième port (raccordé par une nappe) utilise la première adresse (b800).

Il est alors possible d'ouvrir dans l'éditeur le fichier .hal de la machine et d'insérer l'adresse trouvée à l'endroit approprié.

```
loadrt hal\_parport cfg="0x378 0xc000"
```

Noter la présence des guillemets "" encadrant les deux adresses, ils sont obligatoires dès qu'il y a plus d'une carte.

Il est nécessaire également d'ajouter les fonctions de lecture (read) et d'écriture (write) pour la nouvelle carte. Par exemple:

```
addf parport.1.read base-thread 1
addf parport.1.write base-thread -1
```

Noter que les valeurs peuvent être différentes de celles de cet exemple. Les cartes Netmos sont Plug-N-Play, elles peuvent donc changer leur adressage selon le connecteur PCI dans lequel elles sont placées. Si l'installation des cartes PCI de la machine est modifiée, ne pas oublier de vérifier leurs adresses avant de lancer LinuxCNC.

7.2 Réglages des pas à pas

7.2.1 Obtenir le meilleur pilotage logiciel possible

Faire générer les impulsions de pas au logiciel présente un gros 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 inconvénients:

- La fréquence maximum des impulsions est limitée.
- Les impulsions générées sont irrégulières à cause du bruit.
- Elles sont sujettes à la charge du CPU

Ce chapitre présente certaines mesures qui vous aideront à obtenir les meilleurs résultats du logiciel.

7.2.1.1 Effectuer un test de latence

Le CPU n'est pas le seul facteur déterminant la latence. Les cartes mères, cartes graphiques, ports USB et nombre d'autres choses peuvent la dégrader. La meilleure façon de savoir ce que vous pouvez attendre d'un PC consiste à exécuter le test de latence RTAI.

Lancer un test comme décrit au chapitre sur le test de latence.

7.2.1.2 Connaître ce dont vos cartes de pilotage ont besoin

Les différente marques de cartes de pilotage de moteurs pas à pas demandent toutes des timing différents pour les impulsions de commande de pas et de direction. Aussi vous avez besoin d'accéder (ou Google) à la fiche des spécifications techniques de votre carte.

Par exemple, le manuel du Gecko G202 indique:

```
Step Frequency: 0 to 200 kHz
Step Pulse b’’“b’’0b’’”b’’ Time: 0.5 µs min (Step on falling edge)
Step Pulse b’’“b’’1b’’”b’’ Time: 4.5 µs min
Direction Setup: 1 µs min (20 µs min hold time after Step edge)
```

Les spécifications du Gecko G203V indiquent:

```
Step Frequency: 0 to 333 kHz
Step Pulse b’’“b’’0b’’”b’’ Time: 2.0 µs min (Step on rising edge)
Step Pulse b’’“b’’1b’’”b’’ Time: 1.0 µs min

Direction setup:
    200 ns (0.2µs) before step pulse rising edge
    200 ns (0.2µs) hold after step pulse rising edge
```

Un carte Xylotex donne dans ses données techniques un superbe graphe du timing nécessaire, il indique:

```
Minimum DIR setup time before rising edge of STEP Pulse 200ns Minimum
DIR hold time after rising edge of STEP pulse 200ns
Minimum STEP pulse high time 2.0µs
Minimum STEP pulse low time 1.0µs
Step happens on rising edge
```

Notez les valeurs que vous trouvez, vous en aurez besoin pour la prochaine étape.

7.2.1.3 Choisir la valeur de BASE_PERIOD

BASE_PERIOD est l'horloge de votre LinuxCNC. A chaque période, le générateur d'impulsions de pas décide si il est temps pour une autre impulsion. Une période plus courte vous permettra de générer plus d'impulsions par seconde, dans les limites. Mais si vous la réglez trop bas, votre ordinateur va passer autant de temps à générer des impulsions de pas que pour exécuter tous le reste de ses tâches,

il finira peut-être même par se bloquer. La latence et la génération de pas exigent d'affecter la plus courte période utilisable, comme nous le verrons un peu plus loin.

Regardons l'exemple du Gecko en premier. Le G202 peut gérer des impulsions restant à l'état bas pendant 0.5µs et à l'état haut pendant 4.5µs, il a besoin que la broche de direction soit stable 1µs avant le front descendant et qu'elle reste stable pendant 20µs après le front descendant. La plus longue durée est de 20µs, c'est le temps de maintien. Une approche simple consisterait à fixer la période à 20µs. Ce qui signifierait que tous les changements d'état des lignes STEP et DIR serait espacés de 20µs. C'est tout bon, non?

Faux! Si la latence était de zéro, et que tous les fronts soient espacés de 20µs, tout irait bien. Mais tous les ordinateurs ont une latence. Si l'ordinateur a 11µs de latence, celà signifie que, ce que l'ordinateur exécute aura parfois un retard de 11µs et la fois suivante pourra être juste à l'heure, le délai entre le premier et le second sera seulement de 9µs. Si le premier génére l'impulsion de pas et le second change la broche de direction, le timing de 20µs requis par le G202 sera tout simplement violé. Cela signifie que votre moteur aura peut être fait un pas dans la mauvaise direction et que votre pièce ne sera pas à la cote.

Le côté vraiment mauvais de ce problème est qu'il peut être très très rare. Les pires latences sont celles qui ne se produisent que quelques fois par minute. Les chances qu'une mauvaise latence de ce genre arrive juste quand le moteur est en train de changer de direction sont faibles. Ainsi, vous avez de très rares erreurs qui vous ruinent une pièce de temps en temps et qui sont impossibles à résoudre.

La façon la plus simple pour éviter ce problème est de choisir une BASE_PERIOD qui soit la somme de la plus longue période requise par votre carte plus la durée de la pire latence de votre ordinateur. Si vous utilisez un Gecko avec un temps de maintien exigé de 20µs et que votre test de latence vous avait donné une latence maximum de 11µs, alors si vous définissez BASE_PERIOD à 20+11 = 31µs (31000 nanosecondes dans le fichier ini), vous aurez la garantie de répondre aux exigences de votre carte de pilotage.

Mais c'est un compromis. Faire une impulsion de pas demande au moins deux périodes. Une pour débuter l'impulsion, et une pour y mettre fin. Etant donné que la période est de 31µs, il faut 2x31 = 62µs pour créer une impulsion de pas. Ce qui signifie que la fréquence de pas maximum sera seulement de 16129 pas par seconde. Pas très bon. (Mais n'abandonnez pas, nous avons encore quelques réglages à faire dans la section suivante.)

Pour la Xylotex, la configuration demande des temps de maintien très courts de 200ns chacun (0.2µs). Le temps le plus long est de 2µs. Si vous avez 11µs de latence, alors vous pouvez définir BASE_PERIOD aussi bas que 11+2 = 13µs. Se débarrasser du long temps de maintien de 20µs aide vraiment. Avec une période de 13µs, un pas complet ne dure que 26µs = 2x13 et la fréquence maximum est de 38461 pas par seconde!

Mais ne commencez pas à célébrer celà. Notez que 13µs est une période très courte. Si vous essayez d'exécuter le générateur de pas toutes les 13µs, il ne restera peut-être pas assez de temps pour faire autre chose et votre ordinateur se bloquera. Si vous visez des périodes de moins de 25µs, vous devez commencer à 25µs ou plus, lancer LinuxCNC et voir comment les choses réagissent. Si tout va bien, vous pouvez réduire progressivement la période. Si le pointeur de la souris commence à être sacadé et que le reste du PC ralentit, votre période est un peu trop court. Retournez alors à la valeur précédente qui permettent le meilleur fonctionnement.

Dans ce cas, supposons que vous ayez commencé à 25µs, en essayant descendre à 13µs, vous trouvez que c'est autour de 16µs que se situe la limite la plus basse et qu'en dessous l'ordinateur ne répond plus très bien. Alors, vous utilisez 16µs. Avec une période à 16µs et une latence à 11µs, le temps de sortie le plus court sera de 16-11 = 5µs. La carte demande seulement 2µs, ainsi vous aurez une certaine marge. Il est bon d'avoir une marge si vous ne voulez pas perdre de pas parce que vous auriez réglé un timing trop court.

Quel est la fréquence de pas maximum? Rappelez-vous, deux périodes pour faire un pas. Vous avez réglé la période à 16µs alors qu'un pas prend 32µs. Il fonctionnera à 31250 pas par seconde, ce qui n'est pas mal.

7.2.1.4 Utiliser steplen, stepspace, dirsetup, et/ou dirhold

Dans la section précédente, nous avons utilisé la carte de puissance Xylotex pour piloter nos moteurs avec une période de 16µs ce qui nous a donné une fréquence de pas de 31250 pas par seconde maximum. Alors que la Gecko a été bloquée à 31µs avec une assez mauvaise fréquence de pas de 16129 pas par seconde. L'exemple de la Xylotex est au mieux de ce que nous puissions faire. Mais la Gecko peut être ameliorées.

Le problème avec le G202 est le temps de maintien demandé de 20µs. Ca plus la latence de 11µs nous oblige à utiliser une période longue de 31µs. Mais le générateur de pas logiciel de LinuxCNC a un certain nombre de paramètres qui permettent d'augmenter les différentes durées d'une période à plusieurs autres. Par exemple, si steplen passe de 1 à 2, alors il y aura deux périodes entre le début et la fin de l'impulsion. De même, si dirhold passe de 1 à 3, il y aura au moins trois périodes entre l'impulsion de pas et un changement d'état de la broche de direction.

Si nous pouvons utiliser dirhold pour le temps de maintien de 20µs demandé, alors le temps le plus long suivant sera de 4.5µs. Ajoutez les 11µs de latence à ces 4.5µs, et vous obtenez une période minimale de 15.5µs. Lorsque vous essayez 15.5µs, vous trouvez que l'ordinateur est très lent, donc vous régler sur 16µs. Si nous laissons dirhold à 1 (par défaut), alors le temps minimum entre le pas et la direction est de 16µs moins la période de latence de 11µs = 5µs, ce qui n'est pas suffisant. Nous avons besoin de 15 autres µs, puisque la période est de 16µs, nous avons besoin d'une période de plus. Nous allons donc passer dirhold de 1 à 2. Maintenant, le temps minimum entre la fin de l'impulsion et l'impulsion de changement de direction est de 5+16 = 21µs et nous n'avons pas à craindre que la Gecko parte dans la mauvaise direction en raison de la latence.

Si l'ordinateur a une latence de 11µs, alors la combinaison d'une période de base de 16µs et d'une valeur de dirhold de 2 garanti que nous serons toujours dans le respect des délais exigés par la Gecko. Pour les pas normaux (sans changement de direction), l'augmentation de la valeur de dirhold n'aura aucun effet. Il faudra deux périodes d'un total de 32µs pour faire un seul pas et nous avons la même fréquence de 31250 pas par seconde que nous avions eu avec la Xylotex.

Le temps de latence de 11µs utilisé dans cet exemple est très bon. Si vous travaillez par le biais de ces exemples avec des latences plus grandes, comme 20 ou 25µs, la fréquence de pas la plus grande à la fois pour la Xylotex et la Gecko sera plus faible. Mais les mêmes formules sont applicables pour calculer un BASE_PERIOD optimal et pour régler dirhold ou d'autres paramètres du générateur de pas.

7.2.1.5 Pas de secret!

Pour un système à moteurs pas à pas avec générateur de pas logiciel rapide et fiable, vous ne pouvez pas deviner la période et les autres paramètres de configuration. Vous devez faire des mesures sur votre ordinateur et faire les calculs qui garantirons les meilleurs signaux dont les moteurs ont besoin.

Pour rendre le calcul plus facile, j'ai créé une feuille de calcul Open Office: Step Timing Calculator (en) - Calculatrice Calendrier étape (fr). Vous entrez les résultats du test de latence et les timing de votre carte de pilotage et la feuille calcule la meilleure BASE_PERIOD. Ensuite, vous testez la période pour vous assurer que votre PC ne sera pas ralenti ou bloqué. Enfin, vous entrez dans la période actuelle et la feuille de calcul vous indiquera le réglage de stepgen nécessaire pour répondre aux exigences de votre carte de pilotage. Elle calcule aussi la fréquence de pas maximum que vous serez en mesure de générer.

J'ai ajouté quelques petites choses à la feuille de calcul pour calculer la fréquence maximum et quelques autres calculs.

7.3 Moteurs pas à pas

Si ce que vous obtenez ne correspond pas à ce que vous espériez, la plupart du temps c'est juste un petit manque d'expérience. Accroître son expérience permet souvent une meilleure compréhension

globale. Porter un diagnostic sur plusieurs problèmes est toujours plus facile en les prenant séparément, de même qu'une équation dont on a réduit le nombre de variables est toujours plus rapide à résoudre. Dans le monde réel ce n'est pas toujours le cas mais c'est une bonne voie à suivre.

7.3.1 Problèmes communs

7.3.1.1 Le moteur n'avance que d'un pas

La raison la plus fréquente dans une nouvelle installation pour que le moteur ne bouge pas est l'interversion entre le signal de pas et le signal de direction. Si, quand vous pressez le bouton de jog dans un sens puis dans l'autre, le moteur n'avance que d'un pas à chaque fois et toujours dans la même direction, vous êtes dans ce cas.

7.3.1.2 Le moteur ne bouge pas

Certaine interfaces de pilotage de moteurs ont une broche d'activation (enable) ou demandent un signal de pompe de charge pour activer leurs sorties.

7.3.1.3 Distance incorrecte

Si vous commandez une distance de déplacement précise sur un axe et que le déplacement réel ne correspond pas, alors l'échelle de l'axe n'est pas bonne.

7.3.2 Messages d'erreur

7.3.2.1 Erreur de suivi

Le concept d'erreur de suivi est étrange quand il s'agit de moteurs pas à pas. Etant un système en boucle ouverte, aucune contre réaction ne permet de savoir si le suivi est correct ou non. LinuxCNC calcule si il peut maintenir le suivi demandé par une commande, si ce n'est pas possible il stoppe le mouvement et affiche une erreur de suivi. Les erreurs de suivi sur les systèmes pas à pas sont habituellement les suivantes:

- FERROR to small - (FERROR trop petit)
- MIN_FERROR to small - (MIN_FERROR trop petit)
- MAX_VELOCITY to fast - (MAX_VELOCITY trop rapide)
- MAX_ACCELERATION to fast - (MAX_ACCELERATION trop rapide)
- BASE_PERIOD set to long - (BASE_PERIOD trop longue)
- Backlash ajouté à un axe (rattrapage de jeu)

Toutes ces erreurs se produisent lorsque l'horloge temps réel n'est pas capable de fournir le nombre de pas nécessaire pour maintenir la vitesse requise par le réglage de la variable BASE_PERIOD. Ce qui peut se produire, par exemple après un test de latence trop bref pour obtenir un valeur fiable, dans ce cas, revenir à une valeur plus proche de ce qu'elle était et réessayez. C'est également le cas quand les valeurs de vitesse maximum et d'accélération maximum sont trop élevées.

Si un backlash a été ajouté, il est nécessaire d'augmenter STEPGEN_MAXACCEL aux environs du double de MAX_ACCELERATION dans la section [AXIS] du fichier INI et ce, pour chacun des axes sur lesquels un backlash a été ajouté. LinuxCNC utilise une extra accélération au moment de l'inversion de sens pour reprendre le jeu. Sans correction du backlash, l'accélération pour le générateur de pas peut être juste un peu plus basse que celle du planificateur de mouvements.

7.3.2.2 Erreur de RTAPI

Quand vous rencontrez cette erreur:

```
RTAPI: ERROR: Unexpected realtime delay on task n
```

C'est généralement que la variable BASE_PERIOD dans la section [EMCMOT] du fichier ini a une valeur trop petite. Vous devez lancer un Latency Test pendant une durée plus longue pour voir si vous n'avez pas un délai excessif quelque part, responsable de ce problème. Si c'est le cas réajuster alors BASE_PERIOD avec la nouvelle valeur obtenue.

LinuxCNC vérifie le nombre de cycles du CPU entre les invocations du thread temps réel. Si certains éléments de votre matériel provoquent un délai excessif ou que les threads sont ajustés à des valeurs trop rapides, vous rencontrerez cette erreur.

Note
Cette erreur n'est affichée qu'une seule fois par session. En effet, si votre BASE_PERIOD était trop basse vous pourriez avoir des centaines de milliers de messages d'erreur par seconde si plus d'un était affiché.

Plus d'informations sur le test de latence.

7.3.3 Tester

7.3.3.1 Tester le timing des pas

Si un de vos axes vibre, grogne ou fait des petits mouvements dans toutes les directions, c'est révélateur d'un mauvais timing d'impulsions de pas de ce moteur. Les paramètres du pilote matériel sont a vérifier et a ajuster. Il peut aussi y avoir des pertes de pas aux changements de direction. Si le moteur cale complétement, il est aussi possible que les paramètres MAX_ACCELERATION ou MAX_VELOCITY aient des valeurs trop élevées.

Le programme suivant vérifie que la configuration de l'axe Z est correcte. Copiez le programme dans le répertoire de votre linuxcnc/nc_files nommez le TestZ.ngc ou similaire. Initialisez votre machine avec Z = 0.000 sur le dessus de la table. Chargez et lancez le programme. Il va effectuer 200 mouvements d'aller et retour entre 10.00 et 30.00mm. Si vous avez un problème de configuration, la position de l'axe Z affichée à la fin du programme, soit 10.00mm, ne correspondra pas à la position mesurée. Pour tester un autre axe remplacez simplement le Z des G0 par le nouvel axe.

```
( Faire Z=0 au dessus de la table avant de démarrer! )
( Ce programme teste les pertes de position en Z )
( msg, test 1 de la configuration de l'axe Z )
G21 #1000=100 ( boucle 100 fois )
( cette boucle comporte un délai après chaque mouvement )
( test des réglages d'accélération et de vitesse )
o100 while [#1000]
   G0 Z30.000
   G4 P0.250
   G0 Z10.000
   G4 P0.250
   #1000 = [#1000 - 1]
o100 endwhile
( msg, test 2 de la configuration de l'axe Z, pressez S pour continuer)
M1 (un arrêt ici)
#1000=100 ( boucle 100 fois )
( Les boucles suivantes n'ont plus de délai en fin de mouvements )
( test des réglages des temps de maintien des pilotes)
```

```
( et les réglages d’accélération max )
o101 while [#1000]
   G0 Z30.000 .
   G0 Z10.000
   #1000 = [#1000 - 1]
o101 endwhile
( msg, Fin Z doit être à 10mm au dessus de la table )
M2
```

Chapter 8

Configuration

8.1 Configuration rapide pour moteurs pas à pas

Cette section suppose qu'une installation du logiciel à partir du CD Live a été faite. Après cette installation et avant de continuer, il est recommandé de connecter le PC sur Internet pour y faire les dernières mises à jour. Pour les installations plus complexes se référer au Manuel de l'intégrateur.

8.1.1 Test de latence (Latency Test)

Le test de latence détermine la capacité du processeur à répondre aux requêtes qui lui sont faites. Certains matériels peuvent interrompre ce processus, causant des pertes de pas lorsque le PC pilote une machine CNC. Ce test est la toute première chose à faire pour valider un PC. Pour le lancer, suivre les instructions de la section sur le test de latence.

8.1.2 Sherline

Si vous avez une machine Sherline plusieurs configurations prédéfinies sont fournies. Au premier démarrage de LinuxCNC, le sélecteur de configuration s'ouvre, sélectionnez alors le modèle correspondant à votre machine Sherline, puis acceptez d'enregistrer une copie.

8.1.3 Xylotex

Si vous avez une machine Xylotex vous pouvez utiliser l'assistant graphique de configuration fourni avec LinuxCNC et créer rapidement votre configuration personnalisée avec l'assistant Stepconf.

8.1.4 Informations relatives à la machine

But, regrouper les informations à propos des axes de la machine.

Les timings des pilotes sont exprimés en nanosecondes. Si vous n'êtes pas sûr de vous à propos des timings de votre interface, les caractéristiques les plus populaires sont incluses dans l'assistant graphique de configuration. Notez que les pilotes Gecko ont des timings différents les uns des autres. Une liste des caractéristiques courantes est également maintenue sur le Wiki de linuxcnc.org.

Axes	Type de pilote	Step Time ns	Step Space ns	Direction Hold ns	Direction Setup ns
X					
Y					
Z					

[1]

8.1.5 Informations relatives au brochage

But, regrouper les informations à propos des différentes broches du port parallèle utilisées.

Pin de sortie	Fonction	Si différent	Pin d'entrée	Fonction	Si différent
1	Sortie A/U		10	Limite et OM X	
2	X Step		11	Limite et OM Y	
3	X Direction		12	Limite et OM Z	
4	Y Step		13	Limite et OM A	
5	Y Direction		15	Entrée palpeur	
6	Z Step				
7	Z Direction				
8	A Step				
9	A Direction				
14	Broche sens horaire				
16	PWM broche				
17	Valide puissance				

Noter que toutes les broches inutilisées doivent être explicitement indiquées Inutilisé dans le choix déroulant de l'assistant. Elles pourront être modifiées par la suite en relançant Stepconf.

8.1.6 Informations relatives à la mécanique

But, regrouper les informations à propos des réducteurs. Utilisées pour définir la taille d'un pas dans l'unité utilisateur. La taille du pas est utilisée par SCALE dans le fichier .ini.

Axes	Pas par tour	Micropas	Dents moteur	Dents vis	Pas de la vis
X					
Y					
Z					

Pas par tour indique combien de pas moteur sont nécessaires pour que celui-ci fasse un tour. Valeur typique: 200.

[1] ndt: les termes sont laissés dans la langue d'origine pour correspondre aux documentations des constructeurs.

Micropas indique combien d'impulsions le pilote doit recevoir pour que le moteur tourne d'un angle équivalent à un pas.

Si les micropas ne sont pas utilisés, cette valeur devra être mise à 1. Si les micropas sont utilisés, les valeurs les plus courantes sont, 2 pour le demi-pas, 4 pour le quart de pas, 8 ou 10.

Note
Le meilleur choix sera un compromis entre les petites valeurs, qui peuvent rendre le système bruyant à cause des vibrations et les valeurs élevées, qui exigent beaucoup de pas, ce qui diminue la vitesse maximale.

Dents moteur et Dents vis à indiquer si vous avez une réduction poulies/courroie entre le moteur et la vis. Sinon mettez 1 pour les deux.

Pas de la vis indique la longueur de déplacement du mobile pour un tour de la vis d'entrainement de l'axe.

Un exemple en pouces:

```
Moteur            = 200 pas par tour
Pilote            = 10 micropas par pas
Dents côté moteur = 20
Dents côté vis    = 40
Pas de vis        = 0.2000 pouces par tour
```

D'après les informations ci-dessus:

- le mobile se déplacera de 0.200 pouces par tour de vis.
- Le moteur fera 2000 micropas par tour de vis.
- Le pilote demandera 10 micropas pour faire un pas.
- Le pilote recevra 2000 impulsions de pas pour faire tourner le moteur d'un tour.

Encore un autre exemple, en millimètres cette fois:

```
Pas par tour      = 200 pas par tour
Micropas          =   8 micropas
Dents côté moteur =  30
Dents côté vis    =  90
Pas de la vis     =   5.00 mm par tour
```

D'après les informations ci-dessus:

- la vis déplacera le mobile de 5.00 mm par tour.
- Le moteur fera 3 tours pour 1 tour de vis. (90/30)
- Le pilote utilisera 8 micropas pour faire un pas.
- Le pilote aura besoin de 1600 impulsions pour un tour moteur et donc de 4800 pour 1 tour de vis.

8.1.7 Assistants de configuration graphique

- Pour les moteurs pas à pas, voir la documentation de l'assistant graphique Stepconf au chapitre concernant cet assistant.
- Pour les servomoteurs et les moteurs pas à pas, voir la documentation de l'assistant graphique PNCconf au chapitre relatif à cet assistant.

8.2 Configuration de LinuxCNC

8.2.1 Script de lancement

LinuxCNC est lancé par le fichier de script linuxcnc.

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

Avec les options suivantes: * -v = verbose - informations de fonctionnement * -d = commande d'écho à l'écran pour le débogage

Le fichier de script linuxcnc lit le fichier ini puis lance LinuxCNC. La section [HAL] du fichier ini, spécifie l'ordre de chargement des fichiers de HAL, si plusieurs sont utilisés. Après que les fichiers HAL soient chargés, l'interface graphique est chargée à son tour puis le fichier HAL POSTGUI. Si des objets pyvcp ont été créés avec des pins de HAL, le fichier postgui.hal doit effectuer les raccordements à ces pins, se reporter à la section HAL pour plus de détails.

Si aucun fichier ini n'est passé en argument au script linuxcnc, le sélecteur de configuration est lancé pour permettre à l'utilisateur de choisir parmi les exemples de configuration existants.

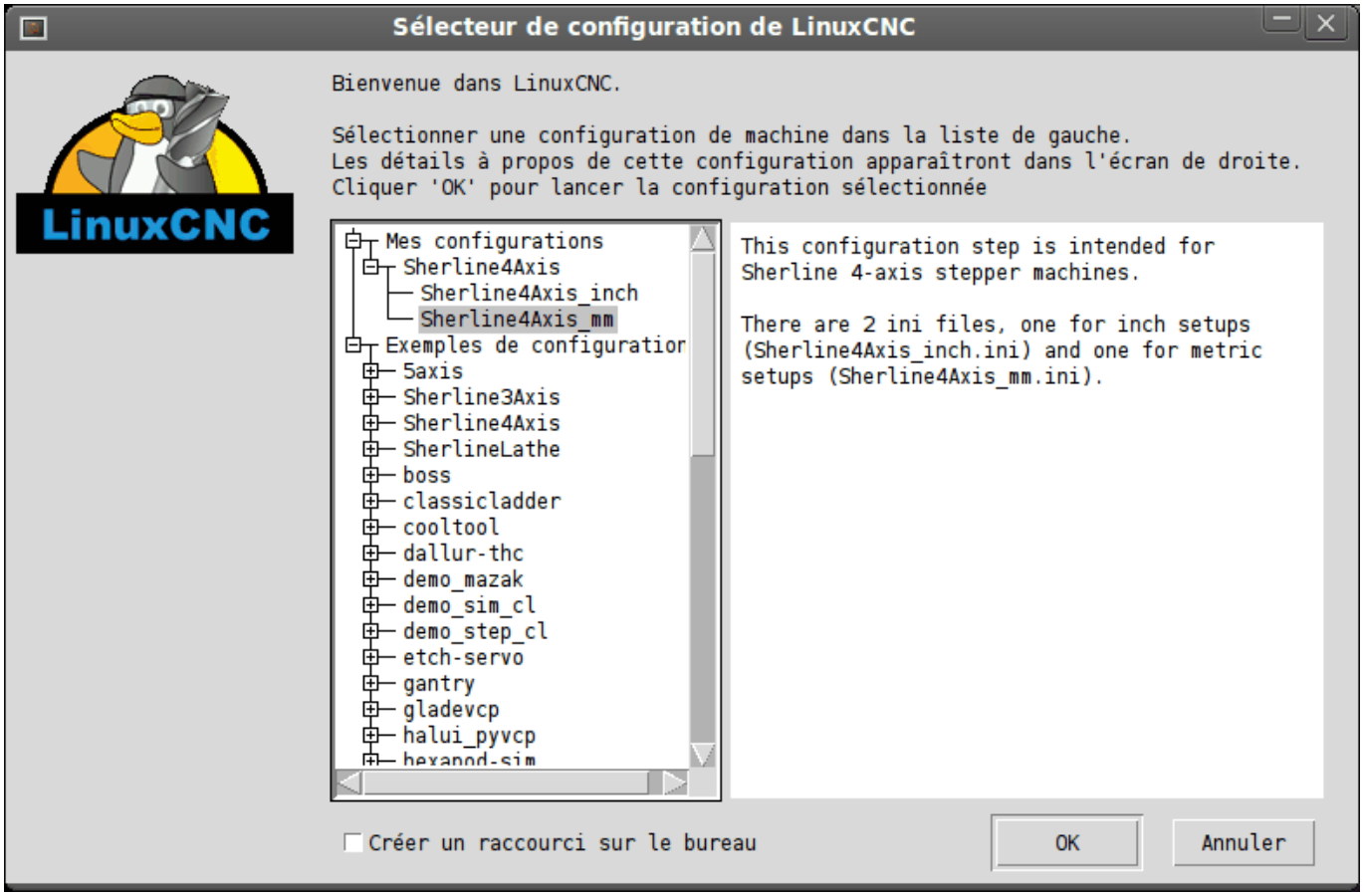


Figure 8.1: Sélecteur de configuration

8.2.2 Fichiers utilisés pour la configuration

LinuxCNC 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.[2] Soyez prudent lorsque vous modifierez ces fichiers, certaines erreurs pourraient empêcher le démarrage de LinuxCNC. Ces fichiers sont lus à chaque fois que le logiciel démarre. Certains d'entre eux sont lus de nombreuses fois pendant l'exécution de LinuxCNC.

Les fichiers de configuration inclus:

- INI Le fichier ini écrase les valeurs par défaut compilées dans le code de LinuxCNC. Il contient également des sections qui sont lues directement par HAL (Hardware Abstraction Layer, couche d'abstraction matérielle).
- HAL Les fichiers hal installent les modules de process, ils créent les liens entre les signaux de LinuxCNC 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 et réutilisées d'une exécution à l'autre.
- TBL Ce fichier contient les informations relatives aux outils. Voir la section Fichier d'outils du Manuel de l'utilisateur pour plus d'infos.
- NML Ce fichier configure les voies de communication utilisées par LinuxCNC. Il est normalement réglé pour lancer toutes les communications avec un seul ordinateur, peut être modifié pour communiquer entre plusieurs ordinateurs.
- .linuxcncrc 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 de LinuxCNC.[3]

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 LinuxCNC et doivent toujours avoir la section et le nom donné à l'item.

8.2.3 Double passe (TWOPASS)

LinuxCNC 2.5 supporte le processus dit TWOPASS des fichiers de configuration hal, ce qui aide à la modularité des fichiers hal et améliore leur lisibilité. (les fichiers Hal sont spécifiés dans le fichier ini de LinuxCNC, dans l'instance HAL sous la forme [HAL]HALFILE=nomdufichier.

Normalement, un jeu de un ou plusieurs fichiers de configuration HAL doivent utiliser une seule et unique ligne loadrt pour charger le module du kernel qui pourra gérer de multiples instances d'un même composant. Par exemple: si vous utilisez une portes AND à deux entrées, composant (and2), à trois endroits différents de votre configuration, vous ne devez avoir que cette seule ligne quelque part pour le spécifier:

```
loadrt and2 count=3
```

Ce qui fournira finalement les composants and2.0, and2.1, and2.2.

Les configurations seront plus lisibles si vous spécifiez les composants sous la forme names=option quand c'est supporté, par exemple:

[2] Ne pas confondre 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.

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

```
loadrt and2 names=aa,ab,ac
```

Ce qui nommera les composants aa, ab, ac.

Il pourrait apparaitre un problème de maintenance pour garder la trace des composants et de leur noms après avoir ajouté (ou enlevé) un composant, vous devrez trouver et mettre à jour, la ligne de directives de loadrt, applicable à ce composant.

Le processus TWOPASS est activé par inclusion d'un paramètre dans le fichier ini:

```
[HAL]TWOPASS=anything
```

Avec ce réglage, vous pouvez avoir de multiples spécifications comme:

```
loadrt and2 names=aa
...
loadrt and2 names=ab,ac
...
loadrt and2 names=ad
```

Ces commandes peuvent être placées dans différents fichiers HALFILES. Les HALFILES sont traités dans leur ordre d'apparition dans le fichier ini.

Avec le processus double passe, tous les [HAL]HALFILES sont lus une première fois et les multiples apparitions de la directive loadrt sont cumulées pour chaque module. Aucune commande hal n'est exécutée lors de cette passe initiale.

Après la passe initiale, les modules sont automatiquement chargés en nombre égal au nombre total lors de l'utilisation de count=option ou de tous les noms spécifiés individuellement lors de l'utilisation de names=option.

Une seconde passe est alors faite pour exécuter toutes les autres instructions de hal spécifiées dans les HALFILES. Les commandes addf qui associent les fonctions de composants avec l'exécution du thread sont exécutées selon leur ordre d'apparition avec les autres commandes dans cette seconde passe.

Bien que vous puissiez utiliser indifféremment les options avec count= ou names=, elles sont toutefois exclusives. Un seul type peut être utilisé pour un même module.

Le processus TWOPASS n'est pas effectif lors de l'usage de names=option. Cette option permet d'avoir un nom unique qui soit mnémonique ou plus pertinent avec la configuration. Par exemple: si vous utilisez un composant dérivé pour estimer la vitesse et l'accélération de chacun des coordonnées (x,y,z), utiliser la méthode count= donnera un composant au nom ésotérique comme ddt.0, ddt.1, ddt.2, etc.

Alternativement, l'utilisation de names=option comme:

```
loadrt ddt names=xvit,yvit,zvit
...
loadrt ddt names=xaccel,yaccel,zaccel
```

donnera des composants plus parlants, nommés xvit,yvit,zvit, xaccel,yaccel, zaccel.

Beaucoup de composants fournis avec la distribution ont été créés avec comp utility et supportent la méthode names=option. Il s'agit notamment de composants logiques qui sont les briques de beaucoup de configurations HAL.

Exemples d'inclusions:

```
and2,ddt,deadzone,flipflop,or2,or4,mux2,mux4,scale,sum2,timedelay,lowpass
```

et beaucoup d'autres.

Les composants utilisateur créés avec comp utility supportent également automatiquement la méthode names=option. En plus des composants générés avec comp utility, quelques autres composants comme encoder et pid supportent aussi names=option.

8.2.4 Organisation du fichier ini

Organisation du fichier ini
Un fichier ini typique suit une organisation simple;

- les commentaires.
- les sections.
- les variables.

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

8.2.4.1 Les 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 ignoré. 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.
```

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

```
DISPLAY = axis
# DISPLAY = touchy
```

Dans cette liste, la variable DISPLAY est positionnée sur axis puisque l'autre est commentée. Si quelqu'un édite une liste comme celle-ci et par erreur, dé-commente deux lignes, c'est la première rencontrée qui sera utilisée.

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

```
INCORRECT = valeur     # et un commentaire

# Commentaire correct
CORRECT = valeur
```

8.2.4.2 Les sections

Les différentes parties d'un fichier .ini sont regroupées en 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 LinuxCNC:

- [EMC] informations générales.
- [DISPLAY] sélection du type d'interface graphique.
- [FILTER] sélection d'un programme de filtrage.
- [RS274NGC] ajustements utilisés par l'interpréteur de g-code.
- [EMCMOT] réglages utilisés par le contrôleur de mouvements temps réel.
- [TASK] réglages utilisés par le contrôleur de tâche.
- [HAL] spécifications des fichiers .hal.

- [HALUI] commandes MDI utilisées par HALUI.
- [TRAJ] réglages additionnels utilisés par le contrôleur de mouvements temps réel.
- [AXIS_n] groupes de variables relatives à chaque axe.
- [EMCIO] réglages utilisés par le contrôleur d'entrées/sorties.

8.2.4.3 Les 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 = jusque 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 LinuxCNC. 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 ceux qui sont utilisés dans les exemples de fichiers de configuration.

Les variables personnalisées peuvent être utilisées dans vos fichiers HAL avec la syntaxe suivante:

```
MACHINE = MaVariable
```

8.2.4.4 Sections et variables utilisateur

Certaines configurations utilisent des sections utilisateur et des variables personnalisées pour regrouper les paramètres en un seul emplacement pour améliorer la lisibilité du fichier ini.

Pour utiliser une section de variable utilisateur dans un fichier HAL, ajouter la section et la variable dans le fichier INI.

Exemple de section utilisateur

```
[OFFSETS]
OFFSET_1 = 0.1234
```

Pour ajouter une variable utilisateur à une section LinuxCNC, inclure simplement cette variable dans la section souhaitée.

Exemple de variable utilisateur

```
[AXIS_0]
TYPE = LINEAR
...
SCALE = 16000
```

Pour utiliser une variable utilisateur dans un fichier HAL, utiliser les noms de section et de variable en lieu et place de leurs valeurs.

Exemple d'utilisation dans un fichier HAL

```
setp offset.1.offset [OFFSETS]OFFSET_1
setp stepgen.0.position-scale [AXIS_0]SCALE
```

Note
La valeur stockée dans la variable doit correspondre au type spécifié pour la pin du composant.

8.2.5 Détails des sections du fichier ini

8.2.5.1 Section [EMC]

- VERSION = $Revision: 1.5 $ - 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.
- DEBUG = 0 - Niveau de débogage 0 signifie qu'aucun message ne sera affiché dans le terminal pendant le fonctionnement de LinuxCNC. Les drapeaux de débogage ne sont généralement utiles que pour les développeurs.

8.2.5.2 Section [DISPLAY]

Les différentes interfaces graphiques utilisent différentes options qui ne sont pas supportées par toutes les interfaces utilisateur. Les deux principales interfaces pour LinuxCNC sont AXIS et Touchy. Axis est une interface pour une utilisation avec un ordinateur classique et son moniteur, Touchy est à utiliser avec les ordinateurs à écran tactile. Pour plus d'informations, voire la section Interfaces du Manuel de l'utilisateur.

- DISPLAY = axis - Le nom de l'interface graphique à utiliser. Les options disponibles sont les suivantes: axis, touchy, tklinuxcnc,
- 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 LinuxCNC. 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_SPINDLE_SPEED = 100 - Vitesse de broche par défaut quand celle-ci démarre en mode manuel. Dans AXIS, si cette variable est absente, la vitesse de démarrage est alors fixée à 1 tr/mn. Ce n'est pas la vitesse minimum.
- PROGRAM_PREFIX = ~/linuxcnc/nc_files - Répertoire par défaut des fichiers de g-codes et emplacement des M-codes définis par l'utilisateur. Les recherches de fichiers s'effectueront d'abords dans cet emplacement, avant les chemins des sous-programmes et des fichiers M utilisateur, si il est spécifié dans la section [RS274NGC].
- INTRO_GRAPHIC = linuxcnc.gif - L'image affichée sur l'écran d'accueil.
- INTRO_TIME = 5 - Durée d'affichage de l'écran d'accueil.
- CYCLE_TIME = 0.05 - Cycle time in seconds that display will sleep between polls.

Les éléments suivants sont utilisés uniquement si AXIS est sélectionné comme programme d'interface utilisateur.

- DEFAULT_LINEAR_VELOCITY = .25 - Vitesse minimum par défaut pour les jogs linéaires, en unités machine par seconde. Seulement utilisé dans l'interface AXIS.
- MIN_VELOCITY = .01 - Valeur approximative minimale du curseur de vitesse de jog.
- MAX_LINEAR_VELOCITY = 1.0 - Vitesse maximum par défaut pour les jogs linéaires, en unités machine par seconde. Seulement utilisé dans l'interface AXIS.
- MIN_LINEAR_VELOCITY = .01 - Approximativement la valeur minimale du curseur de vitesse de jog.
- DEFAULT_ANGULAR_VELOCITY = .25 - Vitesse minimum par défaut pour les jogs angulaires, en unités machine par seconde. Seulement utilisé dans l'interface AXIS.
- MIN_ANGULAR_VELOCITY = .01 - Valeur approximative minimale du curseur de vitesse angulaire de jog.
- MAX_ANGULAR_VELOCITY = 1.0 - Vitesse maximum par défaut pour les jogs angulaires, en unités machine par seconde. Seulement utilisé dans l'interface AXIS.
- INCREMENTS = 1 mm, .5 mm, ... - Définit les incréments disponibles pour le jog incrémental. Les incréments peuvent être utilisés pour remplacer la valeur par défaut. Ces valeurs doivent contenir des nombres décimaux (ex. 0.1000) ou des nombres fractionnaires (ex. 1/16), éventuellement suivis par une unité parmi cm, mm, um, inch, in ou mil. Si aucune unité n'est spécifiée, les unités natives de la machine seront utilisées.
- Distances métriques et impériales peuvent être mélangées
  INCREMENTS = 1 inch, 1 mil, 1 cm, 1 mm, 1 um sont des entrées valides.
- OPEN_FILE = /chemin/complet/du/fichier.ngc Le fichier ngc à utiliser au démarrage d'AXIS. Utilisez une chaîne vide "" et aucun fichier ne sera chargé au démarrage.
- EDITOR = gedit - L'éditeur à utiliser lors du choix Éditer fichier du menu d'AXIS, pour éditer le G-code. Ceci doit être configuré pour que cet item de menu s'active. Une autre possibilité valide est: gnome-terminal -e nano.
- TOOL_EDITOR = tooledit - L'éditeur de texte à utiliser pour éditer les tables d'outils. (par exemple en sélectionnant "Fichiers > Éditer la table. d'outils" dans le menu d'Axis). D'autres entrées comme gedit, gnome-terminal -e vim, gvim ou nano sont valides.
- PYVCP = /filename.xml - Le fichier de description du panneau PyVCP. Voir la section PyVCP.
- LATHE = 1 - Passe l'affichage en mode tour, avec vue de dessus et la visu soit en rayon, soit en diamètre.
- GEOMETRY = XYZABCUVW - Contrôle de prévisualisation du parcours d'outil d'un mouvement rotatif. Cet item consiste en une suite de lettre d'axe, optionnellement précédé d'un signe **-**. Seuls, les axes définis par **[TRAJ]AXES** peuvent être utilisés. Cette séquence spécifie l'ordre dans lequel l'effet de chaque axe est appliqué. Un signe **-** inverse le sens de la rotation. La chaine GEOMETRY correcte dépend de la configuration de la machine et de la cinématique utilisée pour la contrôler. La chaine exemple GEOMETRY=XYZBCUVW est pour une machine à 5 axes pour laquelle la cinématique déplace UVW en coordonnées système de l'outil et XYZ déplace la pièce en coordonnées système. L'ordre des lettres est important, parce qu'il donne expressément l'ordre dans lequel les différentes transformations seront appliquées. Par exemple: tourner autour de C puis de B est différent de tourner autour de B puis de C. La géométrie n'a pas d'effet sans rotation d'axes.

- ARCDIVISION = 64 - Ajuste la valeur de prévisualisation des arcs. Les arcs sont visualisés en les divisant par un nombre de lignes droites; un semi-cercle est divisé en ARCDIVISION de tronçons. Les valeurs élevées donnent une meilleure précision à la pré-visualisation, mais sont plus lentes et donne un écran plus saccadé. Les petites valeurs sont moins précises mais plus rapides, l'affichage résultant est plus rapide. La valeur par défaut de 64 signifie qu'un cercle de 3 pouces maximum sera affiché dans moins de 3 centièmes de mm, (.03%).[4]
- MDI_HISTORY_FILE = - Le nom du fichier d'historique des commandes MDI. Si rien n'est spécifié, Axis enregistrera cet historique dans .axis_mdi_history dans le répertoire home de l'utilisateur. C'est très pratique dans le cas de multiples configurations sur la même machine.
- HELP_FILE = tklinucnc.txt - Chemin du fichier d'aide (non utilisé avec AXIS).

8.2.5.3 Section [FILTER]

AXIS a la possibilité d'envoyer les fichiers chargés au travers d'un programme de filtrage. Ce filtrage peut réaliser toutes sortes de tâches. Parfois aussi simple que s'assurer que le programme se termine bien par M2, ou parfois aussi compliqué que détecter si le fichier d'entrée est une image et en générer le G-code pour graver la forme qu'il à ainsi défini. La section [FILTER] du fichier ini, contrôle comment les filtres fonctionnent. Premièrement, pour chaque type de fichier, écrire une ligne PROGRAM_EXTENSION. Puis, spécifier le programme à exécuter pour chaque type de filtre. Ce programme reçoit le nom du fichier d'entrée dans son premier argument, il doit écrire le code RS274/NGC sur la sortie standard. C'est cette sortie qui sera affichée dans la zone de texte, pré-visualisée dans la zone du parcours d'outil et enfin, exécutée par LinuxCNC quand il sera mis en marche.

```
PROGRAM_EXTENSION = .extension Description
```

Si votre fichier de sortie est tout en majuscules, vous devez ajouter la ligne suivante:

```
PROGRAM_EXTENSION = .NGC XYZ Post Processor
```

Les lignes suivantes ajoutent le support pour le convertisseur image-to-gcode fourni avec LinuxCNC:

```
PROGRAM_EXTENSION = .png,.gif,.jpg Greyscale Depth Image
    png = image-to-gcode
    gif = image-to-gcode
    jpg = image-to-gcode
```

Il est également possible de spécifier un interpréteur:

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

De cette façon, n'importe quel script Python pourra être ouvert et ses sorties seront traitées comme du g-code. Un exemple de script de ce genre est disponible: nc_files/holecircle.py. Ce script crée le G-code pour percer une série de trous séquents à la périphérie d'un cercle. De nombreux générateurs de G-code sont par ailleurs disponibles sur le wiki: à la page des générateurs de G-code.

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

```
FILTER_PROGRESS=%d
```

activeront la barre de progression d'AXIS qui donnera le pourcentage. Cette fonctionnalité devrait être utilisée par tous les filtres susceptibles de fonctionner pendant un long moment.

Les filtres Python doivent utiliser la fonction print pour sortir le résultat dans Axis.

Cet exemple de programme filtre un fichier et ajoute un axe W correspondant à l'axe Z. Il marchera selon la présence d'un espace entre chaque mot d'axe.

[4] Dans LinuxCNC 2.4 et précédents, la valeur par défaut était de 128.

```
#!/usr/bin/env python3

import sys

def main(argv):

  openfile = open(argv[0], 'r')
  file_in = openfile.readlines()
  openfile.close()

  file_out = []
  for line in file_in:
    # print line
    if line.find('Z') != -1:
      words = line.rstrip('\n')
      words = words.split(' ')
      newword = ''
      for i in words:
        if i[0] == 'Z':
          newword = 'W'+ i[1:]
      if len(newword) > 0:
        words.append(newword)
        newline = ' '.join(words)
        file_out.append(newline)
    else:
      file_out.append(line)
  for item in file_out:
    print "%s" % item

if __name__ == "__main__":
   main(sys.argv[1:])
```

8.2.5.4 Section [RS274NGC]

- PARAMETER_FILE = monfichier.var - Le fichier situé dans le même répertoire que le fichier ini qui contiendra les paramètres utilisés par l'interpréteur (enregistré entre chaque lancement).
- ORIENT_OFFSET = 0 - Une valeur flottante ajoutée au paramètre R d'une opération d'orientation de la broche par M19. Utilisée pour définir une position zéro quelconque quelle que soit l'orientation de montage du codeur de broche.
- RS274NGC_STARTUP_CODE = G17 G20 G40 G49 G64 P0.001 G80 G90 G92 G94 G97 G98 - Une chaine de codes NGC qui sera utilisée pour initialiser l'interpréteur. Elle ne se substitue pas à la spécification des G-codes modaux du début de chaque fichier ngc. Les codes modaux des machines diffèrent, ils pourraient être modifiés par les G-codes interprétés plutôt dans la session.
- SUBROUTINE_PATH = ncsubroutines:/tmp/testsubs:lathesubs:millsubs - Spécifie une liste, séparée par (:) d'au maximum 10 répertoires dans lesquels seront cherchés les fichier de sous-programme spécifiés dans le g-code. Ces répertoires sont inspectés après que ne le soit [DISPLAY]PROGRAM_PREFIX (si il est spécifié) et avant que ne le soit [WIZARD]WIZARD_ROOT (si il est spécifié). les recherches s'effectuent dans l'ordre dans lequel les chemins sont listés. La première occurrence avec le sous-programme recherché est utilisée. Les répertoires sont spécifiés relativement au répertoire courant du fichier ini ou par des chemins absolus. La liste ne doit contenir aucun espace blanc.
- USER_M_PATH = myfuncs:/tmp/mcodes:experimentalmcodes - Spécifie une liste de répertoires, séparés par (:) (sans aucun espace blanc) pour les fonctions définies par l'utilisateur. Les répertoires sont spécifiés par rapport au répertoire courant pour les fichiers ini ou en chemins absolus. La liste ne doit contenir aucun espace blanc.

- USER_DEFINED_FUNCTION_MAX_DIRS=5 - Défini le nombre maximum de répertoires au moment de la compilation. Une recherche est faite pour chaque fonction utilisateur définie possible, typiquement M100 à M199.
  L'ordre de recherche est le suivant:

  1. [DISPLAY]PROGRAM_PREFIX (si il est spécifié)
  2. Si [DISPLAY]PROGRAM_PREFIX n'est pas spécifié, cherche dans le répertoire par défaut: nc_files
  3. Recherche ensuite dans chaque répertoire de la liste [RS274NGC]USER_M_PATH Le premier M1xx trouvé au cours de la recherche est utilisé pour chaque M1xx.

Note
[WIZARD]WIZARD_ROOT est un chemin de recherche valide mais l'assistant n'est pas encore complétement implémenté et les résultats, découlant de son utilisation, sont imprévisibles.

8.2.5.5 Section [EMCMOT]

D'autres entrées peuvent être rencontrées dans cette section, elles ne doivent pas être modifiées.

- EMCMOT = motmod - Utilise typiquement le nom du contrôleur de mouvement.
- BASE_PERIOD = 50000 - (HAL) Période de base des tâches, exprimée en ns.
- SERVO_PERIOD = 1000000 - (hal) Période de la tâche Servo, exprimée également en nanosecondes.
- TRAJ_PERIOD = 1000000 - (hal) Période du planificateur de trajectoire, exprimée en nanosecondes.

8.2.5.6 Section [TASK]

- TASK = milltask - Indique le nom de la tâche exécutable. La tâche réalise différentes actions, telles que communiquer avec les interfaces utilisateur au dessus de NML, communiquer avec le planificateur de mouvements temps réel dans la mémoire partagée non-HAL, et interpréter le g-code. Actuellement il n'y a qu'une seule tâche exécutable qui fait sens pour 99,9% des utilisateurs, milltask.
- CYCLE_TIME = 0.010 - Période exprimée en secondes, à laquelle TASK 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.

8.2.5.7 Section [HAL]

- TWOPASS=ON - Utilise le processus twopass (double passe) pour charger les composants HAL. Avec le processus TWOPASS, tous les fichiers [HAL]HALFILES sont premièrement lus et les occurrences multiples des directives à loadrt pour chaque module sont cumulées. Aucune commande HAL n'est exécutée à la première passe.
- 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 LinuxCNC s'arrête. Selon les pilotes de matériel utilisés, il est ainsi possible de positionner les sorties sur des valeurs définies quand LinuxCNC 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 dispositifs logiciels de protection des défauts de fonctionnement comme la pompe de charge ou le watchdog.
- POSTGUI_HALFILE = example2.hal - (Seulement avec les interfaces TOUCHY et AXIS) Exécute example2.hal après que l'interface graphique ait créé ses HAL pins.

8.2.5.8 Section [HALUI]

- MDI_COMMAND = G53 G0 X0 Y0 Z0 - Une commande MDI peut être exécuté en utilisant halui.mdi-command-00. Incrémente le nombre pour chaque commande énumérée dans la section [HALUI].

8.2.5.9 Section [TRAJ]

La section [TRAJ] contient les paramètres généraux du module planificateur de trajectoires de EMCMOT. Vous n'aurez pas à modifier ces valeurs si vous utilisez LinuxCNC 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 3, U est toujours l'articulation 6 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.
- JOINTS = 3 - (Cette variable de configuration est utilisée seulement par Axis et non par le planificateur de trajectoire du contrôleur de mouvement.) Elle spécifie le nombre d'articulations (moteurs) que comporte le système. Par exemple, une machine XYZ avec un seul moteur pour chacun des 3 axes, comporte 3 articulations (joints). Une machine à portique avec un seul moteur sur deux de ses axes et deux moteurs sur le troisième axe, comporte 4 articulations (joints).
- 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 axes (joins) sont enregistrées dans ce fichier. Cela permet donc de redémarrer avec les mêmes coordonnées que lors de l'arrêt, ce qui 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. Si vide, les positions ne seront pas enregistrées et commenceront à 0 à chaque fois que LinuxCNC démarrera.
- NO_FORCE_HOMING = 1 - LinuxCNC oblige implicitement l'utilisateur à référencer la machine par une prise d'origine machine avant de pouvoir lancer un programme ou exécuter une commande dans le MDI, seuls les mouvements de Jog sont autorisés avant les prises d'origines. Mettre NO_FORCE_HOMING = 1 permet à l'opérateur averti de s'affranchir de cette restriction de sécurité lors de la phase de mise au point de la machine.

Warning
NO_FORCE_HOMING mise à 1 permettra à la machine de franchir les limites logicielles pendant les mouvements ce qui n'est pas souhaitable pour un fonctionnement normal!

8.2.5.10 Sections [AXIS_n]

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

Généralement (mais pas toujours):

- AXIS_0 = X
- AXIS_1 = Y
- AXIS_2 = Z
- AXIS_3 = A
- AXIS_4 = B
- AXIS_5 = C
- AXIS_6 = U
- AXIS_7 = V
- AXIS_8 = W
  - TYPE = LINEAR - Type des axes, soit LINEAR, soit ANGULAR.
  - WRAPPED_ROTARY = 1 - Lorsque ce paramètre est réglé à 1 pour un axe angulaire l'axe se déplace de 0 à 359.999 degrés. Les nombres positifs déplacent l'axe dans le sens positif et les nombres négatifs dans le sens négatif.

- LOCKING_INDEXER = 1 - Quand ce paramètre est mis à 1, un mouvement en G0 sur cet axe va produire un signal de déblocage sur la pin axis.N.unlock, puis attendre le signal axis.N.is-unlocked de cet axe pour déplacer l'axe à la vitesse rapide prévue pour cet axe. Après ce mouvement, le signal axis.N.unlock retombera à false et les mouvements attendront que axis.N.is-unlocked redevienne false. Le mouvement des autres axes n'est pas autorisé lors du mouvement d'un axe rotatif à verrou.
- UNITS = inch - Ce réglage écrase celui des variables [TRAJ] UNITS si il est spécifié. (ex: [TRAJ]LINEAR_
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. Si un backlash est ajouté à un axe et que des moteurs pas à pas sont utilisés, la valeur de STEPGEN_MAXACCEL doit être 1.5 à 2 fois plus grande que celle de MAX_ACCELERATION pour cet axe.
- COMP_FILE = file.extension - Fichier dans lequel est enregistrée une structure de compensation spécifique à cet axe. Le fichier peut être nommé xscrew.comp, par exemple, pour l'axe X. Les noms de fichiers sont sensibles à la casse et peuvent contenir des lettres et/ou des chiffres. Les valeurs sont des triplets par ligne séparés par un espace. La première valeur est nominale (où elle devrait l'être). Les deuxième et troisième valeurs dépendront du réglage de COMP_FILE_TYPE. Actuellement la limite de LinuxCNC est de 256 triplets par axe. Si COMP_FILE est spécifié, BACKLASH est ignoré. Les valeurs sont en unités machine.
- COMP_FILE_TYPE = 0 ou 1 -
  * Si 0: Les deuxième et troisième valeurs spécifient la position en avant (de combien l'axe est en avance) et la position en arrière (de combien l'axe est en retard), positions qui correspondent à la position nominale.
  * Si 1: Les deuxième et troisième valeurs spécifient l'ajustement avant (à quelle distance de la valeur nominale lors d'un déplacement vers l'avant) et l'ajustement arrière (à quelle distance de la valeur nominale lors d'un déplacement vers l'arrière), positions qui correspondent à la position nominale.

Exemple de triplet avec COMP_FILE_TYPE = 0: 1.00 1.01 0.99
Exemple de triplet avec COMP_FILE_TYPE = 1: 1.00 0.01 -0.01

- MIN_LIMIT = -1000 - Limite minimale des mouvements de cet axe (limite logicielle), en unités machine. Quand cette limite tend à être dépassée, le contrôleur arrête le mouvement.
- MAX_LIMIT = 1000 - Limite maximale des mouvements de cet axe (limite logicielle), 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.
- FERROR = 1.0 - FERROR est le maximum d'erreur 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. L'utilitaire de calibration peut être utilisé pour les ajuster interactivement et vérifier les résultats, de sorte que les valeurs puissent être mises dans le fichier INI avec un minimum de difficultés. Cet utilitaire est accessible dans Axis depuis le menu Machine puis Calibration et dans TkLinuxCNC depuis le menu Réglages puis Calibration.

8.2.5.11 Section [HOMING]

Les paramètres suivants sont relatifs aux prises d'origine, pour plus d'informations, lire le chapitre sur la POM.

- HOME = 0.0 - La position à laquelle le mobile ira à la fin de la séquence de prise d'origine.
- HOME_OFFSET = 0.0 - Position du contact d'origine machine de l'axe ou de l'impulsion d'index, en unités machine. Lorsque le point d'origine est détecté pendant le processus de prise d'origine, c'est cette position qui est assignée à ce point. Dans le cas du partage de capteur entre l'origine et les limites d'axe et de l'utilisation d'une séquence de prise d'origine qui laisse le capteur dans l'état activé, la valeur de HOME_OFFSET peut être utilisée pour définir une position du capteur différente du 0 utilisé alors pour l'origine.
- HOME_SEARCH_VEL = 0.0 - Vitesse du mouvement initial de prise d'origine, en unités machine par seconde. Une valeur de zéro suppose que la position courante est l'origine machine. Si la machine n'a pas de contact d'origine, laisser cette valeur à zéro.
- HOME_LATCH_VEL = 0.0 - Vitesse du mouvement de dégagement du contact d'origine, en unités machine par seconde.
- HOME_FINAL_VEL = 0.0 - Vitesse du mouvement final entre le contact d'origine et la position d'origine, en unités machine par seconde. Si cette variable est laissée à 0 ou absente, la vitesse de déplacement rapide est utilisée. Doit avoir une valeur positive.
- 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, il est possible de mettre sur Yes. Quand il est sur yes, il aura une incidence sur le type de séquence de prise d'origine utilisée.
- HOME_IGNORE_LIMITS = NO - Si la machine utilise un seul et même contact comme limite d'axe et origine machine de l'axe. Cette variable devra alors être positionnée sur yes. Dans ce cas le contact de limite de cet axe est ignoré pendant la séquence de prise d'origines. Il est nécessaire de configurer la séquence pour qu'à la fin du mouvement le capteur ne reste pas dans l'état activé qui aboutirait finalement à un message d'erreur du capteur de limite.
- HOME_IS_SHARED = <n> - Si l'entrée du contact d'origine est partagée par plusieurs axes, mettre <n> à 0 pour permettre la POM même si un des contacts partagés est déjà attaqué. Le mettre à 1 pour interdire la prise d'origine dans ce cas.
- HOME_SEQUENCE = <n> - Utilisé pour définir l'ordre dans lequel les axes se succéderont lors d'une séquence de POM générale. **<n>** commence à 0, aucun numéro ne peut être sauté. Si cette variable est absente ou à -1, la POM de l'axe ne pourra pas être exécutée par la commande POM générale. La POM de plusieurs axes peut se dérouler simultanément.
- VOLATILE_HOME = 0 - Lorsqu'il est activé (mis à 1), l'origine machine de cette articulation sera effacée si la machine est en marche et que l'arrêt d'urgence est activé. Ceci est utile si la machine possède des contacts d'origine mais n'a pas de retour de position comme une machine à moteur pas à pas de type pas/direction.

8.2.5.12 Variables relatives aux servomoteurs

Les éléments suivants sont pour les systèmes à servomoteurs et à pseudos servomoteurs. Cette description suppose que les unités en sortie du composant PID sont des Volts.

- DEADBAND = 0.000015 - (dans HAL) Quelle distance est assez proche de la consigne pour considérer le moteur en position, en unités machine. Cette variable est fréquemment réglée pour une distance équivalente à 1, 1.5, 2, ou 3 impulsions de comptage du codeur, mais cela n'a rien d'une règle stricte. Un réglage lâche (large) permet de moins solliciter le servo au détriment de la précision. Un réglage serré (petit) permettra d'atteindre une grande précision mais le servo sera plus sollicité. Est-ce vraiment plus précis si c'est plus incertain ? En règle générale, il est préférable d'éviter le plus possible de solliciter le servo, si c'est possible.

Ayez la prudence de ne pas chercher à aller en dessous d'une impulsion de codeur, sinon vous enverrez votre servo quelque part où il ne sera pas heureux ! Cela peut arriver entre réglage lent et réglage nerveux et même un réglage impropre peut provoquer des couinements, des grincements dus aux oscillations provoquées par ce mauvais réglage. Il est préférable de perdre une ou deux impulsions au début des réglages, au moins jusqu'à avoir bien dégrossi les réglages.

Exemple de calcul en unités machine par top de codeur à utiliser pour décider de la valeur de DEADBAND (bande morte):

X pouces / top de codeur = 1 tour / 1000 top de codeur * 1 top de codeur / 4 top en quadrat
*** 0.2 pouce / tour = 0.200 pouce / 4000 top de codeur = 0.000050 pouce / top de codeur.**

- BIAS = 0.000 - (dans HAL) (Parfois appelé offset) il est utilisé par hm2-servo et quelques autres. Le Bias est une valeur constante qui est ajoutée sur la sortie. Dans la plupart des cas, elle peut rester à zéro. Toutefois, il peut être intéressant pour compenser un décalage de l'ampli du servo, ou équilibrer le poids d'un objet se déplaçant verticalement. Le bias est mis à zéro quand la boucle PID est désactivée, comme tous les autres composants de la sortie.
- P = 50 - (hal) La composante Proportionnelle 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, par exemple: **Volt/mm** si l'unité machine est le millimètre.
- I = 0 - (hal) La composante Intégrale 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: **Volt/mm*s** si l'unité machine est le millimètre.
- D = 0 - (hal) La composante Dérivée 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: **Volt/(mm/s)** si l'unité machine est le millimètre.
- FF0 = 0 - (hal) Gain à priori (retour vitesse) d'ordre 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: **Volt/mm** si l'unité machine est le millimètre.
- FF1 = 0 - (hal) Gain à priori (retour vitesse) de premier ordre. 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: **Volt/(mm/s)** si l'unité machine est le millimètre.
- FF2 = 0 - (hal) Gain à priori (retour vitesse) de second ordre. 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: **Volt/mm/s^2** si l'unité machine est le millimètre.

- OUTPUT_SCALE = 1.000 -
- OUTPUT_OFFSET = 0.000 - (hal) Ces deux valeurs sont les facteurs d'échelle et offset pour la sortie de l'axe à l'amplificateurs moteur. 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, LinuxCNC 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 = 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/s :
  amplifier [volts] = (output[mm/s] - offset[mm/s]) / 250mm/(s/Volt)

  Notez que les unités d'offset sont en unités machine, exemple: mm/s 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 normalement 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:

- Construire un tableau de calibrage pour la sortie, piloter le DAC avec la tension souhaitée et mesurer le résultat. Voir le tableau ci-dessous pour un exemple de mesures de tension.
- Par la méthode des moindres carrés, obtenir les coefficients **a**,**b** tels que: **mesure = a*raw+b**
- Noter que nous voulons des sorties brutes de sorte que nos résultats mesurés soient identiques à la sortie commandée. Ce qui signifie:
- **cmd = a*raw+b**
- **raw = (cmd-b)/a**
- En conséquence, les coefficients **a** et **b** d'ajustement linéaire peuvent être directement utilisés comme valeurs d'échelle et d'offset pour le contrôleur.

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

- 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 mise à l'échelle de la sortie en unités brutes. La valeur est appliquée de manière symétrique aux deux côtés, le positif et le négatif.
- INPUT_SCALE = 20000 - (hal) Spécifie le nombre d'impulsions qui correspond à un mouvement de une unité machine telle que fixée dans la section TRAJ. Pour un axe linéaire, une unité machine sera égale à la valeur de LINEAR_UNITS. Pour un axe angulaire, une unité machine sera égale à la valeur de ANGULAR_UNITS. Un second chiffre, si spécifié, sera ignoré. Par exemple, sur un codeur de 2000 impulsions par tour, un réducteur de 10 tours/pouce et des unités demandées en pouces, nous avons:
  INPUT_SCALE = 2000 top/tour * 10 tour/pouce = 20000 top/pouce

8.2.5.13 Variables relatives aux moteurs pas à pas

- SCALE = 4000 - (hal) Spécifie le nombre d'impulsions qui correspond à un mouvement d'une unité machine comme indiqué dans la section [TRAJ]. Pour les systèmes à moteurs pas à pas, c'est le nombre d'impulsions de pas nécessaires pour avancer d'une unité machine. Pour un axe linéaire, une unité machine sera égale à la valeur de LINEAR_UNITS. Pour un axe angulaire, une unité machine sera égale à la valeur de ANGULAR_UNITS. Pour les systèmes à servomoteurs, c'est le nombre d'impulsions de retour signifiant que le mobile a avancé d'une unité machine. Un second nombre, si spécifié, sera ignoré. Par exemple, un pas moteur de 1.8 degré, en mode demi-pas, avec une réduction de 10 tours/pouce et des unités souhaitées en pouces, nous avons:
  `scale = 2 pas/1.8 degrés * 360 degrés/tour * 10 tour/pouce = 4000 pas/pouce`

(D'anciens fichiers .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. Si une correction de jeu a été appliquée sur un axe, alors STEPGEN_MAXACCEL doit être 1,5 à 2 fois plus grande que MAX_ACCELERATION.
- STEPGEN_MAXVEL = 1.4 - (hal) Les anciens fichiers de configuration avaient également une limite de vitesse du générateur de pas. Si spécifiée, elle doit aussi être 1% à 10% supérieure à celle de l'axe MAX_VELOCITY. Des tests ultérieurs ont montré que l'utilisation de STEPGEN_MAXVEL n'améliore pas le réglage de la boucle de position de stepgen.

8.2.5.14 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 que EMCIO tournera en permanence. Il est préférable de ne pas modifier cette valeur.
- TOOL_TABLE = tool.tbl - Ce fichier contient les informations des outils, décrites dans le Manuel de l'utilisateur.
- TOOL_CHANGE_POSITION = 0 0 2 - Quand trois digits sont utilisés, spécifie la position XYZ ou le mobile sera déplacé pour le changement d'outil. Si six digits sont utilisés, spécifie l'emplacement ou sera envoyé le mobile pour réaliser le changement d'outil sur une machine de type XYZABC et de même, sur une machine de type XYZABCUVW lorsque 9 digits sont utilisés. Les variables relatives à la position du changement d'outil peuvent être combinées, par exemple; en combinant TOOL_CHANGE_POSITION avec TOOL_CHANGE_QUILL_UP il est possible de déplacer d'abord Z puis X et Y.
- TOOL_CHANGE_WITH_SPINDLE_ON = 1 - Avec cette valeur à 1, la broche reste en marche pendant le changement d'outil. Particulièrement utile sur les tours.
- TOOL_CHANGE_QUILL_UP = 1 - Avec cette valeur à 1, l'axe Z sera déplacé sur son origine machine avant le changement d'outil. C'est l'équivalent d'un G0 G53 Z0.
- TOOL_CHANGE_AT_G30 = 1 - Avec cette valeur à 1, le mobile sera envoyé sur un point de référence prédéfini par G30 dans les paramètres 5181-5186. Pour plus de détails sur les paramètres de G30, voir le chapitre relatif au G-code dans le Manuel de l'utilisateur.
- RANDOM_TOOLCHANGER = 1 - C'est pour des machines qui ne peuvent pas placer l'outil dans la poche il vient. Par exemple, les machines qui change l'outil dans la poche active avec l'outil dans la broche.

8.3 Prise d’origine

8.3.1 La prise d’origine

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 de LinuxCNC en conséquence. Toutefois, les machines étant différentes les unes des autres, la prise d’origine est maintenant devenue assez complexe.

8.3.2 Séquences de prise d’origine

Il existe quatre séquences de prise d’origine possibles. Elles sont définies par le signe des variables SEARCH_VEL et LATCH_VEL ainsi que les paramètres de configuration associés, la figure suivante donne le détail de ces séquences.

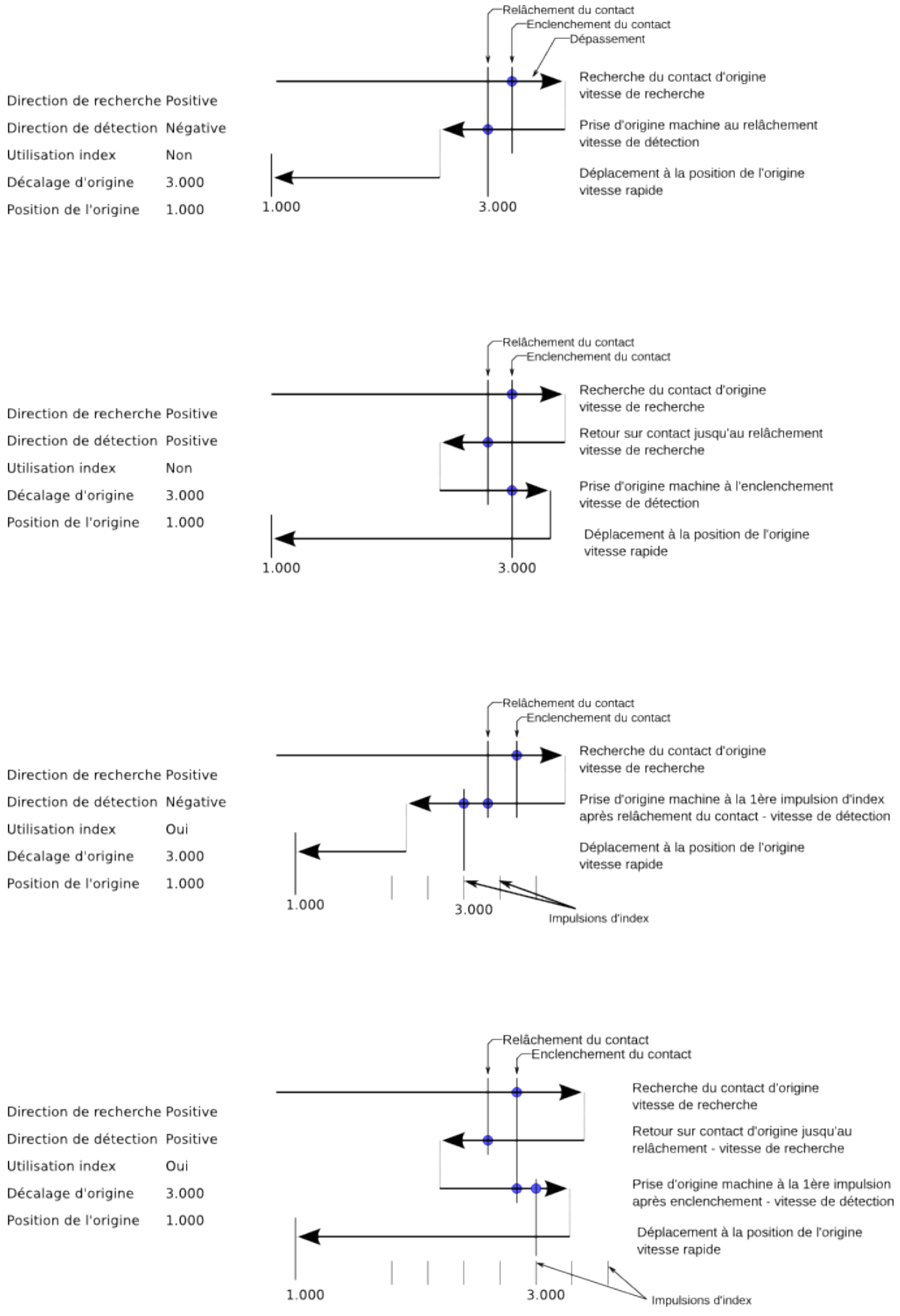


Figure 8.2: Les séquences de POM possibles

1. Comme on le voit sur la figure, les deux conditions de base sont les suivantes:
   a. La direction de recherche (SEARCH_VEL) et la direction de détection (LATCH_VEL) sont de même signe.
   b. La direction de recherche (SEARCH_VEL) et la direction de détection (LATCH_VEL) sont de signe opposé.

8.3.3 Configuration

Le tableau suivant détermine exactement comment se déroule la séquence de prise d'origines définie dans la section [AXIS] du fichier ini.

Table 8.1: Combinaisons des variables de la POM

Type de POM	SEARCH_VEL	LATCH_VEL	USE_INDEX
Immediate	0	0	NON
Index-seul	0	nonzero	OUI
Contact-seul	nonzero	nonzero	NO
Contact et Index	nonzero	nonzero	OUI

Note
Toute autre combinaison produira une erreur.

8.3.3.1 Vitesse de recherche (HOME_SEARCH_VEL)

Vitesse de la phase initiale de prise d'origine, pendant la recherche du contact d'origine machine. Une valeur différente de zéro indique à LinuxCNC la présence d'un contact d'origine machine. LinuxCNC va alors commencer par vérifier si ce contact est déjà attaqué. 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 dû à l'inertie et 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.

8.3.3.2 Vitesse de détection (HOME_LATCH_VEL)

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 de recherche 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 l'origine machine 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, LinuxCNC dégagera le contact à la vitesse définie ici. L'acquisition de l'origine machine 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, l'acquisition de l'origine machine se fera à cet position. Si HOME_SEARCH_VEL est différent de zéro et que HOME_LATCH_VEL est égale à zéro, c'est une cause d'erreur, l'opération de prise d'origine échouera. La valeur par défaut est zéro.

8.3.3.3 HOME_IGNORE_LIMITS

Peut contenir les valeurs YES ou NO. Cette variable détermine si LinuxCNC 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, LinuxCNC 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.

8.3.3.4 HOME_USE_INDEX

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), LinuxCNC fera l'acquisition de l'origine machine sur le premier front de l'impulsion d'index. Si elle est fausse (=NO), LinuxCNC 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.

8.3.3.5 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 du point d'origine ou de l'impulsion d'origine, LinuxCNC ajuste les coordonnées de l'axe à la valeur de HOME_OFFSET. La valeur par défaut est zéro.

8.3.3.6 Position de l'origine (HOME)

C'est la position sur laquelle ira le mobile à la fin de la séquence de prise d'origine machine. 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.

8.3.3.7 HOME_IS_SHARED

Si cet axe n'a pas un contact d'origine séparé des autres, mais plusieurs contacts câblés sur la même broche d'entrée, mettre 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.

8.3.3.8 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, leurs POM s'effectueront simultanément. Si HOME_SEQUENCE est égale à -1 ou n'est pas spécifiée, l'axe ne sera pas compris dans la séquence HOME_ALL. Les valeurs de HOME_SEQUENCE débutent à 0, il ne peut pas y avoir de valeur inutilisée.

8.3.3.9 VOLATILE_HOME

Si ce paramètre est vrai, l'origine machine de cet axe sera effacée chaque fois que la machine sera mise à l'arrêt. Cette variable est appropriée pour les axes ne maintenant pas la position si le moteur est désactivé (gravité de la broche par exemple). Certains moteurs pas à pas, en particulier fonctionnant en micropas, peuvent se comporter de la sorte.

8.3.3.10 LOCKING_INDEXER

Si cet axe comporte un verrouillage d'indexeur rotatif, celui-ci sera déverrouillé avant le début de la séquence de prise d'origine, et verrouillé à la fin.

8.4 Tours

8.4.1 Plan par défaut

Quand l'interpréteur de LinuxCNC à été créé, il à été écrit pour les fraiseuses. C'est pourquoi le plan par défaut est le plan XY (G17). Sur un tour standard on utilise seulement les axes du plan XZ (G18). Pour changer le plan par défaut d'un tour, mettez la ligne suivante dans la section RS274NGC du fichier ini.

```
RS274NGC_STARTUP_CODE = G18
```

La ligne de commande ci-dessus peut être remplacées par des G codes placés dans le préambule du programme G-code.

8.4.2 Réglages INI

Les réglages du fichier .ini suivants sont nécessaires dans Axis en mode tour, en remplacement ou ajout au fichier .ini normal.

```
[DISPLAY]
DISPLAY = axis
LATHE = 1
[TRAJ]
AXES = 3
COORDINATES = X Z
[AXIS_0]
...
[AXIS_2]
...
```

8.5 Fichiers TCL pour HAL

Le langage de halcmd excelle pour spécifier des composants et des connexions mais il n'offre aucune possibilité de calcul. Par conséquent, les fichiers ini sont limités en clarté et n'ont pas la concision qu'ils pourraient avoir avec un langage de haut niveau.

Haltcl fourni un moyen facile d'utiliser les scripts tcl et leurs fonctions pour les calculs, les boucles, les branchements, les procédures, etc dans les fichiers ini. Pour utiliser cette fonctionnalité, il est nécessaire d'utiliser le langage TCL ainsi que l'extension .tcl pour les fichiers de HAL.

L'extension .tcl est comprise par le script principal (linuxcnc) qui traite les fichiers ini. Les fichiers haltcl sont identifiés dans la section [HAL] des fichiers ini (comme les fichiers .hal).

Exemple

```
[HAL]
HALFILE = fichier_conventionnel.hal
HALFILE = fichier_tcl.tcl
```

Avec quelques précautions appropriées, les fichiers .hal et .tcl peuvent être mélangés.

8.5.1 Compatibilité

Le langage utilisé dans les fichiers .tcl a une syntaxe simple qui est un sous-ensemble du puissant et polyvalent langage de script Tcl.

8.5.2 Commandes haltcl

Les fichiers haltcl utilisent le langage de script Tcl, auquel s'ajoutent les commandes spécifiques de la couche d'abstraction matériel de LinuxCNC (HAL). Les commandes spécifiques sont les suivantes:

```
addf, alias,
delf, delsig,
getp, gets
ptype,
stype,
help,
linkpp, linkps, linksp, list, loadrt, loadusr, lock,
net, newsig,
save, setp, sets, show, source, start, status, stop,
unalias, unlinkp, unload, unloadrt, unloadusr, unlock,
waitusr
```

Il existe deux cas particuliers, les commandes gets et list en raison de conflits avec les commandes internes de Tcl. Pour haltcl, ces commandes doivent être précédées du mot clé hal, comme ci-dessous:

```
halcmd   haltcl
------   ------
gets     hal gets
list     hal list
```

8.5.3 Variables du fichier ini et haltcl

Les variables du fichier ini sont accessibles par halcmd comme par haltcl mais avec une syntaxe différente.

Les fichiers ini de LinuxCNC utilisent des spécificateurs SECTION et ITEM pour identifier les items de configuration:

```
[SECTION_A]
ITEM1 = value_1
ITEM2 = value_2
...
[SECTION_B]
...
```

Les valeurs du fichier ini sont accessibles par substitution de texte dans les fichiers .hal en utilisant la forme:

```
[SECTION]ITEM
```

Les mêmes valeurs de fichiers ini sont accessibles dans les fichiers .tcl sous la forme de tableau de variables globales.

```
$::SECTION(ITEM)
```

Par exemple, un item de fichier ini comme:

```
[AXIS_0]
MAX_VELOCITY = 4
```

est exprimé comme [AXIS_0]MAX_VELOCITY dans les fichiers .hal pour halcmd et comme $::AXIS_0(MAX_VE
dans les fichiers .tcl pour haltcl.

8.5.4 Conversion des fichiers .hal en fichiers .tcl

Les fichiers .hal existants peuvent être convertis manuellement en fichiers .tcl en les éditant pour adapter les différences mentionnées précédemment. Ce processus peut être automatisé à l'aide de scripts de conversion procédant à ces substitutions:

```
[SECTION]ITEM ---> $::SECTION(ITEM)
gets          ---> hal gets
list          ---> hal list
```

8.5.5 Notes à propos de haltcl

Dans haltcl, la valeur de l'argument pour les commandes sets et setp est implicitement traitée comme une expression dans le langage Tcl.

exemple

```
# set gain to convert deg/sec to units/min for AXIS_0 radius
setp scale.0.gain 6.28/360.0*$::AXIS_0(radius)*60.0
```

Les espaces blancs ne sont pas autorisés dans les expressions, utiliser des guillemets doubles pour s'en affranchir:

```
setp scale.0.gain "6.28 / 360.0 * $::AXIS_0(radius) * 60.0"
```

Dans d'autres contextes, tels que loadrt, il est nécessaire d'utiliser explicitement la commande Tcl expr ([expr {}]) pour des expressions de calcul.

exemple

```
loadrt motion base_period=[expr {500000000/$::TRAJ(MAX_PULSE_RATE)}]
```

8.5.6 Exemples pour haltcl

Prenons la question de la marge haute de stepgen (stepgen headroom). Le générateur de pas logiciel stepgen fonctionne mieux avec une contrainte d'accélération légèrement supérieure à celle du planificateur de mouvement. Ainsi, lorsqu'on utilise des fichiers halcmd, on force une valeur calculée manuellement dans le fichier ini.

```
[AXIS_0]
MAXACCEL = 10.0
STEPGEN_MAXACCEL = 10.5
```

Avec haltcl, il est possible d'utiliser des commandes Tcl pour effectuer le calcul et éliminer totalement l'item STEPGEN_MAXACCEL du fichier ini.

```
setp stepgen.0.maxaccel $::AXIS_0(MAXACCEL)*1.05
```

Autres caractéristiques de haltcl, les boucles et les tests. Par exemple, beaucoup de configurations utilisent les fichiers .hal core_sim.hal ou core_sim9.hal. Ceux-ci diffèrent du fait de la nécessité de connecter plus ou moins d'axes. Le code haltcl suivant devrait fonctionner pour n'importe quelle combinaison d'axes dans une machine à cinématique triviale (trivkins).

```
# Crée les signaux position, vitesse et accélération pour chaque axe
set ddt 0
foreach axis {X Y Z A B C U V W} axno {0 1 2 3 4 5 6 7 8} {
  # 'list pin' retourne une liste vide si la pin n'existe pas
  if {[hal list pin axis.$axno.motor-pos-cmd] == {}} {
    continue
  }
  net ${axis}pos axis.$axno.motor-pos-cmd => axis.$axno.motor-pos-fb \
                                         => ddt.$ddt.in
  net ${axis}vel <= ddt.$ddt.out
  incr ddt
  net ${axis}vel => ddt.$ddt.in
  net ${axis}acc <= ddt.$ddt.out
  incr ddt
}
puts [show sig *vel]
puts [show sig *acc]
```

8.5.7 Interactivité de haltcl

La commande halrun reconnaît les fichiers halctl. Avec l'option -T, haltcl peut être exécuté interactivement comme un interpréteur Tcl. Cette fonctionnalité est utile pour les tests et pour les applications hal autonomes.

exemple

```
$ halrun -T fichierhaltcl.tcl
```

8.5.8 Exemples pour haltcl fournis avec la distribution (sim)

Le répertoire configs/sim/axis/simtcl contient un fichier ini qui utilise un fichier .tcl pour démontrer une configuration haltcl en conjonction avec l'utilisation du processus "twopass". L'exemple montre l'utilisation des procédures Tcl, les boucles, l'utilisation des commentaires avec sortie sur le terminal.

8.6 Configuration d'un système pas/direction (dir/step)

8.6.1 Introduction

Ce chapitre décrit quelques uns des réglages les plus fréquents, sur lesquels l'utilisateur aura à agir lors de la mise au point de LinuxCNC. En raison de l'adaptabilité de LinuxCNC, il serait très difficile de les documenter tous en gardant ce document relativement concis.

Le système rencontré le plus fréquemment chez les utilisateurs de LinuxCNC est un système à moteurs pas à pas. Les interfaces de pilotage de ces moteurs recoivent de LinuxCNC des signaux de pas et de direction.

C'est le système le plus simple à mettre en oeuvre parce que les moteurs fonctionnent en boucle ouverte (pas d'information de retour des moteurs), le système nécessite donc d'être configuré correctement pour que les moteurs ne perdent pas de pas et ne calent pas.

Ce chapitre s'appuie sur la configuration fournie d'origine avec LinuxCNC appelée stepper et qui se trouve habituellement dans /etc/linuxcnc/sample-configs/stepper.

8.6.2 Fréquence de pas maximale

Avec la génération logicielle des pas la fréquence maximale en sortie, pour les impulsions de pas et de direction, est de une impulsion pour deux BASE_PERIOD. La fréquence de pas maximale accessible pour un axe est le produit de MAX_VELOCITY et de INPUT_SCALE. Si la fréquence demandée est excessive, une erreur de suivi se produira (following error), particulièrement pendant les jog rapides et les mouvements en G0.

Si votre interface de pilotage des moteurs accepte des signaux d'entrée en quadrature, utilisez ce mode. Avec un signal en quadrature, un pas est possible pour chaque BASE_PERIOD, ce qui double la fréquence maximum admissible.

Les autres remèdes consistent à diminuer une ou plusieurs variables: BASE_PERIOD (une valeur trop faible peux causer un bloquage du PC), INPUT_SCALE (s'il est possible sur l'interface de pilotage de sélectionner une taille de pas différente, de changer le rapport des poulies ou le pas de la vis mère), ou enfin MAX_VELOCITY et STEPGEN_MAXVEL.

Si aucune combinaison entre BASE_PERIOD, INPUT_SCALE et MAX_VELOCITY n'est fonctionnelle, il faut alors envisager un générateur de pas externe (parmis les contrôleurs de moteurs pas à pas universels supportés par LinuxCNC)

8.6.3 Brochage

LinuxCNC est très flexible et grâce à la couche d'abstraction de HAL (Hardware Abstraction Layer) il est facile de spécifier que tel signal ira sur telle broche.

Voir le tutoriel de HAL pour plus d'informations.

Comme décrit dans l'introduction et la manuel de HAL, il comporte des composants dont il fourni les signaux, les pins et les paramètres.

Les premiers signaux et pins relatifs au brochage sont:[5]

[5] Note: pour rester concis, nous ne présenterons qu'un seul axe, tous les autres sont similaires.

```
signaux: Xstep, Xdir et Xen
pins: parport.0.pin-XX-out & parport.0.pin-XX-in
```

Pour configurer le fichier ini, il est possible de choisir entre les deux brochages les plus fréquents, devenus des standards de fait, le brochage standard_pinout.hal ou le brochage xylotex_pinout.hal. Ces deux fichiers indiquent à HAL comment raccorder les différents signaux aux différentes pins. Dans la suite, nous nous concentrerons sur le brochage standard_pinout.hal.

8.6.4 Le fichier standard_pinout.hal

Ce fichier contient certaines commandes de HAL et habituellement ressemble à celà:

```
# standard pinout config file for 3-axis steppers
# using a parport for I/O
#

# first load the parport driver
loadrt hal_parport cfg="0x0378"

#
# next connect the parport functions to threads
# read inputs first
addf parport.0.read base-thread 1

# write outputs last
addf parport.0.write base-thread -1

#
# finally connect physical pins to the signals
net Xstep => parport.0.pin-03-out
net Xdir  => parport.0.pin-02-out
net Ystep => parport.0.pin-05-out
net Ydir  => parport.0.pin-04-out
net Zstep => parport.0.pin-07-out
net Zdir  => parport.0.pin-06-out

# create a signal for the estop loopback
net estop-loop iocontrol.0.user-enable-out iocontrol.0.emc-enable-in

# create signals for tool loading loopback
net tool-prep-loop iocontrol.0.tool-prepare iocontrol.0.tool-prepared
net tool-change-loop iocontrol.0.tool-change iocontrol.0.tool-changed

# connect "spindle on" motion controller pin to a physical pin
net spindle-on spindle.0.on => parport.0.pin-09-out

###
### You might use something like this to enable chopper drives when machine ON
### the Xen signal is defined in core_stepper.hal
###
# net Xen => parport.0.pin-01-out

###
### If you want active low for this pin, invert it like this:
###
# setp parport.0.pin-01-out-invert 1

###
### A sample home switch on the X axis (axis 0).  make a signal,
```

```
### link the incoming parport pin to the signal, then link the signal
### to LinuxCNC's axis 0 home switch input pin
###
# net Xhome parport.0.pin-10-in => axis.0.home-sw-in

###
### Shared home switches all on one parallel port pin?
### that's ok, hook the same signal to all the axes, but be sure to
### set HOME_IS_SHARED and HOME_SEQUENCE in the ini file.  See the
### user manual!
###
# net homeswitches <= parport.0.pin-10-in
# net homeswitches => axis.0.home-sw-in
# net homeswitches => axis.1.home-sw-in
# net homeswitches => axis.2.home-sw-in

###
### Sample separate limit switches on the X axis (axis 0)
###
# net X-neg-limit parport.0.pin-11-in => axis.0.neg-lim-sw-in
# net X-pos-limit parport.0.pin-12-in => axis.0.pos-lim-sw-in

###
### Just like the shared home switches example, you can wire together
### limit switches.  Beware if you hit one, LinuxCNC will stop but can't tell
### you which switch/axis has faulted.  Use caution when recovering from this.
###
# net Xlimits parport.0.pin-13-in => axis.0.neg-lim-sw-in axis.0.pos-lim-sw-in
```

Les lignes commençant par **#** sont des commentaires, aident à la lecture du fichier.

8.6.5 Vue d'ensemble du fichier standard_pinout.hal

Voici les opérations qui sont exécutées quand le fichier standard_pinout.hal est lu par l'interpréteur:

1. Le pilote du port parallèle est chargé (voir le Parport section de le Manuel de HAL pour plus de détails)
2. Les fonctions de lecture/écriture du pilote sont assignée au thread «Base thread» [6]
3. Les signaux du générateur de pas et de direction des axes X,Y,Z... sont raccordés aux broches du port parallèle
4. D'autres signaux d'entrées/sorties sont connectés (boucle d'arrêt d'urgence, boucle du changeur d'outil...)
5. Un signal de marche broche est défini et raccordé à une broche du port parallèle

8.6.6 Modifier le fichier standard_pinout.hal

Pour modifier le fichier standard_pinout.hal, il suffit de l'ouvrir dans un éditeur de texte puis d'y localiser les parties à modifier.

Si vous voulez par exemple, modifier les broches de pas et de direction de l'axe X, il vous suffit de modifier le numéro de la variable nommée parport.0.pin-XX-out:

[6] Le thread le plus rapide parmis les réglages de LinuxCNC, habituellement il n'y a que quelques microsecondes entre les exécutions de ce code.

```
net Xstep parport.0.pin-03-out
net Xdir  parport.0.pin-02-out
```

peut être modifiée pour devenir:

```
net Xstep parport.0.pin-02-out
net Xdir  parport.0.pin-03-out
```

ou de manière générale n'importe quel numéro que vous souhaiteriez.

Attention: il faut être certain de n'avoir qu'un seul signal connecté à une broche.

8.6.7 Modifier la polarité d'un signal

Si une interface attends un signal actif bas, ajouter une ligne avec le paramètre d'inversion de la sortie, -invert. Par exemple, pour inverser le signal de rotation de la broche:

```
setp parport.0.pin-09-invert TRUE
```

8.6.8 Ajouter le contrôle de vitesse broche en PWM

Si votre vitesse de broche peut être contrôlée par un signal de PWM, utilisez le composant pwmgen pour créer ce signal:

```
loadrt pwmgen output_type=0
addf pwmgen.update servo-thread
addf pwmgen.make-pulses base-thread
net spindle-speed-cmd spindle.N.speed-out => pwmgen.0.value
net spindle-on spindle.0.on => pwmgen.0.enable
net spindle-pwm pwmgen.0.pwm => parport.0.pin-09-out
setp pwmgen.0.scale 1800 # Change to your spindleb'''b''s top speed in RPM
```

Ce qui donnera le fonctionnement suivant, pour un signal PWM à: 0% donnera une vitesse de 0tr/mn, 10% une vitesse de 180tr/mn, etc. Si un signal PWM supérieur à 0% est requis pour que la broche commence à tourner, suivez l'exemple du fichier de configuration nist-lathe qui utilise un composant d'échelle (scale).

8.6.9 Ajouter un signal de validation enable

Certains pilotes de moteurs requiert un signal de validation enable avant d'autoriser tout mouvement du moteur. Pour celà des signaux sont déjà définis et appelés Xen, Yen, Zen.

Pour les connecter vous pouvez utilisez l'exemple suivant:

```
net Xen parport.0.pin-08-out
```

Il est possible d'avoir une seule pin de validation pour l'ensemble des pilotes, ou plusieurs selon la configuration que vous voulez. Notez toutefois qu'habituellement quand un axe est en défaut, tous les autres sont invalidés aussi de sorte que, n'avoir qu'un seul signal/pin de validation pour l'ensemble est parfaitement sécurisé.

8.6.10 Ajouter un bouton d’Arrêt d’Urgence externe

Comme vous pouvez le voir ici, par défaut la configuration standard n’utilise pas de bouton d’Arrêt d’Urgence externe. footnote:[Une explication complète sur la manière de gérer les circuiteries d’Arrêt d’Urgence se trouve sur le wiki(en) et dans le Manuel de l’intégrateur.

Pour ajouter un simple bouton d’AU externe (ou plusieurs en série) vous devez remplacer la ligne suivante:

```
net estop-loop iocontrol.0.user-enable-out iocontrol.0.emc-enable-in
```

par

```
net estop-loop parport.0.pin-01-in iocontrol.0.emc-enable-in
```

Ce qui implique qu’un bouton d’Arrêt d’Urgence soit connecté sur la broche 01 du port parallèle. Tant que le bouton est enfoncé (le contact ouvert)[7], LinuxCNC restera dans l’état Arrêt d’Urgence (ESTOP). Quand le bouton externe sera relâché, LinuxCNC passera immédiatement dans l’état Arrêt d’Urgence Relâché (ESTOP-RESET) vous pourrez ensuite mettre la machine en marche en pressant le bouton Marche machine et vous êtes alors prêt à continuer votre travail avec LinuxCNC.

[7] Utiliser exclusivement des contacts normalement fermés pour les A/U.

Chapter 9

Control Panels

9.1 PyVCP

9.1.1 Introduction

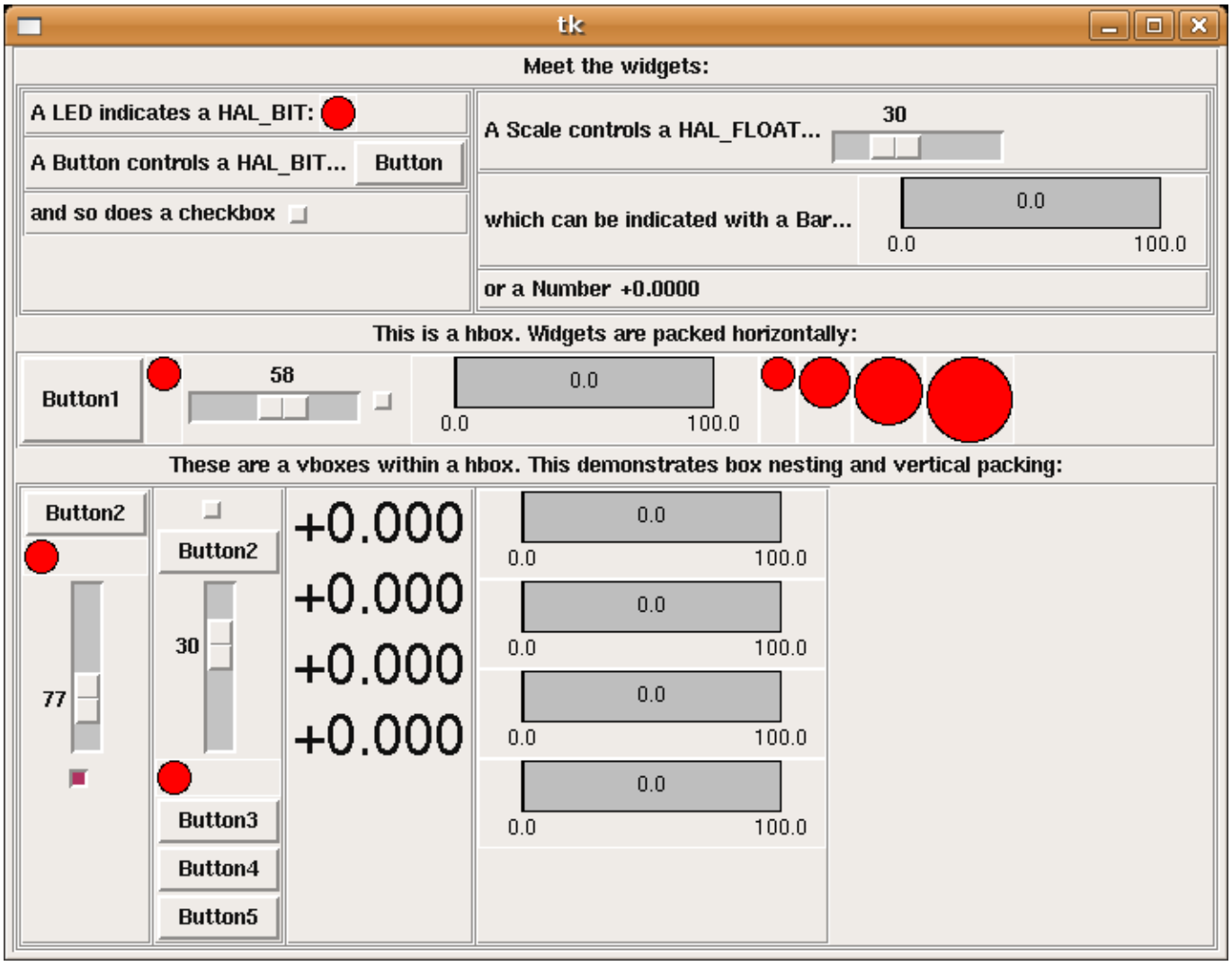
Panneau virtuel de contrôle en python (**Py**thon **V**irtual **C**ontrol **P**anel)

Le panneau de contrôle virtuel pyVCP a été créé pour donner à l'intégrateur la possibilité de personnaliser l'interface graphique d'AXIS avec des boutons et des indicateurs destinés aux tâches spéciales.

Le coût d'un panneau de contrôle physique est très élevé et il peut utiliser un grand nombre de broches d'entrées/sorties. C'est là que le panneau virtuel prends l'avantage car il ne coûte rien d'utiliser pyVCP.

Les panneaux de contrôle virtuels peuvent être utilisés pour tester ou monitorer le matériel, les entrées/sorties, remplacer temporairement d'autres matériels d'entrées/sorties pendant le déboguage d'une logique ladder ou pour simuler un panneau physique, avant de le construire et de le câbler vers les cartes électroniques.

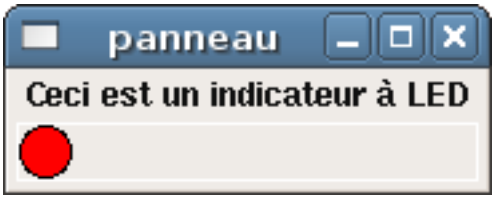
L'image suivante montre quelques widgets pyVCP.



9.1.2 Construction d'un panneau pyVCP

La disposition d'un panneau pyVCP est spécifiée dans un fichier XML qui contient les balises des widgets entre <pyvcp> et </pyvcp>. Par exemple:

```
<pyvcp>
   <label text="Ceci est un indicateur à LED"/>
   <led/>
</pyvcp>
```



Si vous placez ce texte dans un fichier nommé tiny.xml et que vous le lancez avec:

```
pyvcp -c panneau tiny.xml
```

pyVCP va créer le panneau pour vous, il y inclut deux widgets, un Label avec le texte Ceci est un indicateur à LED et une LED rouge, utilisée pour afficher l'état d'un signal HAL de type BIT. Il va aussi créer un composant HAL nommé panneau (tous les widgets dans ce panneau sont connectés aux pins qui démarrent avec panneau). Comme aucune balise <halpin> n'était présente à l'intérieur de la balise <led>, pyVCP nomme automatiquement la pin HAL pour le widget LED panneau.led.0

Pour obtenir la liste des widgets, leurs balises et options, consultez la documentation des widgets.

Un fois que vous avez créé votre panneau, connectez lui les signaux HAL, vers et à partir des pins pyVCP et avec la commande habituelle:

```
net <signal-name> <pin-name> <opt-direction> <opt-pin-name>signal-name
```

Si vous débutez avec HAL, le tutoriel de HAL est vivement recommandé.

9.1.3 Sécurité avec pyVCP

Certaines parties de pyVCP sont évaluées comme du code Python, elles peuvent donc exécuter n'importe quelle action disponible dans les programmes Python. N'utilisez que des fichiers pyVCP en .xml à partir d'une source de confiance.tag

9.1.4 Utiliser pyVCP avec AXIS

Puisque AXIS utilise le même environnement graphique et les même outils (Tkinter) que pyVCP, il est possible d'inclure un panneau pyVCP sur le côté droit de l'interface utilisateur normale d'AXIS. Un exemple typique est présenté ci-dessous.

Placer le fichier pyVCP XML décrivant le panneau dans le même répertoire que le fichier .ini. Nous voulons afficher la vitesse courante de la broche sur un widget barre de progression. Copier le code XML suivant dans un fichier appelé broche.xml:

```
<pyvcp>
    <label>
        <text>"Vitesse broche:"</text>
    </label>
    <bar>
        <halpin>"spindle-speed"</halpin>
        <max_>5000</max_>
    </bar>
</pyvcp>
```

Ici nous avons fait un panneau avec un label Vitesse broche: et un widget barre de progression. Nous avons spécifié que la pin HAL connectée à la barre de progression devait s'appeler spindle-speed et régler la valeur maximum de la barre à 5000 (se reporter à la documentation des widgets, pour toutes les options disponibles). Pour faire connaître ce fichier à AXIS et qu'il l'appelle au démarrage, nous devons préciser ce qui suit dans la section [DISPLAY] du fichier .ini:

```
PYVCP = broche.xml
```

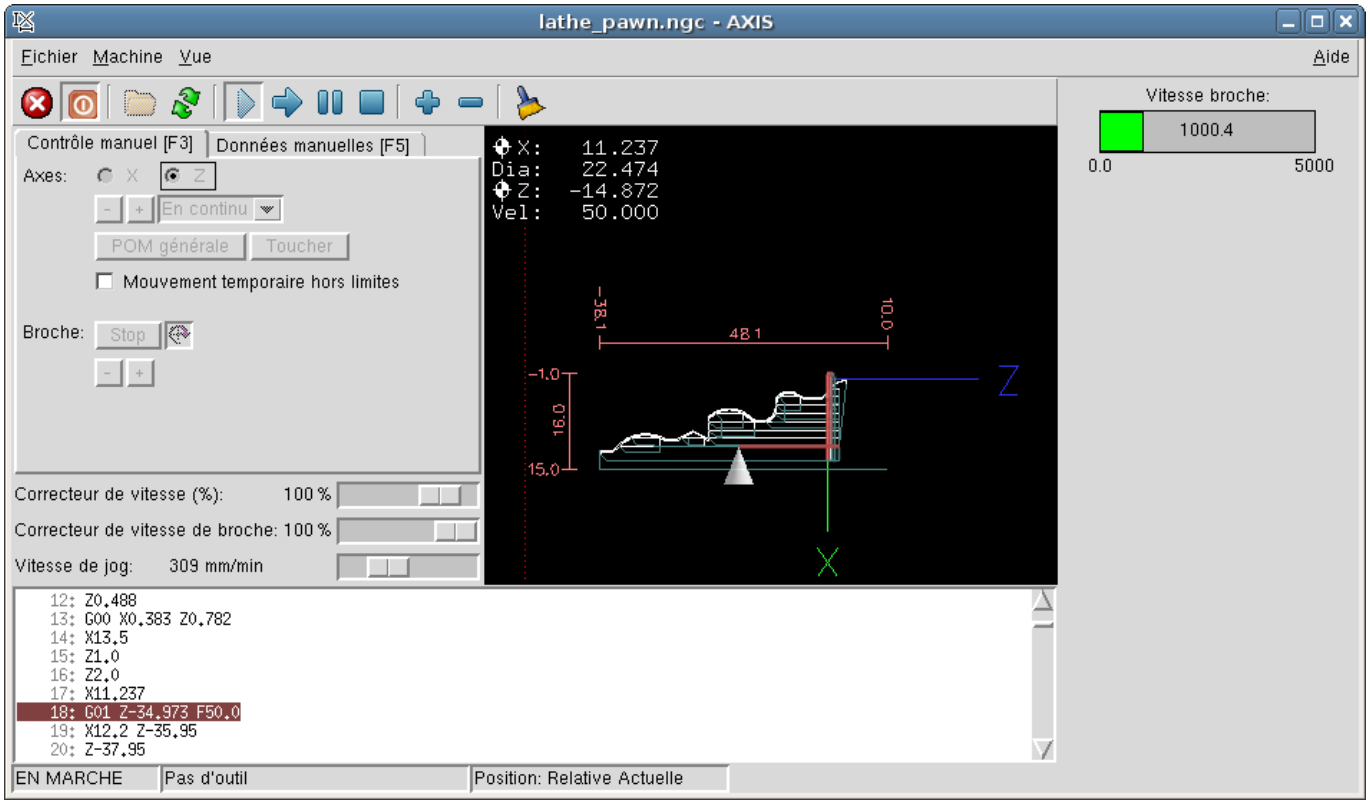
Pour que notre widget affiche réellement la vitesse de la broche spindle-speed, il doit être raccordé au signal approprié de HAL. Le fichier .hal qui sera exécuté quand AXIS et pyVCP démarreront doit être spécifié, de la manière suivante, dans la section [HAL] du fichier .ini:

```
POSTGUI_HALFILE = broche_vers_pyvcp.hal
```

Ce changement lancera la commande HAL spécifiée dans broche_vers_pyvcp.hal. Dans notre exemple, ce fichier contiendra juste la commande suivante:

```
net spindle-rpm-filtered  => pyvcp.spindle-speed
```

ce qui suppose que le signal appelé spindle-rpm-filtered existe aussi. Noter que lors de l'exécution avec AXIS, toutes les pins des widgets de pyVCP ont des noms commençant par pyvcp..



Voila à quoi ressemble le panneau pyVCP que nous venons de créer, incorporé à AXIS. La configuration sim/lathe fournie en exemple, est configurée de cette manière.

9.1.5 Panneaux PyVCP autonomes

Cette section va décrire comment les panneaux PyVCP peuvent être affichés par eux même, par l'intermédiaire ou non des contrôleurs machine de LinuxCNC.

Pour charger un panneau PyVCP autonome avec LinuxCNC utiliser cette commande:

```
loadusr -Wn monpanneau pyvcp -g WxH+X+Y -c monpanneau <path/>fichier_panneau.xml
```

Vous l'utiliserez pour avoir un panneau flottant ou un panneau avec une interface graphique autre que Axis.

- -Wn monpanneau - Fait attendre à HAL que le composant monpanneau soit chargé (devienne ready en langage HAL), avant d'exécuter d'autres commandes HAL. C'est important parce-que les panneaux PyVCP exportent des pins de HAL ainsi que d'autres composants de HAL qui doivent être présents pour pouvoir se connecter à eux. Noter la lettre **W** en majuscule et la lettre **n** en minuscule. Si vous utilisez l'option -Wn vous devez également utiliser l'option -c pour nommer le panneau.
- pyvcp < -g> < -c> panneau.xml - Construit le panneau avec la géométrie optionnelle et/ou le nom de panneau depuis le fichier panneau.xml. Le fichier panneau.xml peut avoir n'importe quel nom avec l'extension .xml. Le fichier .xml décrit comment construire le panneau. Il est nécessaire d'ajouter le nom du chemin si le panneau n'est pas dans le répertoire dans lequel se trouve le script HAL.

- -g <WxH><+X+Y> - Spécifie la géométrie à utiliser quand le panneau est construit. La syntaxe est Largeur x Hauteur + Ancrage X + Ancrage Y. La taille ou la position, ou les deux peuvent être fixés. Le point d'ancrage est le coin supérieur gauche du panneau. Par exemple; -g 250x500+800+0 fixe le panneau à 250 pixels de large, 500 pixels de haut avec le point d'ancrage placé en X800 Y0.
- -c nompanneau - Indique à PyVCP quel composant appeler et le titre de la fenêtre. Le nom du fichier nompanneau peut être n'importe quel nom sans espace.

Pour charger un panneau PyVCP autonome, sans LinuxCNC utiliser cette commande:

```
loadusr -Wn monpanneau pyvcp -g 250x500+800+0 -c monpanneau monpanneau.xml
```

La commande minimale pour charger un panneau pyvcp est la suivante:

```
loadusr pyvcp monpanneau.xml
```

Vous pourrez utiliser cette commande si vous voulez un panneau sans passer par un des contrôleurs machine de LinuxCNC, par exemple pour des tests ou une visu autonome.

La commande loadusr est utilisée quand vous chargez aussi un composant qui stoppera HAL depuis la fermeture jusqu'à ce qu'il soit prêt. Si vous avez chargé un panneau puis chargé Classic Ladder en utilisant la commande loadusr -w classicladder, CL maintiendra HAL et le panneau ouverts jusqu'à ce que vous fermiez Classic Ladder. Le -Wn signifie d'attendre que le composant -Wn "nom" devienne prêt. (nom peut être n'importe quel nom. Noter la lettre **W** en majuscule et le **n** en minuscule.) Le -c indique à PyVCP de construire un panneau avec le nom monpanneau en utilisant les infos contenues dans le fichier monpanneau.xml. Le nom du fichier monpanneau.xml est sans importante mais doit porter l'extension .xml. C'est le fichier qui décrit comment construire le panneau. Il est nécessaire d'ajouter le nom du chemin si le panneau n'est pas dans le répertoire dans lequel se trouve le script HAL.

Une commande optionnelle à utiliser si vous voulez que le panneau stoppe HAL depuis les commandes Continuer / Quitter. Après avoir chargé n'importe quelles autres composants la dernière commande HAL sera:

```
waituser nompanneau
```

Cette commande indique à HAL d'attendre que le composant nompanneau soit fermé avant de continuer avec d'autres commandes. C'est généralement défini comme étant la dernière commande, de sorte que HAL s'arrêtera si le panneau est fermé.

9.1.6 Documentation des widgets de pyVCP

Les signaux de HAL existent en deux variantes, BIT et FLOAT. pyVCP peut afficher la valeur d'un signal avec un widget indicateur, ou modifier la valeur d'un signal avec un widget de contrôle. Ainsi, il y a quatre classes de widgets pyVCP connectables aux signaux de HAL. Une cinquième classe de widgets d'aide permet d'organiser et d'appliquer des labels aux panneaux.

- Widgets de signalisation, signaux de type bit: led, rectled
- Widgets de contrôle, signaux de type bit: button, checkbutton, radiobutton
- Widgets de signalisation de type nombre: number, s32, u32, bar, meter
- Widgets de contrôle de type nombre: spinbox, scale, jogwheel
- Widgets d'aide: hbox, vbox, table, label, labelframe

9.1.6.1 Syntaxe

Chaque widget sera décrit brièvement, suivi par la forme d'écriture utilisée et d'une capture d'écran. Toutes les balises contenues dans la balise du widget principal, sont optionnelles.

9.1.6.2 Notes générales

Á l'heure actuelle, les deux syntaxes, basée sur les balises et basée sur les attributs, sont supportées. Par exemple, les deux fragments de code XML suivants sont traités de manière identique:

```
<led halpin="ma-led"/>
```

et

```
<led><halpin>"ma-led"</halpin></led>
```

Quand la syntaxe basée sur les attributs est utilisée, les règles suivantes sont utilisées pour convertir les valeurs des attributs en valeurs Python:

1. Si le premier caractère de l'attribut est un des suivants: {([" ' , il est évalué comme une expression Python.
2. Si la chaine est acceptée par int(), la valeur est traitée comme un entier.
3. Si la chaine est acceptée par float(), la valeur est traitée comme un flottant.
4. Autrement, la chaine est acceptée comme une chaine.

Quand la syntaxe basée sur les balises est utilisée, le texte entre les balises est toujours évalué comme un expression Python.

Les exemples ci-dessous montrent un mélange des deux formats.

9.1.6.3 Commentaires

Pour ajouter un commentaire utiliser la syntaxe de xml.

```
<!-- Mon commentaire -->
```

9.1.6.4 Editer un fichier XML

Editer le fichier XML avec un éditeur de texte. La plupart du temps un double click sur le nom de fichier permet de choisir ouvrir avec l'editeur de texte ou similaire.

9.1.6.5 Couleurs

Les couleurs peuvent être spécifiées en utilisant les couleurs RGB de X11 soit par le nom, par exemple: gray75 ou soit en hexa décimal, par exemple: #0000ff. Une liste complète est consultable ici: http://sedition.com/perl/rgb.html.

Couleurs les plus courantes (les numéros suivant la couleur indiquent la nuance de la couleur)

- white (blanc)
- black (noir)

- blue et blue1 - blue4 (bleu)
- cyan et cyan1 - cyan4 (cyan)
- green et green1 - green4 (vert)
- yellow et yellow1 - yellow4 (jaune)
- red et red1 - red4 (rouge)
- purple et purple1 - purple4 (violet/pourpre)
- gray et gray0 - gray100 (gris)

9.1.6.6 Pins de HAL

Les pins de HAL fournisse le moyen de connecter les widgets aux autres éléments. Quand une pin de HAL est créée pour un widget, il est possible de la connecter à une autre pin de HAL avec une commande net dans un fichier .hal. Pour plus de détails, voir la commande net dans le manuel de HAL.

9.1.6.7 Label

Un label est un texte qui s'affiche sur le panneau.

Le label a une pin optionnelle de désactivation en ajoutant: <disable_pin>True</disable_pin>.

```
<label>
    <text>”Ceci est un label:”</text>
    <font>(”Helvetica”,20)</font>
</label>
```

Ce code produira:

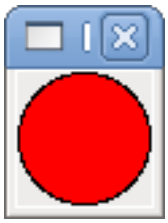
9.1.6.8 Les leds

Une led est utilisée pour indiquer l'état d'une pin de HAL de type bit. La couleur de la led sera on_color quand le signal est vrai et off_color autrement. * <halpin> définit le nom de la pin, par défaut: led.n, où n est un entier. * <size> définit la taille de la led, par défaut: 20. * <on_color> définit la couleur de la led led quand la pin est vraie, par défaut: green * <off_color> définit la couleur de la led quand la pin est fausse, par défaut: ref

9.1.6.9 La led ronde

```
<led>
    <halpin>”ma-led”</halpin>
    <size>50</size>
    <on_color>”verte”</on_color>
    <off_color>”rouge”</off_color>
</led>
```

Le résultat du code ci-dessus.



9.1.6.10 La led rectangulaire

C'est une variante du widget led.

```
<vbox>
    <relief>RIDGE</relief>
    <bd>6</bd>
    <rectled>
        <halpin>"ma-led-rect"</halpin>
        <height>"50"</height>
        <width>"100"</width>
        <on_color>"green"</on_color>
        <off_color>"red"</off_color>
    </rectled>
</vbox>
```

Le code ci-dessus produit cette led, entourée d'un relief.

9.1.6.11 Le bouton (button)

Un bouton permet de contrôler une pin de type bit. La pin sera mise vraie quand le bouton sera pressé et maintenu enfoncé, elle sera mise fausse quand le bouton sera relâché.

Les boutons peuvent suivre les options de formatage suivantes:

- <padx>n</padx> où n est le nombre d'espaces horizontaux supplémentaires
- <pady>n</pady> où n est le nombre d'espaces verticaux supplémentaires
- <activebackground>"color"</activebackground> Couleur au survol du curseur
- <bg>"color"</bg> Couleur du bouton

```
<button>
    <halpin>"Bouton-OK"</halpin>
    <text>"OK"</text>
</button>
<button>
    <halpin>"Bouton-Abandon"</halpin>
    <text>"Abort"</text>
</button
```

Le code ci-dessus produit:

Une case à cocher contrôle une pin de type bit. La pin sera mise vraie quand la case est cochée et fausse si la case est décochée.

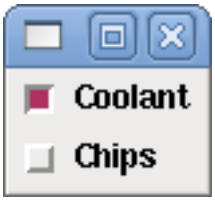
Une case non cochée:

et une case cochée:

Exemple de code:

```
<checkbutton>
    <halpin>"coolant-chkbtn"</halpin>
    <text>"Coolant"</text>
</checkbutton>
<checkbutton>
    <halpin>"chip-chkbtn"</halpin>
    <text>"Chips    "</text>
</checkbutton>
```

Le code ci-dessus produit:



Un bouton radio placera une seule des pins vraie. Les autres seront mises fausses.

```
<radiobutton>
    <choices>["un","deux","trois"]</choices>
    <halpin>"mon-radiobtn"</halpin>
</radiobutton>
```

Le code ci-dessus donne ce résultat:

Noter que dans l'exemple ci-dessus, les pins de HAL seront nommées mon-radiobtn.un, mon-radiobtn.deux et mon-radiobtn.trois. Dans l'image précédente, trois est la valeur sélectionnée courante.

9.1.6.12 Affichage d’un nombre (number)

L’affichage d’un nombre peux recevoir les options de formatage suivantes:

- (”Font Name”,n) où n est la taille de la police
- <width>n</width> où n est la largeur totale utilisée
- <justify>pos</justify> où ”pos” peut être LEFT, CENTER ou RIGHT (devrait marcher)
- <padx>n</padx> où ”n” est le nombre d’espaces horizontaux supplémentaires
- <pady>n</pady> où ”n” est le nombre d’espaces verticaux supplémentaires

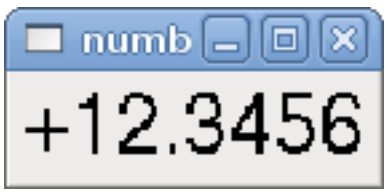
Le widget number affiche la valeur d’un signal de type flottant.

```
<number>
    <halpin>”number”</halpin>
    <font>(”Helvetica”,24)</font>
    <format>”+4.4f”</format>
</number>
```

Le code ci-dessus donne ce résultat:

Le widget number affiche la valeur d’un signal de type flottant.

```
<number>
    <halpin>”my-number”</halpin>
    <font>(”Helvetica”,24)</font>
    <format>”+4.4f”</format>
</number>
```



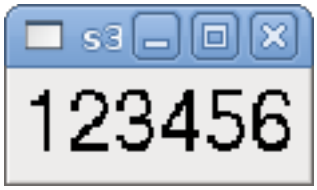
<font> est une police de caractères de type Tkinter avec la spécification de sa taille. Une police qui peut être agrandie jusqu’à la taille 200 est la police courier 10 pitch, que vous pouvez spécifier de la manière suivante, pour afficher des chiffres réellement grands:

```
<font>(’courier 10 pitch’,100)</font>
```

<format> est un format style C, spécifié pour définir le format d’affichage du nombre.

Le widget s32 affiche la valeur d’un nombre s32. La syntaxe est la même que celle de number excepté le nom qui est <s32>. Il faut prévoir une largeur suffisante pour afficher le nombre dans sa totalité.

```
<s32>
    <halpin>”simple-number”</halpin>
    <font>(”Helvetica”,24)</font>
    <format>”6d”</format>
    <width>6</width>
</s32>
```



Le widget u32 affiche la valeur d’un nombre u32. La syntaxe est la même que celle de number excepté le nom qui est <u32>.

9.1.6.13 Affichage d’images

Seul l’affichage d’images au format gif est possible. Toutes les images doivent avoir la même taille. Les images doivent être toutes dans le même répertoire que le fichier ini (ou dans le répertoire courant pour un fonctionnement en ligne de commande avec halrun/halcmd).

La bascule image_bit bascule entre deux images selon la position vraie ou fausse de halpin.

```
<pyvcp>
    <image name=’fwd’ file=’fwd.gif’/>
    <image name=’rev’ file=’rev.gif’/>
    <vbox>
        <image_bit halpin=’selectimage’ images=’fwd rev’/>
    </vbox>
</pyvcp>
```

En utilisant les deux images fwd.gif et rev.gif. FWD est affiché quand selectimage est fausse et REV est affiché quand selectimage est vraie.

Figure 9.1: selectimage fausse

Figure 9.2: selectimage vraie

La bascule image_u32 est la même que image_bit excepté que le nombre d'images n'est pratiquement plus limité, il suffit de selectionner l'image en ajustant halpin à une valeur entière commençant à 0 pour la première image de la liste, à 1 pour la seconde image etc.

```
<pyvcp>
    <image name='stb' file='stb.gif'/>
    <image name='fwd' file='fwd.gif'/>
    <image name='rev' file='rev.gif'/>
    <vbox>
        <image_u32 halpin='selectimage' images='stb fwd rev'/>
    </vbox>
</pyvcp>
```

Même résultat mais en ajoutant l'image stb.gif.

Figure 9.3: Halpin = 0

Figure 9.4: Halpin = 1

Figure 9.5: Halpin = 2

9.1.6.14 Barre de progression (bar)

Le widget barre de progression affiche la valeur d'un signal FLOAT, graphiquement dans une barre de progression et simultanément, en numérique.

```
<bar>
    <halpin>"bar"</halpin>
    <min_>0</min_>
    <max_>123</max_>
    <bgcolor>"grey"</bgcolor>
    <fillcolor>"red"</fillcolor>
</bar>
```

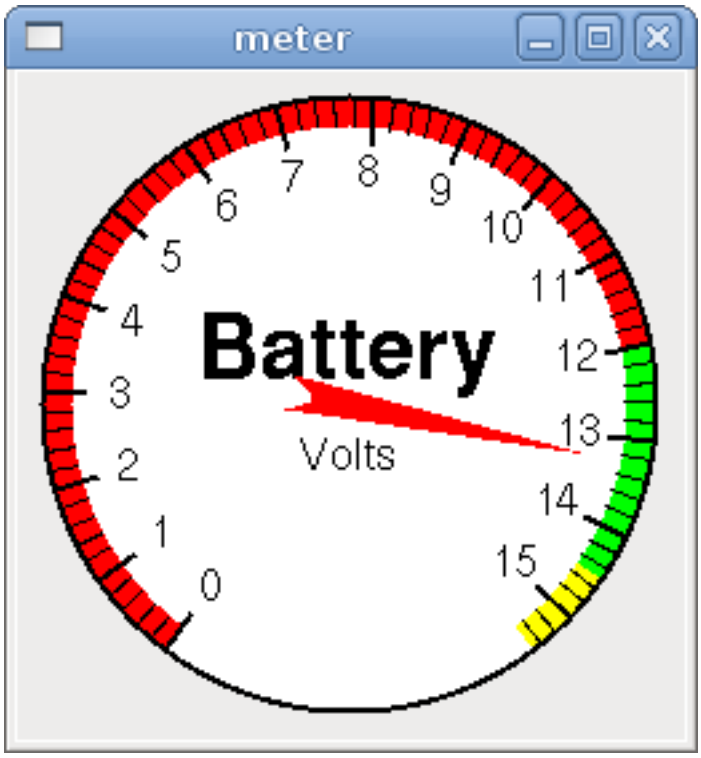
Le code ci-dessus donne ce résultat:

9.1.6.15 Galvanomètre (meter)

Le galvanomètre affiche la valeur d'un signal FLOAT dans un affichage à aiguille à l'ancienne.

```
<meter>
    <halpin>"mymeter"</halpin>
    <text>"Battery"</text>
    <subtext>"Volts"</subtext>
    <size>250</size>
    <min_>0</min_>
    <max_>15.5</max_>
    <majorscale>1</majorscale>
    <minorscale>0.2</minorscale>
    <region1>(14.5,15.5,"yellow")</region1>
    <region2>(12,14.5,"green")</region2>
    <region3>(0,12,"red")</region3>
</meter>
```

Le code ci-dessus donne ce résultat:



9.1.6.16 Boîte d'incrément (spinbox)

La boîte d'incrément contrôle une pin FLOAT. La valeur de la pin est augmentée ou diminuée de la valeur de resolution, à chaque pression sur une flèche, ou en positionnant la souris sur le nombre puis en tournant la molette de la souris.

```
<spinbox>
    <halpin>"my-spinbox"</halpin>
    <min_>-12</min_>
    <max_>33</max_>
    <inival>0</inival>
    <resolution>0.1</resolution>
    <format>"2.3f"</format>
    <font>("Arial",30)</font>
</spinbox>
```

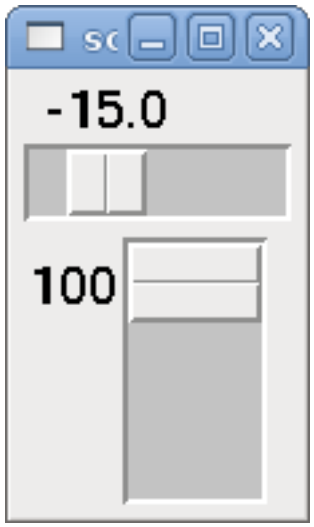
Le code ci-dessus donne ce résultat:

9.1.6.17 Curseur (scale)

Le curseur contrôle une pin FLOAT. La valeur de la pin est augmentée ou diminuée en déplaçant le curseur, ou en positionnant la souris sur le curseur puis en tournant la molette de la souris.

```
<scale>
    <font>("Helvetica",16)</font>
    <width>"25"</width>
    <halpin>"my-hscale"</halpin>
    <resolution>0.1</resolution>
    <orient>HORIZONTAL</orient>
    <initval>-15</initval>
    <min_>-33</min_>
    <max_>26</max_>
</scale>
<scale>
    <font>("Helvetica",16)</font>
    <width>"50"</width>
    <halpin>"my-vscale"</halpin>
    <resolution>1</resolution>
    <orient>VERTICAL</orient>
    <min_>100</min_>
    <max_>0</max_>
</scale>
```

Le code ci-dessus donne ce résultat:



Noter que par défaut c'est min qui est affiché même si il est supérieur à max, à moins que min ne soit négatif.

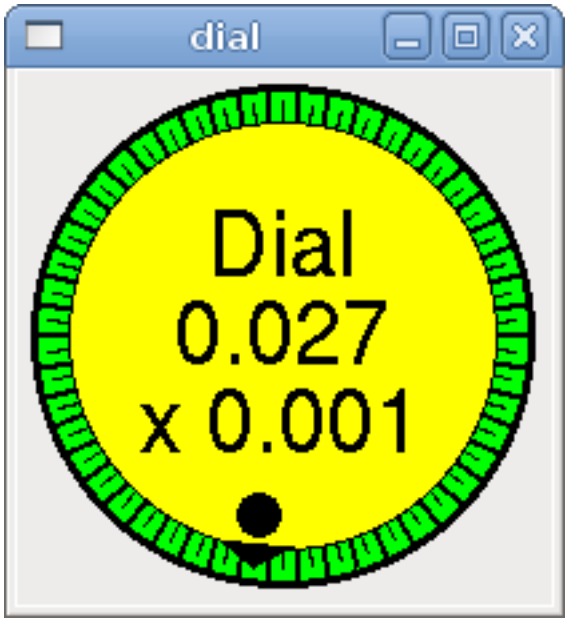
9.1.6.18 Bouton tournant (dial)

Le bouton tournant imite le fonctionnement d'un vrai bouton tournant, en sortant sur un FLOAT HAL la valeur sur laquelle est positionné le curseur, que ce soit en le faisant tourner avec un mouvement circulaire, ou en tournant la molette de la souris. Un double click gauche augmente la résolution et un double click droit la diminue d'un digit. La sortie est limitée par les valeurs min et max. La variable cpr fixe le nombre de graduations sur le pourtour du cadran (prudence avec les grands nombres).

```
<dial>
    <size>200</size>
    <cpr>100</cpr>
    <min_>-15</min_>
    <max_>15</max_>
    <text>"Dial"</text>
    <initval>0</initval>
```

```
    <resolution>0.001</resolution>
    <halpin>"anaout"</halpin>
    <dialcolor>"yellow"</dialcolor>
    <edgecolor>"green"</edgecolor>
    <dotcolor>"black"</dotcolor>
</dial>
```

Le code ci-dessus donne ce résultat:

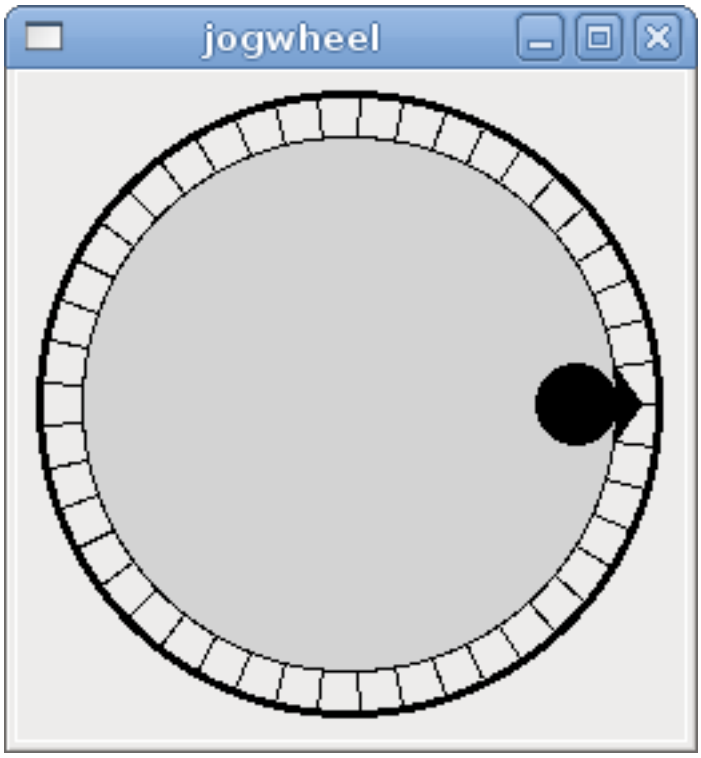


9.1.6.19 Manivelle (jogwheel)

La manivelle imite le fonctionnement d'une vraie manivelle, en sortant sur une pin FLOAT la valeur sur laquelle est positionné le curseur, que ce soit en le faisant tourner avec un mouvement circulaire, ou en tournant la molette de la souris.

```
<jogwheel>
    <halpin>"my-wheel"</halpin>
    <cpr>45</cpr>
    <size>250</size>
</jogwheel>
```

Le code ci-dessus donne ce résultat:



9.1.7 Documentation des containers de pyVCP

Les containers sont des widgets qui contiennent d'autres widgets.

9.1.7.1 Bordures

Le container bordure est spécifié avec deux balises utilisées ensembles. La balise <relief> spécifie le type de bordure et la balise <bd> spécifie la largeur de la bordure.

<relief>type</relief>
La valeur de type peut être: FLAT, SUNKEN, RAISED, GROOVE, ou RIDGE

<bd>n</bd>
La valeur de **n** fixe la largeur de la bordure.

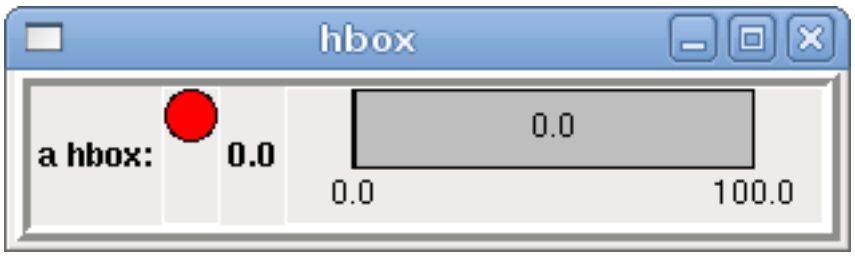
```
<hbox>
    <button>
        <relief>FLAT</relief>
        <text>"FLAT"</text>
        <bd>3</bd>
    </button>

    <button>
        <relief>SUNKEN</relief>
        <text>"SUNKEN"</text>
        <bd>3</bd>
    </button>

    <button>
        <relief>RAISED</relief>
        <text>"RAISED"</text>
```

```
        <bd>3</bd>
    </button>

    <button>
        <relief>GROOVE</relief>
        <text>"GROOVE"</text>
        <bd>3</bd>
    </button>

    <button>
        <relief>RIDGE</relief>
        <text>"RIDGE"</text>
        <bd>3</bd>
    </button>
</hbox>
```

9.1.7.2 Hbox

Utilisez une Hbox lorsque vous voulez aligner les widgets, horizontalement, les uns à côtés des autres.

```
<hbox>
    <relief>RIDGE</relief>
    <bd>6</bd>
    <label><text>"a hbox:"</text></label>
    <led></led>
    <number></number>
    <bar></bar>
</hbox>
```

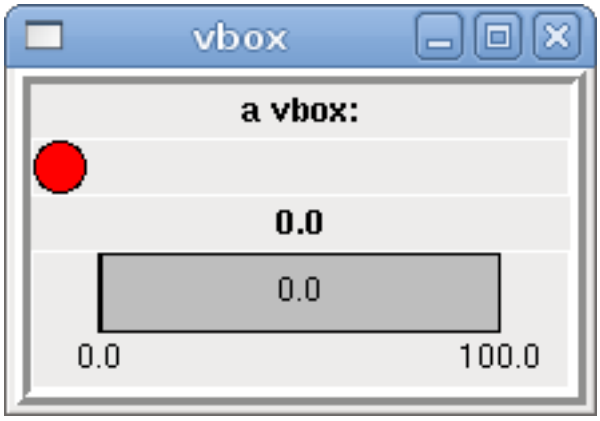


Á l'intérieur d'une Hbox, il est possible d'utiliser les balises <boxfill fill=/>, <boxanchor anchor=/> et <boxexpand expand=/> pour choisir le comportement des éléments contenus dans la boîte, lors d'un redimensionnement de la fenêtre. Pour des détails sur le comportement de fill, anchor, et expand, référez vous au manuel du pack Tk, pack(3tk). Valeurs par défaut, fill='y', anchor='center', expand='yes'.

9.1.7.3 Vbox

Utilisez une Vbox lorsque vous voulez aligner les widgets verticalement, les uns au dessus des autres.

```
<vbox>
    <relief>RIDGE</relief>
    <bd>6</bd>
    <label><text>"a vbox:"</text></label>
    <led></led>
    <number></number>
    <bar></bar>
</vbox>
```



Á l'intérieur d'une Vbox, vous pouvez utiliser les balises <boxfill fill=/>, <boxanchor anchor=/> et <boxexpand expand=/> pour choisir le comportement des éléments contenus dans la boîte, lors d'un redimensionnement de la fenêtre. Pour des détails sur le comportement de fill, anchor, et expand, référez vous au manuel du pack Tk, pack(3tk). Valeurs par défaut, fill='y', anchor='center', expand='yes'.

9.1.7.4 Labelframe

Un labelframe est un cadre entouré d'un sillon et un label en haut à gauche.

```
<labelframe text="Label: Leds groupées">
```

```
<labelframe text="Label: Leds groupées">
    <font>("Helvetica",16)</font>
    <hbox>
    <led/>
    <led/>
    <led/>
    </hbox>
</labelframe>
```

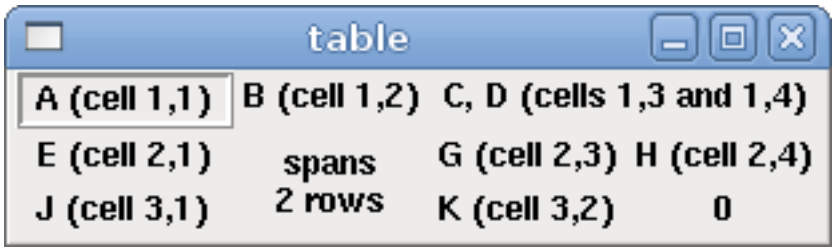
9.1.7.5 Table

Une table est un container qui permet d'écrire dans une grille de lignes et de colonnes. Chaque ligne débute avec la balise <tablerow/> . Un widget contenu peut être en lignes ou en colonnes par l'utilisation de la balise <tablespan rows= cols=/>. Les bordures des cellules contenant les widgets

sticky peuvent être réglées grâce à l'utilisation de la balise <tablesticky sticky=/>. Une table flexible peut s'étirer sur ses lignes et ses colonnes (sticky).

Exemple:

```
<table flexible_rows="[2]" flexible_columns="[1,4]">
<tablesticky sticky="new"/>
<tablerow/>
    <label>
        <text>" A (cell 1,1) "</text>
        <relief>RIDGE</relief>
        <bd>3</bd>
    </label>
    <label text="B (cell 1,2)"/>
    <tablespan columns="2"/>
    <label text="C, D (cells 1,3 and 1,4)"/>
<tablerow/>
    <label text="E (cell 2,1)"/>
    <tablesticky sticky="nsew"/>
    <tablespan rows="2"/>
    <label text="'spans\n2 rows'"/>
    <tablesticky sticky="new"/>
    <label text="G (cell 2,3)"/>
    <label text="H (cell 2,4)"/>
<tablerow/>
    <label text="J (cell 3,1)"/>
    <label text="K (cell 3,2)"/>
    <u32 halpin="test"/>
</table>
```

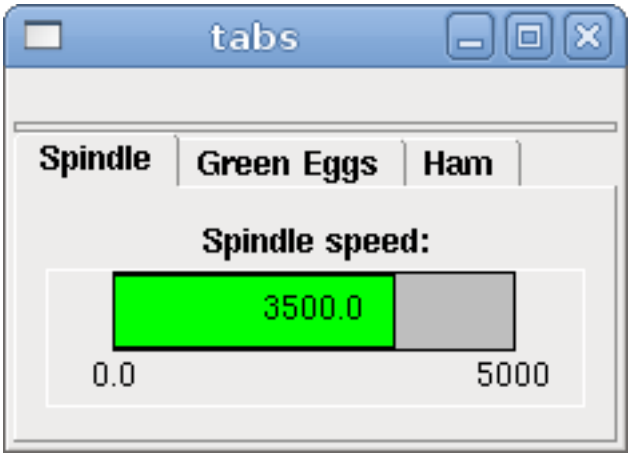


9.1.7.6 Onglets (Tabs)

Une interface à onglets permet d'économiser l'espace en créant un container pour chaque nom d'onglet (tabs). Une seule section tabs peut exister, les tabs ne peuvent pas être imbriqués ni empilés. La largeur de l'onglet le plus large, determine la largeur des onglets.

```
<tabs>
    <names>["Spindle", "Green Eggs", "Ham"]</names>
    <vbox>
        <label>
            <text>"Spindle speed:"</text>
        </label>
        <bar>
            <halpin>"spindle-speed"</halpin>
            <max_>5000</max_>
        </bar>
    </vbox>
    <vbox>
        <label>
            <text>"(this is the green eggs tab)"</text>
        </label>
```

```
    </vbox>
    <vbox>
        <label>
            <text>"(this tab has nothing on it)"</text>
        </label>
    </vbox>
</tabs>
```



9.2 Exemples d'utilisation de PyVCP

9.2.1 Panneau PyVCP dans AXIS

Procédure pour créer un panneau PyVCP et l'utiliser, attaché dans la partie droite de l'interface AXIS.

- Créer un fichier .xml contenant la description du panneau et le placer dans le répertoire de la configuration.
- Ajouter une entrée, avec le nom du fichier .xml, dans la section [DISPLAY] du fichier ini.
- Ajouter une entrée POSTGUI_HALFILE, avec le nom du fichier postgui HAL.
- Ajouter les liens vers les pins de HAL pour le panneau dans le fichier postgui.hal pour "connecter" le panneau PyVCP avec LinuxCNC.

9.2.2 Panneaux flottants

Pour créer des panneaux flottants PyVCP pouvant être utilisés avec n'importe quelle interface, suivre les points suivants:

- Créer un fichier .xml contenant la description du panneau et le placer dans le répertoire de configuration.
- Ajouter les lignes loadusr dans le fichier.hal pour charger chaque panneau.
- Ajouter les liens vers les pins de HAl du panneau dans le fichier postgui.hal, pour "connecter" les panneaux PyVCP à LinuxCNC.

L'exemple suivant montre le chargement de deux panneaux PyVCP.

```
loadusr -Wn btnpanel pyvcp -c btnpanel panel1.xml
loadusr -Wn sppanel pyvcp -c sppanel panel2.xml
```

Les paramètres -Wn font que HAL **W**ait for **n**ame, attends le composant nommé btnpanel.

Les paramètres pyvcp -c font que PyVCP nomme le panneau.

Les pins de HAL de panel1.xml seront nommées btnpanel.<pin name>

Les pins de HAL de panel2.xml seront nommées sppanel.<pin name>

Bien s'assurer qu'aucune ligne loadusr ne fasse déjà appel à une de ces pins PyVCP.

9.2.3 Boutons de Jog

Dans cet exemple nous allons créer un panneau PyVCP avec 3 boutons utilisables pour déplacer en manuel les axes X, Y et Z. Cette configuration sera réalisée avec l'assistant Stepconf qui générera la configuration de la machine. Premièrement, nous lançons l'assistant Stepconf et le configurons pour la machine, ensuite dans la page Advanced Configuration Options nous effectuons les sélections pour ajouter un panneau PyVCP vierge, comme indiqué sur l'image suivante. Pour cet exemple nous avons nommé la configuration pyvcp_xyz sur la page Basic Machine Information de l'assistant Stepconf.

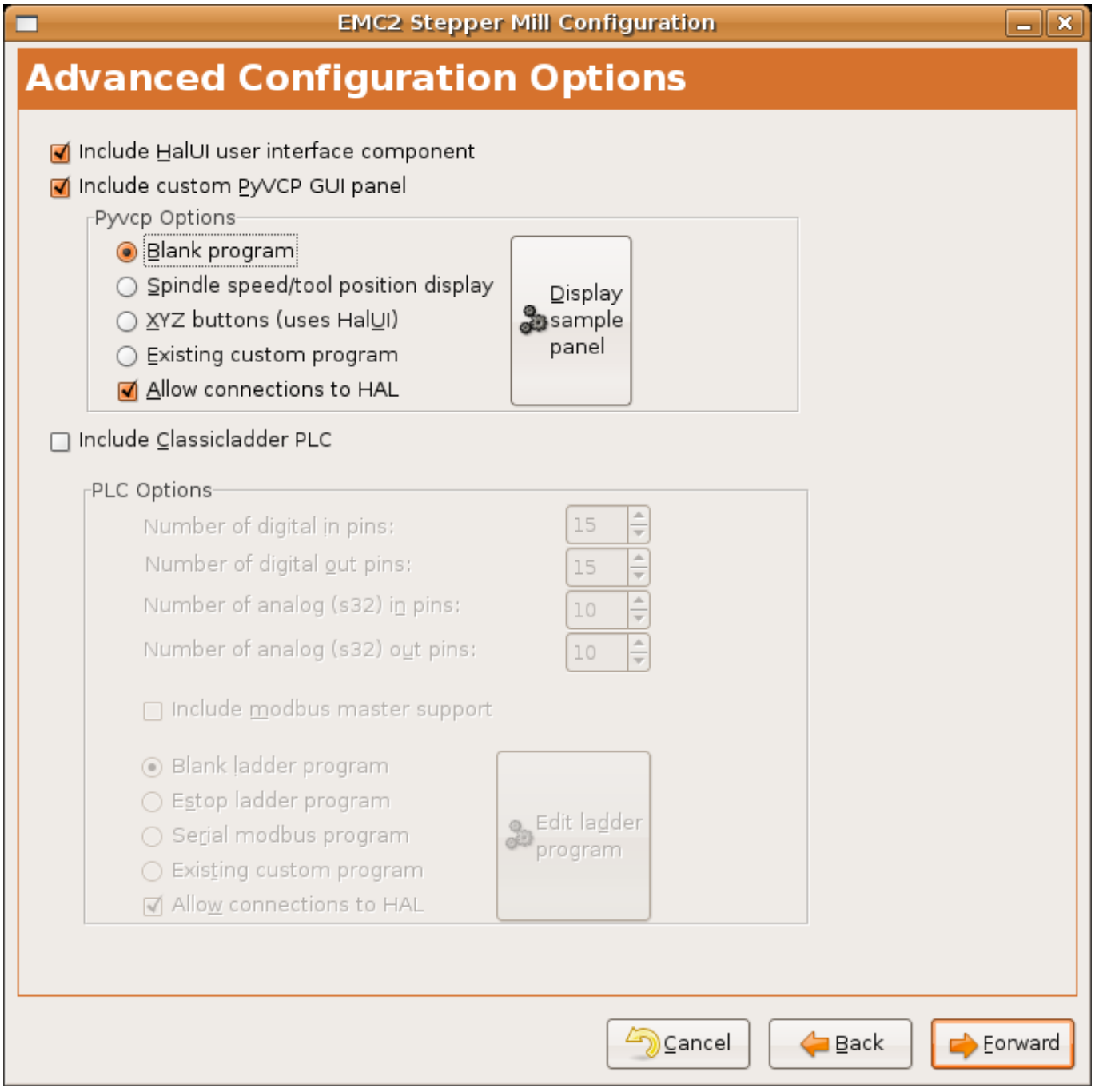


Figure 9.6: Assistant de configuration XYZ

L'assistant Stepconf Wizard va créer plusieurs fichiers et les placer dans le répertoire /emc/configs/pyvcp_xyz. Si la case Créer un lien est cochée, un lien vers ces fichiers sera créé sur le bureau.

9.2.3.1 Créer les Widgets

Ouvrir le fichier custompanel.xml par un clic droit sur son nom, puis en sélectionnant Ouvrir avec l'éditeur de texte. Entre les balises <pyvcp> et </pyvcp> nous ajouterons les widgets pour le panneau.

Tous les détails sur chacun des Widgets de PyVCP sont donnés dans la documentation des widgets.

Dans le fichier custompanel.xml nous ajoutons la description des widgets.

```
<pyvcp>

    <labelframe text="Boutons de Jog">
       <font>("Helvetica",16)</font>

        <!-- le bouton de jog de l'axe X -->
        <hbox>
            <relief>RAISED</relief>
            <bd>3</bd>
            <button>
                <font>("Helvetica",20)</font>
                <width>3</width>
                <halpin>"x-plus"</halpin>
                <text>"X+"</text>
            </button>
            <button>
                <font>("Helvetica",20)</font>
                <width>3</width>
                <halpin>"x-moins"</halpin>
                <text>"X-"</text>
            </button>
        </hbox>

        <!-- le bouton de jog de l'axe Y -->
        <hbox>
            <relief>RAISED</relief>
            <bd>3</bd>
            <button>
                <font>("Helvetica",20)</font>
                <width>3</width>
                <halpin>"y-plus"</halpin>
                <text>"Y+"</text>
            </button>
            <button>
                <font>("Helvetica",20)</font>
                <width>3</width>
                <halpin>"y-moins"</halpin>
                <text>"Y-"</text>
            </button>
        </hbox>

        <!-- le bouton de jog de l'axe Z -->
        <hbox>
            <relief>RAISED</relief>
            <bd>3</bd>
            <button>
                <font>("Helvetica",20)</font>
                <width>3</width>
                <halpin>"z-plus"</halpin>
                <text>"Z+"</text>
            </button>
            <button>
                <font>("Helvetica",20)</font>
                <width>3</width>
                <halpin>"z-moins"</halpin>
                <text>"Z-"</text>
            </button>
        </hbox>

        <!-- le curseur de vitesse de jog -->
```

```
        <vbox>
            <relief>RAISED</relief>
            <bd>3</bd>
            <label>
                <text>”Vitesse de Jog”</text>
                <font>(”Helvetica”,16)</font>
            </label>
            <scale>
                <font>(”Helvetica”,14)</font>
                <halpin>”jog-speed”</halpin>
                <resolution>1</resolution>
                <orient>HORIZONTAL</orient>
                <min_>0</min_>
                <max_>80</max_>
            </scale>
        </vbox>
    </labelframe>
</pyvcp>
```

Après les ajouts précédents, nous avons un panneau PyVCP tel que celui de l’image suivante, attaché à droite d’Axis. Il est beau mais ne fait rien tant que les boutons ne sont pas ”connectés” à halui. Si, à ce stade, une erreur se produit lors du déplacement de la fenêtre vers le bas, c’est généralement dû à une erreur de syntaxe ou d’écriture, elle est donc dans cette partie qu’il conviendra tout d’abord de vérifier soigneusement.

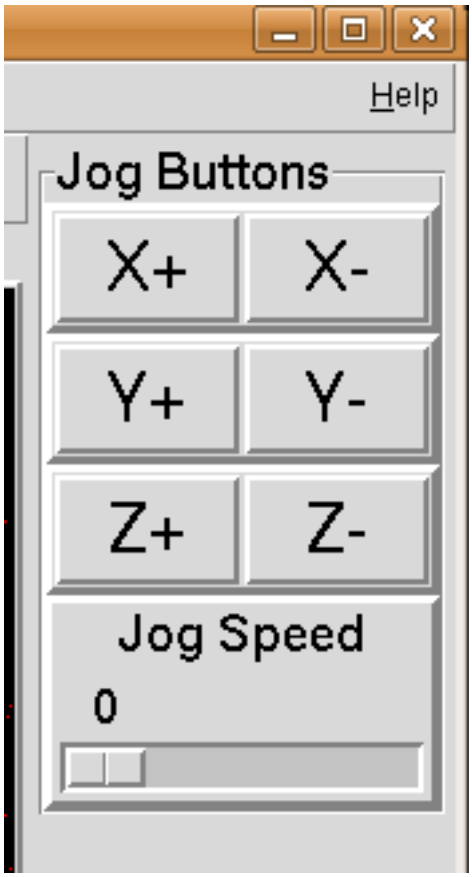


Figure 9.7: Boutons de Jog

9.2.3.2 Effectuer les connections

Pour effectuer les connections nécessaires, ouvrir le fichier custom_postgui.hal et y ajouter le code suivant:

```
# connecte les boutons PyVCP pour X
net my-jogxmoins halui.jog.0.minus <= pyvcp.x-moins
net my-jogxplus halui.jog.0.plus <= pyvcp.x-plus

# connecte les boutons PyVCP pour Y
net my-jogymoins halui.jog.1.minus <= pyvcp.y-moins
net my-jogyplus halui.jog.1.plus <= pyvcp.y-plus

# connecte les boutons PyVCP pour Z
net my-jogzmoins halui.jog.2.minus <= pyvcp.z-moins
net my-jogzplus halui.jog.2.plus <= pyvcp.z-plus

# connecte le curseur de vitesse de jog PyVCP
net my-jogspeed halui.jog-speed <= pyvcp.jog-speed-f
```

Après avoir désactivé l'A/U (E-Stop) et activé la marche machine en mode Jog, le déplacement du curseur du panneau PyVCP devrait agir dès qu'il est placé au delà de zéro et les boutons de jog devraient fonctionner. Il est impossible de jogger alors qu'un fichier G-code s'exécute ou pendant qu'il est en pause ni quand l'onglet Données manuelles [F5] du (MDI), est ouvert.

9.2.4 Testeur de port

Cet exemple montre comment faire un simple testeur de port parallèle en utilisant PyVCP et HAL.

Premièrement, créer le fichier ptest.xml qui contiendra le code suivant pour créer la description du panneau.

```
<!-- Panneau de test pour la config. du port parallèle -->
<pyvcp>
  <hbox>
    <relief>RIDGE</relief>
    <bd>2</bd>
    <button>
      <halpin>"btn01"</halpin>
      <text>"Pin 01"</text>
    </button>
    <led>
      <halpin>"led-01"</halpin>
      <size>25</size>
      <on_color>"green"</on_color>
      <off_color>"red"</off_color>
    </led>
  </hbox>
  <hbox>
    <relief>RIDGE</relief>
    <bd>2</bd>
    <button>
      <halpin>"btn02"</halpin>
      <text>"Pin 02"</text>
    </button>
    <led>
      <halpin>"led-02"</halpin>
      <size>25</size>
      <on_color>"green"</on_color>
      <off_color>"red"</off_color>
```

```
    </led>
  </hbox>
  <hbox>
    <relief>RIDGE</relief>
    <bd>2</bd>
    <label>
      <text>"Pin 10"</text>
      <font>("Helvetica",14)</font>
    </label>
    <led>
      <halpin>"led-10"</halpin>
      <size>25</size>
      <on_color>"green"</on_color>
      <off_color>"red"</off_color>
    </led>
  </hbox>
  <hbox>
    <relief>RIDGE</relief>
    <bd>2</bd>
    <label>
      <text>"Pin 11"</text>
      <font>("Helvetica",14)</font>
    </label>
    <led>
      <halpin>"led-11"</halpin>
      <size>25</size>
      <on_color>"green"</on_color>
      <off_color>"red"</off_color>
    </led>
  </hbox>
</pyvcp>
```

Le panneau flottant contenant deux pins de HAL d'entrée et deux pins de HAL de sortie.

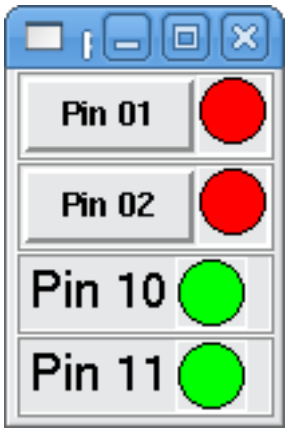


Figure 9.8: Panneau flottant testeur de port parallèle

Pour lancer les commandes de HAL dont nous avons besoin et démarrer tout ce qi'il nous faut, nous avons mis le code suivant dans notre fichier ptest.hal.

```
loadrt hal_parport cfg="0x378 out"
loadusr -Wn ptest pyvcp -c ptest ptest.xml
loadrt threads name1=porttest period1=1000000
addf parport.0.read porttest
addf parport.0.write porttest
```

```
net pin01 ptest.btn01 parport.0.pin-01-out ptest.led-01
net pin02 ptest.btn02 parport.0.pin-02-out ptest.led-02
net pin10 parport.0.pin-10-in ptest.led-10
net pin11 parport.0.pin-11-in ptest.led-11
start
```

Pour lancer le fichier HAL, nous utilisons, dans un terminal, les commandes suivantes:

```
~$ halrun -I -f ptest.hal
```

La figure suivante montre à quoi ressemble le panneau complet.

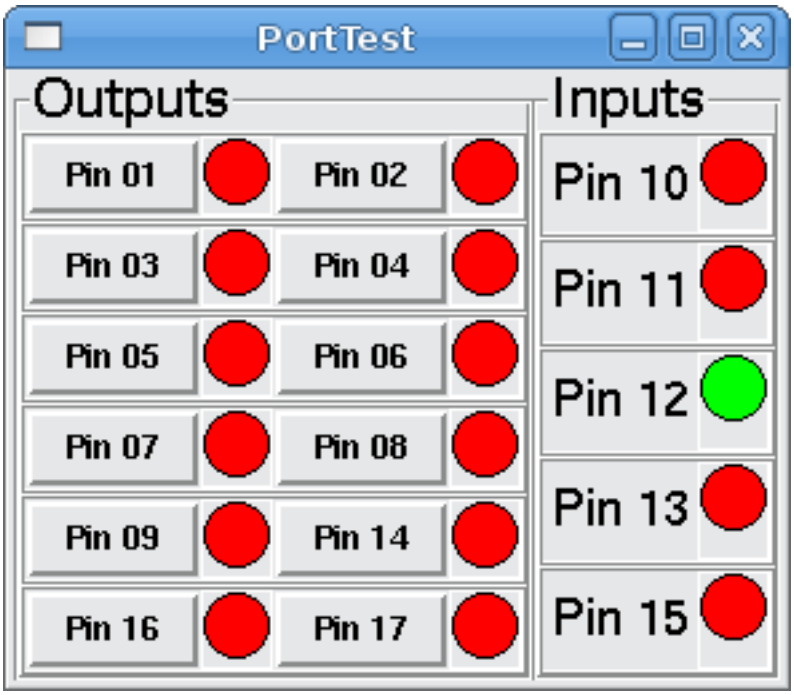


Figure 9.9: Testeur de port parallèle, complet

Pour ajouter le reste des pins du port parallèle, il suffi de modifier les fichiers .xml et .hal.

Pour visualiser les pins après avoir lancé le script HAL, utiliser la commande suivante au prompt halcmd:

```
halcmd: show pin
Component Pins:
Owner Type  Dir Value  Name
    2 bit   IN  FALSE  parport.0.pin-01-out <== pin01
    2 bit   IN  FALSE  parport.0.pin-02-out <== pin02
    2 bit   IN  FALSE  parport.0.pin-03-out
    2 bit   IN  FALSE  parport.0.pin-04-out
    2 bit   IN  FALSE  parport.0.pin-05-out
    2 bit   IN  FALSE  parport.0.pin-06-out
    2 bit   IN  FALSE  parport.0.pin-07-out
    2 bit   IN  FALSE  parport.0.pin-08-out
    2 bit   IN  FALSE  parport.0.pin-09-out
    2 bit   OUT TRUE   parport.0.pin-10-in ==> pin10
    2 bit   OUT FALSE  parport.0.pin-10-in-not
    2 bit   OUT TRUE   parport.0.pin-11-in ==> pin11
    2 bit   OUT FALSE  parport.0.pin-11-in-not
    2 bit   OUT TRUE   parport.0.pin-12-in
```

```
    2 bit   OUT FALSE  parport.0.pin-12-in-not
    2 bit   OUT TRUE   parport.0.pin-13-in
    2 bit   OUT FALSE  parport.0.pin-13-in-not
    2 bit   IN  FALSE  parport.0.pin-14-out
    2 bit   OUT TRUE   parport.0.pin-15-in
    2 bit   OUT FALSE  parport.0.pin-15-in-not
    2 bit   IN  FALSE  parport.0.pin-16-out
    2 bit   IN  FALSE  parport.0.pin-17-out
    4 bit   OUT FALSE  ptest.btn01 ==> pin01
    4 bit   OUT FALSE  ptest.btn02 ==> pin02
    4 bit   IN  FALSE  ptest.led-01 <== pin01
    4 bit   IN  FALSE  ptest.led-02 <== pin02
    4 bit   IN  TRUE   ptest.led-10 <== pin10
    4 bit   IN  TRUE   ptest.led-11 <== pin11
```

Cela montre quelles pins sont IN est lesquelles sont OUT, ainsi que toutes les connections.

9.2.5 Compte tours pour GS2

L'exemple suivant utilise un variateur de fréquence GS2 de la société Automation Direct. [1] Il permet le pilotage du moteur, la visualisation de la vitesse ainsi que d'autres informations dans un panneau PyVCP. Cet exemple est basé sur un autre, relatif au variateur GS2 et se trouvant dans la section des exemples matériels de ce manuel. Ce dernier exemple s'appuie lui même sur la description du composant de HAL gs2_vfd.

9.2.5.1 Le panneau

Pour créer le panneau nous ajoutons ce code au fichier .xml.

```
<pyvcp>

    <!-- Compte tours -->
    <hbox>
        <relief>RAISED</relief>
        <bd>3</bd>
        <meter>
            <halpin>"spindle_rpm"</halpin>
            <text>"Broche"</text>
            <subtext>"tr/mn"</subtext>
            <size>200</size>
            <min_>0</min_>
            <max_>3000</max_>
            <majorscale>500</majorscale>
            <minorscale>100</minorscale>
            <region1>0,10,"yellow"</region1>
        </meter>
    </hbox>

    <!-- La Led On -->
    <hbox>
        <relief>RAISED</relief>
        <bd>3</bd>
        <vbox>
            <relief>RAISED</relief>
            <bd>2</bd>
            <label>
```

[1] En Europe on trouve ce type de variateur sous la marque Omron.

```
                <text>”On”</text>
                <font>(”Helvetica”,18)</font>
            </label>
            <width>5</width>
            <hbox>
                <label width=”2”/> <!-- utilisé pour centrer la Led -->
                <rectled>
                    <halpin>”on-led”</halpin>
                    <height>”30”</height>
                    <width>”30”</width>
                    <on_color>”green”</on_color>
                    <off_color>”red”</off_color>
                </rectled>
            </hbox>
        </vbox>

        <!-- La Led Sens horaire -->
        <vbox>
            <relief>RAISED</relief>
            <bd>2</bd>
            <label>
                <text>”Sens horaire”</text>
                <font>(”Helvetica”,18)</font>
                <width>5</width>
            </label>
            <label width=”2”/>
            <rectled>
                <halpin>”fwd-led”</halpin>
                <height>”30”</height>
                <width>”30”</width>
                <on_color>”green”</on_color>
                <off_color>”red”</off_color>
            </rectled>
        </vbox>

        <!-- La Led Sens inverse -->
        <vbox>
            <relief>RAISED</relief>
            <bd>2</bd>
            <label>
                <text>”Sens inverse”</text>
                <font>(”Helvetica”,18)</font>
                <width>5</width>
            </label>
            <label width=”2”/>
            <rectled>
                <halpin>”rev-led”</halpin>
                <height>”30”</height>
                <width>”30”</width>
                <on_color>”red”</on_color>
                <off_color>”green”</off_color>
            </rectled>
        </vbox>
    </hbox>
</pyvcp>
```

L’image ci-dessous montre notre panneau PyVCP en fonctionnement.

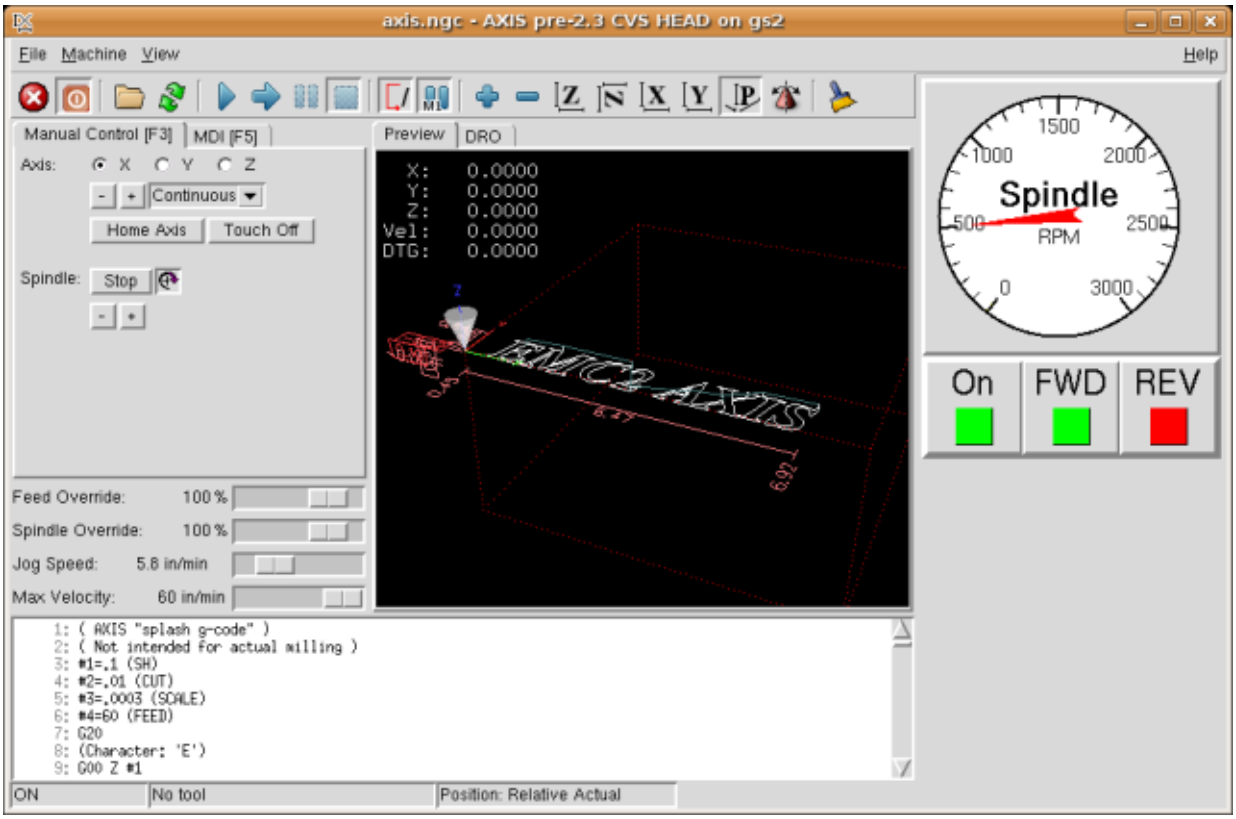


Figure 9.10: Panneau pour GS2

9.2.5.2 Les connections

Pour qu'il fonctionne, il est nécessaire d'ajouter le code suivant au fichier custom_postgui.hal, il réalise les connections entre PyVCP et LinuxCNC.

```
# affiche le compte tours, calcul basé sur freq * rpm par hz
loadrt mult2
addf mult2.0 servo-thread
setp mult2.0.in1 28.75
net cypher_speed mult2.0.in0 <= spindle-vfd.frequency-out
net speed_out pyvcp.spindle_rpm <= mult2.0.out

# la led On
net gs2-run => pyvcp.on-led

# la led Sens horaire
net gs2-fwd => pyvcp.fwd-led

# la led Sens anti-horaire
net running-rev spindle-vfd.spindle-rev => pyvcp.rev-led
```

Certaines lignes demandent quelques explications.

- La ligne de la led Sens horaire utilise le signal créé dans le fichier custom.hal dans lequel la led Sens inverse doit utiliser le bit spindle-rev.
- On ne peut pas lier deux fois le bit spindle-fwd pour utiliser le signal auquel il est déjà lié.

9.3 Création d'interfaces graphiques avec GladeVCP

9.3.1 Qu'est-ce que GladeVCP?

GladeVCP est un composant de LinuxCNC qui donne la possibilité d'ajouter de nouvelles interfaces graphiques utilisateur à LinuxCNC telles qu'Axis ou Touchy. À la différence de PyVCP, GladeVCP n'est pas limité à l'affichage et aux réglages des pins de HAL, toutes les actions peuvent être exécutées en code Python. En fait, une interface utilisateur LinuxCNC complète peut être construite avec GladeVCP et Python.

GladeVCP utilise l'environnement graphique et WYSIWYG Glade qui simplifie l'édition et la création visuelle de panneaux esthétiquement très réussis. Il s'appuie sur les liaisons entre PyGTK et le riche jeu de widgets GTK+, finalement, tous peuvent être utilisés dans une application GladeVCP et pas seulement les widgets spécialisés pour interagir avec HAL et LinuxCNC présentés ici.

9.3.1.1 PyVCP par rapport à GladeVCP

Tous les deux supportent la création de panneaux avec des widgets de HAL, des éléments utilisateur visuels tels que boutons, Leds, curseurs etc. dont les valeurs sont liées à des pins de HAL qui à leur tour, sont des interfaces pour le reste de LinuxCNC.

PyVCP

- Jeu de widgets: utilise les widgets TkInter.
- Cycle de création d'interfaces utilisateur:
  - Éditer les fichiers XML
  - Lancer
  - Évaluer le look.
- Pas de support pour intégrer une gestion des événements définie par l'utilisateur.
- Pas d'interaction avec LinuxCNC au-delà des interactions avec les pins d'E/S de HAL supportées.

GladeVCP

- Jeu de widgets: Liaison avec le jeu de widgets de GTK+.
- Création d'interface utilisateur: utilise l'interface graphique Glade qui est un éditeur WYSIWYG.
- Tout changement sur une pin de HAL peut diriger un appel vers une gestion d'événements définie en Python par l'utilisateur.
- Tous les signaux GTK (touches/appui sur un bouton, fenêtre, E/S, timer, événements réseau) peuvent être associés avec la gestion d'événements définie en Python par l'utilisateur.
- Interaction directe avec LinuxCNC: exécution de commandes, telle qu'initialiser une commande MDI pour appeler un sous-programme G-code.
- Plusieurs panneaux GladeVCP indépendants peuvent tourner dans des onglets différents.
- Séparation entre l'apparence de l'interface et les fonctionnalités: change d'apparence sans passer par aucun code.

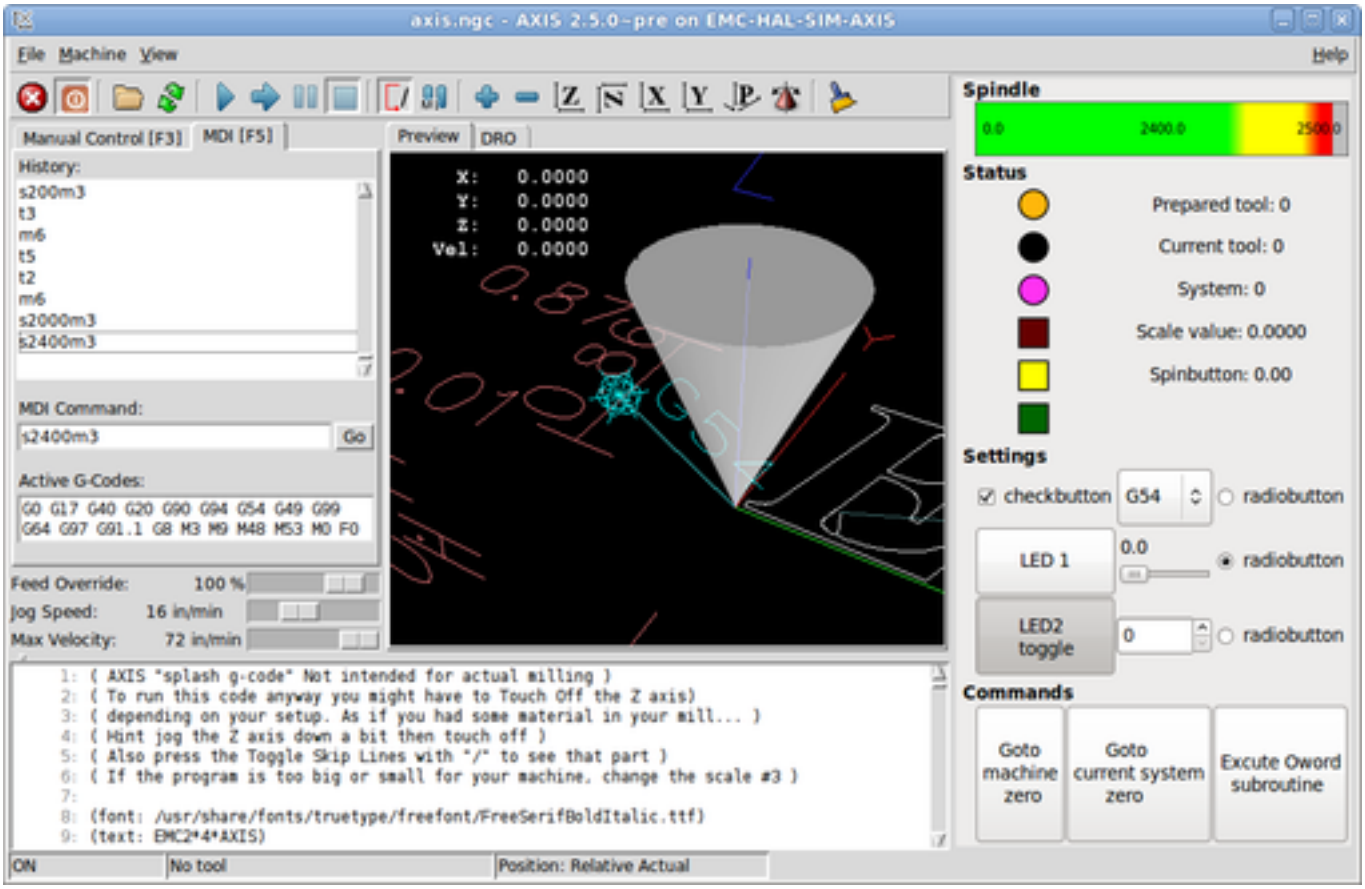
9.3.2 Description du fonctionnement, avec un exemple de panneau

Une fenêtre de panneau GladeVCP peut démarrer avec trois différentes configuration:

- Toujours visible, intégré dans Axis, du côté droit, exactement comme un panneau PyVCP.
- Dans un onglet dans Axis ou Touchy; dans Axis un troisième onglet sera créé à côté des deux d'origine, ils doivent être choisis explicitement.
- Comme une fenêtre indépendante, qui peut être iconisée ou agrandie, indépendamment de la fenêtre principale.

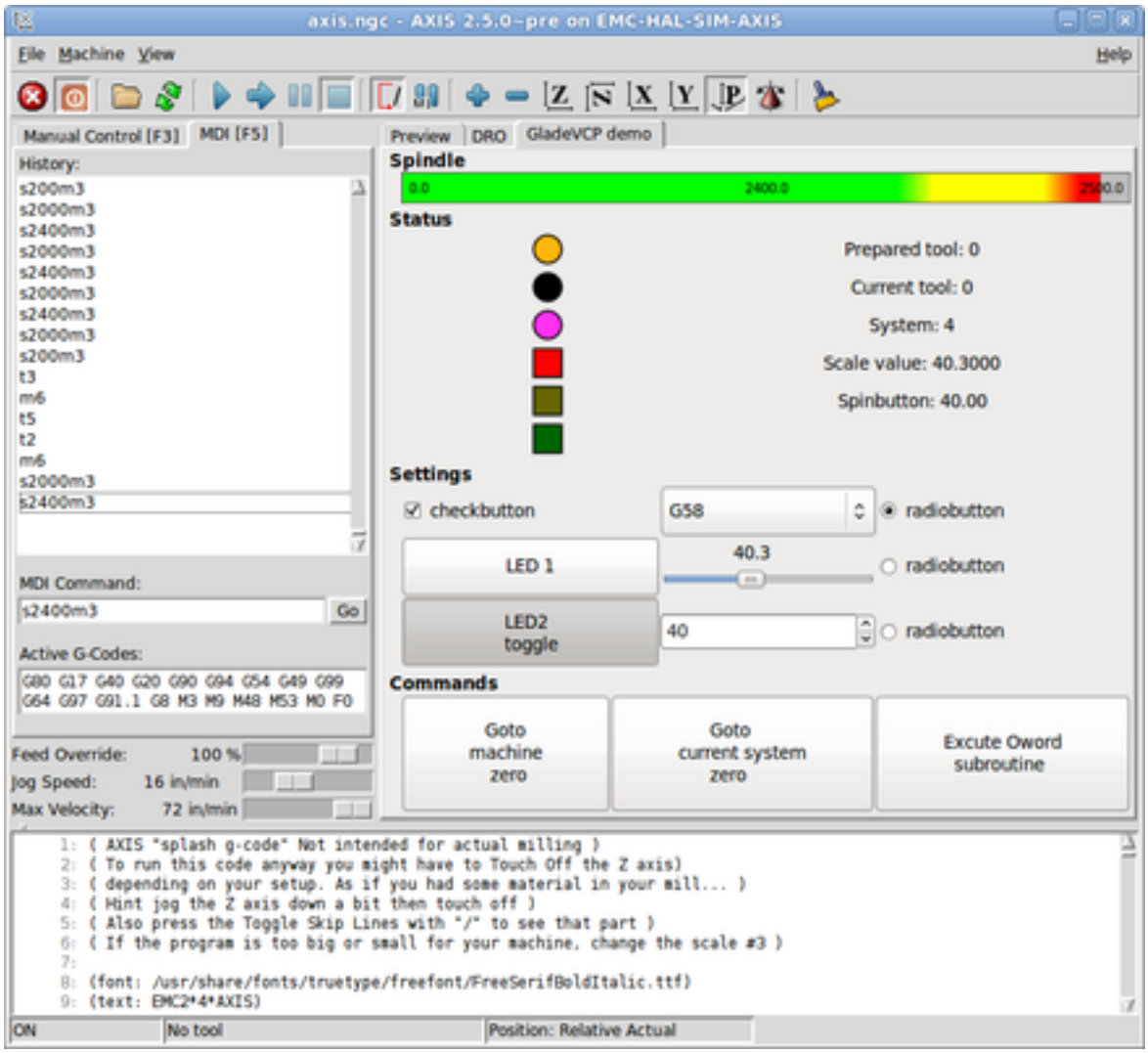
Lancer un panneau GladeVCP simple, intégré dans Axis comme PyVCP, taper les commandes suivantes:

```
$ cd configs/sim/gladevcp

$ linuxcnc gladevcp_panel.ini
```



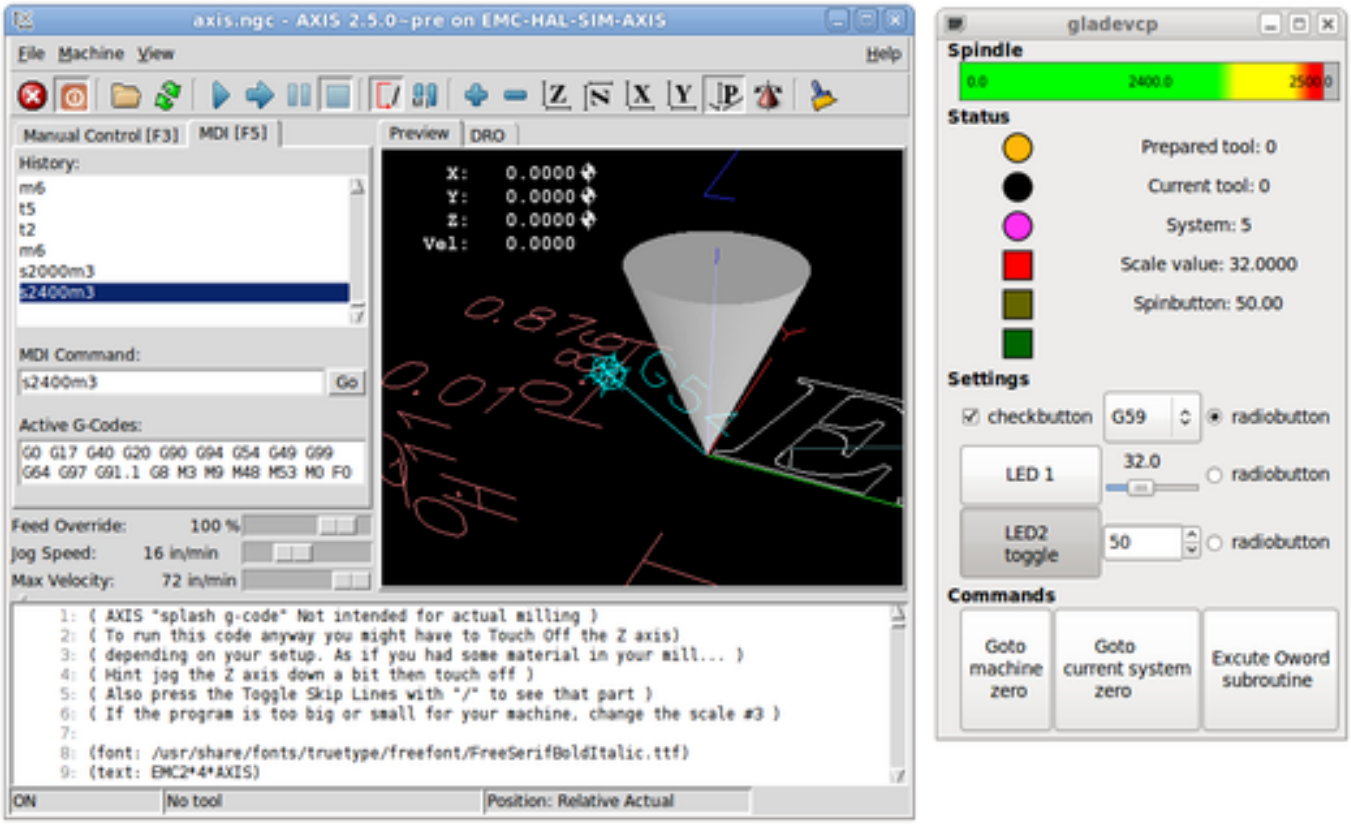
Lancer le même panneau, mais dans un onglet d'Axis avec:

```
$ cd configs/sim/gladevcp

$ linuxcnc gladevcp_tab.ini
```



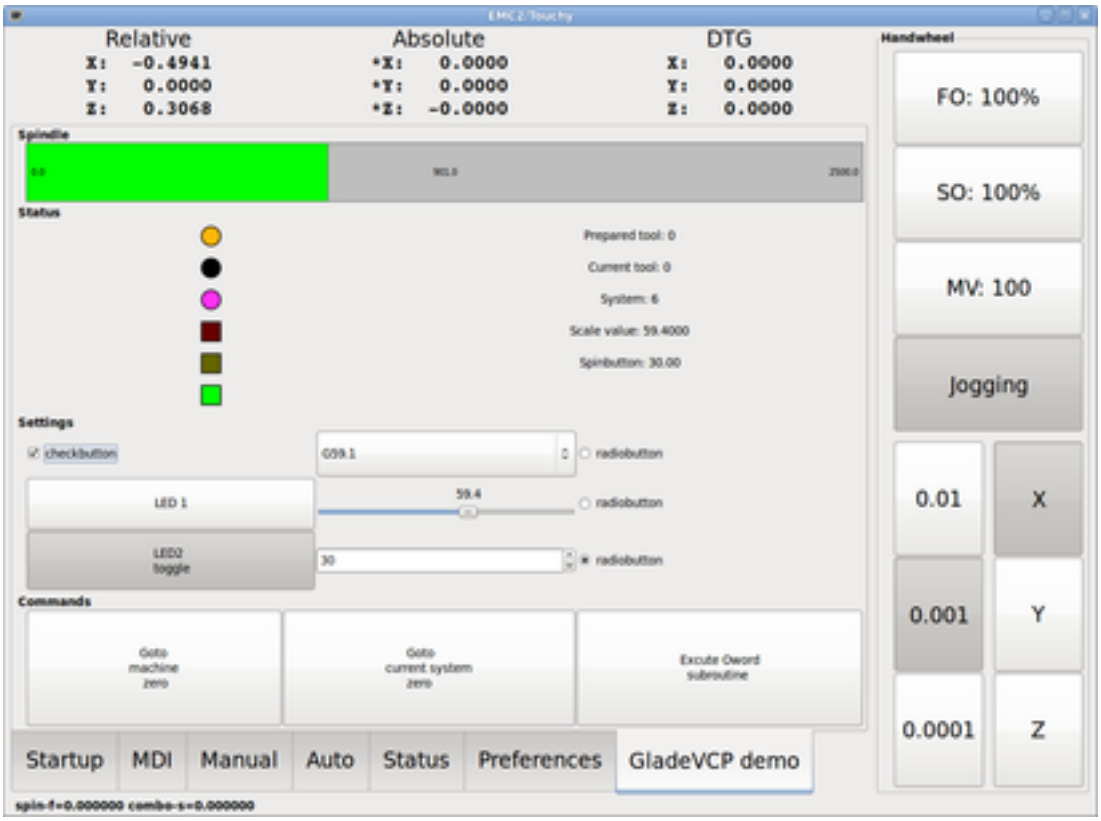
Pour lancer ce même panneau comme une fenêtre autonome à côté d'Axis, démarrer Axis en arrière plan puis démarrer gladevcp de la manière suivante:

```
$ cd configs/sim/gladevcp

$ linuxcnc axis.ini &

$ gladevcp -c gladevcp -u ../gladevcp/hitcounter.py -H
../gladevcp/manual-example.hal ../gladevcp/manual-example.ui
```



Pour lancer ce panneau dans Touchy:

```
$ cd configs/sim

$ linuxcnc gladevcp_touchy.ini
```



Fonctionnellement, ces configurations sont identiques. La seule différence porte sur l'état et la visibilité de l'écran. Puisqu'il est possible de lancer plusieurs composants GladeVCP en parallèle (avec des noms de modules de HAL différents), le mélange des configurations est également possible. Par exemple, un panneau sur le côté droit et un ou plusieurs en onglets pour des parties d'interface moins souvent utilisées.

9.3.2.1 Description de l'exemple de panneau

Pendant qu'Axis est en marche, explorons Afficher configuration de HAL dans lequel nous trouvons le composant de HAL gladevcp et dont nous pouvons observer la valeur des pins pendant l'interaction avec les widgets du panneau. La configuration de HAL peut être trouvée dans configs/gladevcp/manual-example.hal.

Usage des deux cadres en partie basse. Le panneau est configuré pour que, quand l'Arrêt d'Urgence est désactivé, le cadre Settings s'active et mette la machine en marche, ce qui active à son tour le cadre Commandes du dessous. Les widgets de HAL du cadre Settings sont liés aux Leds et labels du cadre Status ainsi qu'au numéros de l'outil courant et à celui de l'outil préparé. Les utiliser pour bien voir leur effet. L'exécution des commandes T<numéro d'outil> et M6 dans la fenêtre du MDI aura pour effet de changer les numéros de l'outil courant et de l'outil préparé dans les champs respectifs.

Les boutons du cadre Commandes sont des widgets d'action MDI. Les presser exécutera une commande MDI dans l'interpréteur. Le troisième bouton Execute Oword subroutine est un exemple avancé, il prends plusieurs pins de HAL du cadre Settings et leur passe comme paramètres, le sous-programme Oword. Les paramètres actuels reçus par la routine sont affichés par une commande (DEBUG,). Voir configs/gladevcp/nc_files/oword.ngc pour le corps du sous-programme.

Pour voir comment le panneau est intégré dans Axis, voir la déclaration de [DISPLAY]GLADEVCP dans gladevcp_panel.ui, ainsi que les déclarations de [DISPLAY]EMBED et de [HAL]POSTGUI_HALFILE dans gladevcp_tab.ini, respectivement.

9.3.2.2 Description de l'éditeur de Glade

L'interface utilisateur est créée avec l'éditeur graphique de Glade. Pour l'essayer il faut avoir le prérequis nécessaire, que glade soit installé. Pour éditer l'interface utilisateur, lancer la commande:

```
$ glade configs/gladevcp/manual-example.ui
```

La zone centrale de la fenêtre montre l'apparence de l'interface en création. Tous les objets de l'interface et les objets supportés se trouvent dans la partie haute à droite de la fenêtre, où il est possible de choisir un widget spécifique (ou en cliquant sur lui au centre de la fenêtre). Les propriétés du widget choisi sont affichées et peuvent être modifiées, dans le bas à droite de la fenêtre.

Pour voir comment les commandes MDI sont passées depuis les widgets d'action MDI, explorer la liste des widgets sous Actions en haut à droite de la fenêtre, et dans le bas à droite de la fenêtre, sous l'onglet Général, les propriétés des commandes MDI.

9.3.2.3 Explorer la fonction de rappel de Python

Voici comment une fonction de rappel Python est intégrée dans l'exemple:

- Dans glade, regarder le label du widget `hits` (un widget GTK+).
- Dans le widget `button1`, regarder dans l'onglet Signaux et trouver le signal pressed associé avec le gestionnaire on_button_press.
- Dans ../gladevcp/hitcounter.py, regarder la méthode on_button_press et comment elle place la propriété du label dans l'objet hits.

C'était juste pour toucher le concept du doigt. Le mécanisme de fonction de rappel sera détaillé plus en détails dans la section Programmation de GladeVCP.

9.3.3 Créer et intégrer une interface utilisateur Glade

9.3.3.1 Pré-requis: Installation de Glade

Pour visualiser ou modifier les fichiers d'une interface Glade, Glade doit être installé. Ce n'est pas nécessaire pour seulement essayer un panneau GladeVCP. Si la commande glade est manquante, l'installer de la manière suivante:

```
$ sudo apt-get install glade
```

Vérifier ensuite la version installée, qui doit être égale ou supérieure à 3.6.7:

```
$ glade --version
```

```
glade3 3.6.7
```

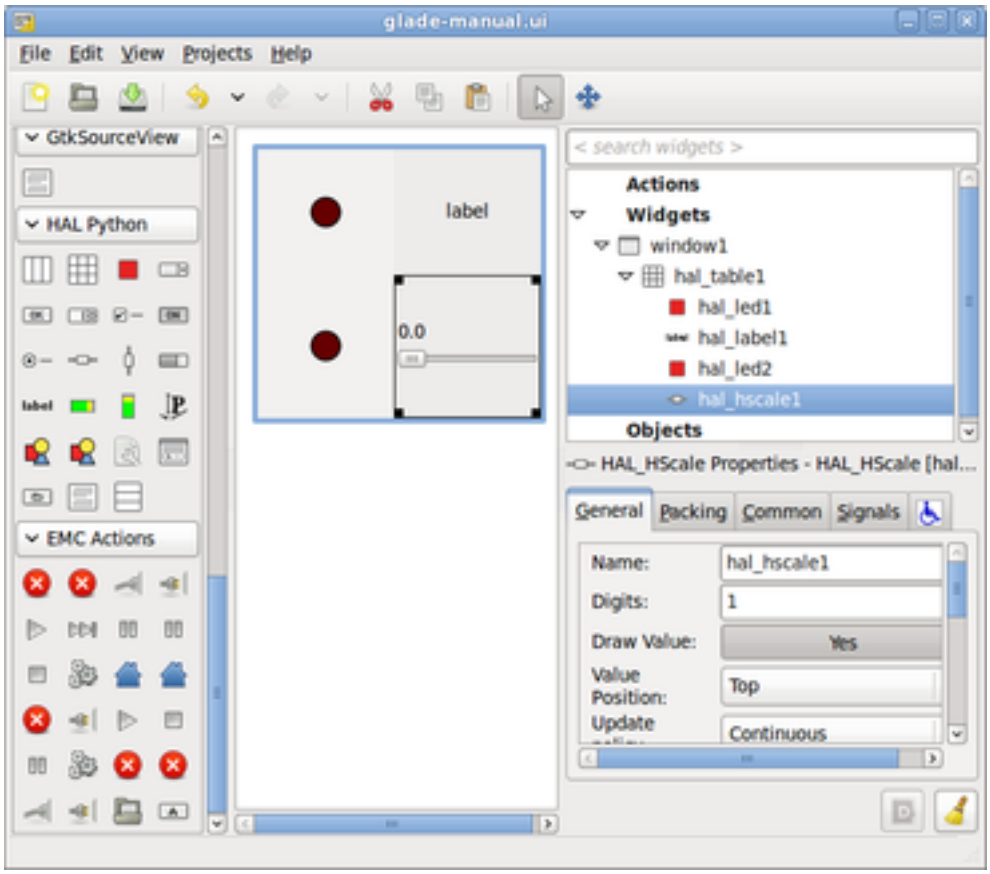
9.3.3.2 Lancer Glade pour créer une nouvelle interface utilisateur

Cette section souligne juste les étapes initiales spécifiques à LinuxCNC. Pour plus d'informations et un tutoriel sur Glade, voir http://glade.gnome.org. Certains trucs & astuces sur Glade, peuvent aussi être trouvés sur youtube.

Soit modifier une interface existante en lançant `glade <fichier>.ui` ou, démarrer une nouvelle en lançant juste la commande `glade` depuis un terminal.

- Si LinuxCNC n'a pas été installé depuis un paquetage, l'environnement LinuxCNC du shell doit être configuré avec . <linuxcncdir>/scripts/rip-environment, autrement Glade ne trouverait pas les widgets spécifiques à LinuxCNC.
- Quand l'éditeur demande pour enregistrer les préférences, accepter ce qui est proposé par défaut et presser Close.
- Depuis les Niveaux supérieurs (cadre de gauche), choisir Fenêtre (première icône) en haut des Niveaux supérieurs, par défaut cette fenêtre sera nommée window1. Ne pas changer ce nom, GladeVCP lui est relié.
- Dans le bas des onglets de gauche, dérouler HAL Python et LinuxCNC Actions.
- Ajouter au nouveau cadre, un conteneur comme une boîte HAL_Box ou une HAL_Table depuis HAL Python.
- Pointer et placer dans un conteneur d'autres éléments, comme une LED, un bouton, etc.

Le résultat pourrait ressembler à cela:



Glade a tendance à écrire beaucoup de messages dans la fenêtre du terminal, la plupart peuvent être ignorés. Sélectionner Fichier → Enregistrer sous, donner lui un nom comme myui.ui et bien vérifier qu'il sera enregistré comme un fichier GtkBuilder (bouton radio en bas à gauche du dialogue d'enregistrement). GladeVCP peut aussi traiter correctement l'ancien format libglade mais il n'y a aucune raison de l'utiliser. Par convention, l'extension des fichier GtkBuilder est .ui.

9.3.3.3 Tester un panneau

Vous êtes maintenant prêt à faire un essai (avec LinuxCNC, par exemple Axis en marche) faites:

```
gladevcp myui.ui
```

GladeVCP crée le composant de HAL portant le nom qui a été donné au fichier, par exemple, le très original myui.ui dans notre cas, à moins qu'il n'ait été surchargé pat l'option `-c <nom du composant>`. Si Axis est en marche, essayer de trouver le composant dans Afficher configuration de HAL et inspecter ses pins.

Vous vous demandez peut être pourquoi les widgets conteneurs comme HAL_Hbox ou HAL_Table apparaissent grisés (inactifs). Les conteneurs HAL ont une pin de HAL associée qui est désactivée par défaut, c'est ce qui cause ce rendu grisé des widgets conteneurs inactifs. Un cas d'utilisation courante pourrait être pour associer les pins de HAL du conteneur `halui.machine.is-on` ou un des signaux `halui.mode.`, pour s'assurer que certains widgets n'apparaissent actifs que dans un certain état.

Pour activer un conteneur, exécuter la commande HAL `setp gladevcp.<nom-du-conteneur> 1`.

9.3.3.4 Préparer le fichier de commande HAL

La voie suggérée pour lier les pins de HAL dans un panneau GladeVCP consiste à les collecter dans un fichier séparé portant l'extension `.hal`. Ce fichier est passé via l'option `POSTGUI_HALFILE=`, dans la section `[HAL]` du fichier de configuration.

ATTENTION: Ne pas ajouter le fichier de commandes HAL de GladeVCP à la section ini d'Axis `[HAL]HALFILE=` ça n'aurait pas l'effet souhaité. Voir les sections suivantes.

9.3.3.5 Intégration dans Axis, comme pour PyVCP

Pour placer le panneau GladeVCP dans la partie droite d'Axis, ajouter les lignes suivantes dans le fichier ini:

```
[DISPLAY]
# ajouter le panneau GladeVCP à l'emplacement de PyVCP:
GLADEVCP= -u ../gladevcp/hitcounter.py ../gladevcp/manual-example.ui

[HAL]
# Les commandes HAL pour les composants GladeVCP dans un onglet, doivent être
exécutées via POSTGUI_HALFILE
POSTGUI_HALFILE =  ../gladevcp/manual-example.hal

[RS274NGC]
# les sous-programmes Oword spécifiques à gladevcp se placent ici
SUBROUTINE_PATH = ../gladevcp/nc_files/
```

Le nom de composant HAL d'une application GladeVCP lancé avec l'option GLADEVCP est toujours: `gladevcp`. La ligne de commande actuellement lancée par Axis dans la configuration ci-dessous est la suivante:

```
halcmd loadusr -Wn gladevcp gladevcp -c gladevcp -x {XID} <arguments pour GLADEVCP>
```

Ce qui veux dire que n'importe quelle option gladevcp, peut être ajoutée ici, tant qu'elle n'entre pas en collision avec les options des lignes de commande suivantes.

Note
L'option `[RS274NGC]SUBROUTINE_PATH=` est fixée seulement pour que l'exemple de panneau puisse trouver le sous-programme Oword pour le widget de commande MDI. Il n'est peut être pas nécessaire dans votre configuration.

9.3.3.6 Intégration dans un nouvel onglet d'Axis, à la suite des autres

Pour cela, éditer le fichier .ini et ajouter dans les sections DISPLAY et HAL, les lignes suivantes:

```
[DISPLAY]
# ajoute le panneau GladeVCP dans un nouvel onglet:
EMBED_TAB_NAME=GladeVCP demo
EMBED_TAB_COMMAND=halcmd loadusr -Wn gladevcp gladevcp -c gladevcp -x {XID} -u
../gladevcp/hitcounter.py ../gladevcp/manual-example.ui

[HAL]
# commandes HAL pour le composant GladeVCP dans un onglet doit être exécuté via
POSTGUI_HALFILE
POSTGUI_HALFILE =  ../gladevcp/manual-example.hal

[RS274NGC]
# les sous-programmes Oword spécifiques à gladevcp se placent ici
SUBROUTINE_PATH = ../gladevcp/nc_files/
```

Noter le halcmd loadusr pour charger la commande d'onglet, elle assure que POSTGUI_HALFILE ne sera lancé que seulement après que le composant de HAL ne soit prêt. Dans de rares cas, une commande pourrait être lancée ici, pour utiliser un onglet sans être associée à un composant de HAL. Une telle commande pourrait être lancée sans halcmd loadusr, ce qui indiquerait à Axis qu'il ne doit plus attendre un composant de HAL, puisqu'il n'existe pas.

Noter que quand le nom du composant est changé dans l'exemple suivant, les noms utilisés dans `-Wn <composant>` et `-c <composant>` doivent être identiques.

Essayer en lançant Axis, il doit avoir un nouvel onglet appelé GladeVCP demo à droite de l'onglet de la visu. Sélectionner cet onglet, le panneau de l'exemple devrait être visible, bien intégré à Axis.

Note
Bien vérifier que le fichier de l'interface est la dernière option passée à GladeVCP dans les deux déclarations `GLADEVCP=` et `EMBED_TAB_COMMAND=`.

9.3.3.7 Intégration dans Touchy

Pour ajouter un onglet GladeVCP à Touchy, éditer le fichier .ini comme cela:

```
[DISPLAY]
# ajoute un panneau GladeVCP dans un onglet
EMBED_TAB_NAME=GladeVCP demo
EMBED_TAB_COMMAND=gladevcp -c gladevcp -x {XID} -u ../gladevcp/hitcounter.py -H
../gladevcp/gladevcp-touchy.hal ../gladevcp/manual-example.ui

[RS274NGC]
# les sous-programmes Oword spécifiques à gladevcp se placent ici
SUBROUTINE_PATH = ../gladevcp/nc_files/
```

Noter les différences suivantes avec la configuration de l'onglet d'Axis:

- Le fichier de commandes HAL est légèrement modifié puisque Touchy n'utilise pas le composant halui, ses signaux ne sont donc pas disponibles et certains raccourcis ont été pris.
- Il n'y a pas d'option POSTGUI_HALFILE=, mais il est correct, de passer le fichier de commandes HAL,
  par la ligne EMBED_TAB_COMMAND=.
- L'appel halcmd loaduser -Wn … n'est pas nécessaire.

9.3.4 Options de GladeVCP en ligne de commande

Voir également, man gladevcp. Ce sont les options pour cette ligne de commande:

Usage: gladevcp [options] myfile.ui

Options:

-h, --help
Affiche ce message d'aide et sort.

-c NAME
Fixe le nom du composant à NAME. Par défaut, le nom de base des fichiers UI

-d
Active la sortie débogage

-g GEOMETRY
Fixe la géométrie à WIDTHxHEIGHT+XOFFSET+YOFFSET. Les valeurs sont en pixels, XOFFSET/YOFFSET est référencé à partir du coin haut, à gauche de l'écran.
Utilise -g WIDTHxHEIGHT pour fixer une taille ou -g +XOFFSET+YOFFSET pour fixer une position

-H FILE
exécute les déclarations de HAL depuis FILE, avec halcmd après que le composant soit chargé et prêt

-m MAXIMUM
force la fenêtre du panneau à se maximiser. Toutefois avec l'option -g geometry le panneau est déplaçable d'un moniteur à un autre en le forçant à utiliser toute l'écran

-t THEME
fixe le thème gtk. Par défaut, le thème système. Différents panneaux peuvent avoir différents thèmes. Un exemple de thème peut être trouvé sur le Wiki de LinuxCNC.

-x XID
Redonne un parent GladeVCP dans une fenêtre existante XID au lieu d'en créer une nouvelle au niveau supérieur

-u FILE
Utilise les FILE comme modules définis par l'utilisateur avec le gestionnaire

-U USEROPT
passe les modules python USEROPT

9.3.5 Références des Widgets HAL

GladeVcp inclus une collection de widgets Gtk qui ont des pins de HAL attachées, appelés widgets HAL, il sont destinés à contrôler, à afficher et à avoir d'autres interactions avec la couche HAL de LinuxCNC. Il sont destinés à être utilisés avec les interfaces créées par l'éditeur de Glade. Avec une installation correcte, les widgets HAL devraient être visibles, dans l'éditeur Glade, dans le groupe des Widgets HAL Python. Beaucoup de champs spécifiques à HAL dans l'onglet Général affichent une infobulle au survol de la souris.

Il y a deux variantes de signaux de HAL, bits et nombres. Les signaux bits sont les on/off. Les nombres peuvent être des "float", des "s32" ou des "u32". Pour plus d'informations sur les types de données de HAL, voir le manuel de HAL. Les widgets GladeVcp peuvent soit, afficher la valeur d'un signal avec un widget d'indication, soit, modifier la valeur d'un signal avec un widget de contrôle. Ainsi, il existe quatre classes de widgets gladvcp qui peuvent être connectés à un signal de HAL. Une autre classe de widgets d'aide permettent d'organiser et d'étiqueter les panneaux.

- Widgets d'indications "bit" signals: Led HAL
- Widgets de contrôle "bit" signals: HAL Bouton, HAL Bouton radio, HAL Case à cocher
- Widgets d'indications "nombre" signals: [gladevcp:HAL_Label], HAL Barre de progression, HAL HBar, HAL VBar, HAL Indicateur
- Widgets de contrôle "nombre" signals: boîte d'incrément, HAL HScale, HAL VScale
- widgets d'aide: HAL Table, HAL HBox
- Tracé du parcours d'outil: HAL Gremlin

Les widgets HAL héritent des méthodes, propriétés et signaux des widgets Gtk sous-jacents, il est donc utile de consulter le site du GTK+ ainsi que la documentation pour les liaisons avec PyGTK.

9.3.5.1 Nommage des Widgets HAL et de leurs pins

La plupart des widgets HAL on une simple pin de HAL associée et portant le même nom que le widget (glade: Général→Nom).

Les exceptions à cette règle sont actuellement:

- HAL_Spinbutton et HAL_ComboBox, qui ont deux pins: une pin `<nomwidget>-f` (float) et une pin `<nomwidget>-s` (s32)
- HAL_ProgressBar, qui a une pin d'entrée `<nomwidget>-value`, et une pin d'entrée `<nomwidget>-scale`.

9.3.5.2 Donner des valeurs aux Widgets HAL et à leurs pins

En règle générale, si une valeur doit être attribuée à la sortie d'un widget HAL depuis un code Python, le faire en appelant le setter Gtk sous-jacent (par exemple `set_active()`, `set_value()`), ne pas essayer de donner directement la valeur à la pin associée par un `halcomp[nompin] = value`, parce-que le widget ne verra jamais le changement!.

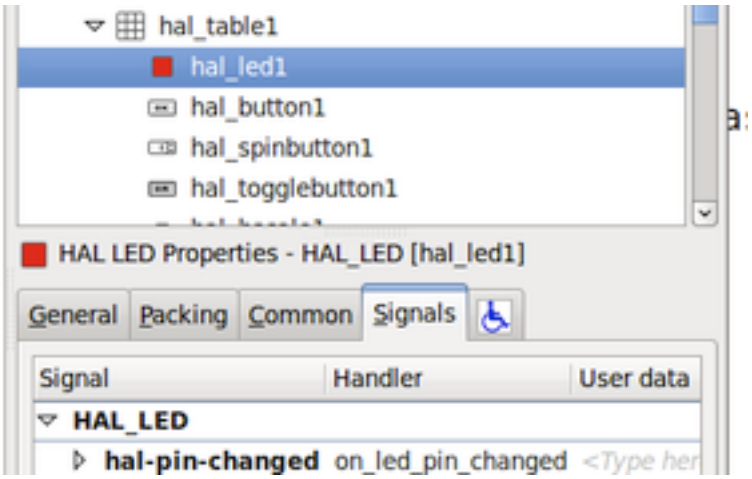
Il pourrait être tentant de fixer une pin d'entrée de widget HAL par programme. Noter que cela va à l'encontre du but premier d'une pin d'entrée. Elle devrait être attachée à un autre composant de HAL et réagir au signal qu'il génère. Bien qu'aucune protection, empêchant d'écrire sur les pins d'entrée HAL Python, ne soit présente actuellement, cela n'aurait aucun sens. Il faut utiliser `setp nompin valeur` dans un fichier Hal associé, pour les essais.

Il est par contre, parfaitement autorisé de mettre une valeur sur une pin de sortie de Hal avec `halcomp[nompin] = valeur` à condition que cette pin ne soit pas déjà associée avec un autre widget, ce qui aurait pu être créé par la méthode `hal_glib.GPin(halcomp.newpin(<nom>,<type>,<direction>)`. Voir la programmation de Glade-VCP pour d'autres exemples.

9.3.5.3 Le signal hal-pin-changed

La programmation événementielle signifie que l'interface graphique indique au code quand "quelque chose se produit", grâce à une fonction de rappel, comme quand un bouton est pressé, la sortie du widget HAL (ceux qui affichent la valeur des pins de HAL) comme une LED, une barre, une VBar, un indicateur à aiguille etc, supportent le signal hal-pin-changed qui peut provoquer une fonction de rappel dans le code Python quand une pin de HAL change de valeur. Cela veut dire qu'il n'est plus nécessaire d'interroger en permanence les pins de HAL dans le code pour connaitre les changements, les widgets font ça en arrière plan et le font savoir.

Voici un exemple montrant comment régler un signal `hal-pin-changed` pour une Hal Led, dans l'éditeur de Glade:

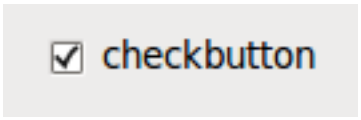


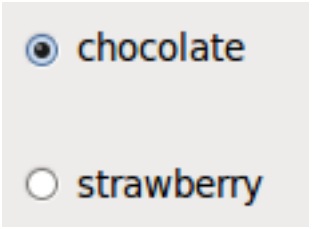
L'exemple dans `configs/gladevcp/examples/complex` montre comment c'est géré en Python.

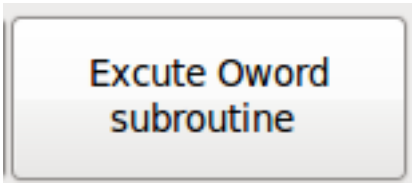
9.3.5.4 Les boutons (HAL Button)

Ce groupe de widgets est dérivé de divers boutons Gtk, ce sont les widgets HAL_Button, HAL_ToggleButton, HAL_RadioButton et CheckButton. Tous ont une seule pin de sortie BIT portant un nom identique au widget. Les boutons n'ont pas d'autres propriétés additionnelles, contrairement à leurs classes de base Gtk.

- HAL_Button: Action instantanée, ne retient pas l'état. Signal important: `pressed`.
- HAL_ToggleButton, HAL_CheckButton: Retiennent l'état on/off. Signal important: `toggled`.
- HAL_RadioButton: Un parmi un groupe. Signal important: `toggled` (par bouton).
- Importantes méthodes communes: `set_active()`, `get_active()`
- Importantes propriétés: `label`, `image`

Case à cocher: 

Boutons radio: 

Bouton à bascule: 

Tip
Définir les groupes de boutons radio dans Glade:

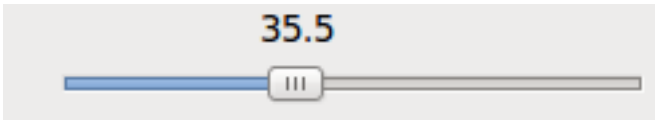
- Décider du bouton actif par défaut
- Dans les boutons radio, Général→Groupe sélectionner le nom du bouton actif par défaut dans le dialogue Choisir un Bouton radio pour ce projet.

Voir `configs/gladevcp/by-widget/radiobutton` pour une application GladeVCP avec un fichier d'interface utilisateur, pour travailler sur les boutons radio.

9.3.5.5 Les échelles (Scales)

HAL_HScale et HAL_VScale sont respectivement dérivées de GtkHScale et GtkVScale. Elles ont une pin de sortie FLOAT portant le même nom que le widget. Les échelles n'ont pas de propriété additionnelle.

Pour créer une échelle fonctionnelle dans Glade, ajouter un Ajustement (Général→Ajustement→Nouveau ou existant) et éditer l'objet ajustement. Il défini les valeurs défaut/min/max/incrément. Fixer la Sensibilité de l'incrément de l'ajustement sur automatique pour éviter les warnings.

Exemple d'échelle (HAL_hscale): 

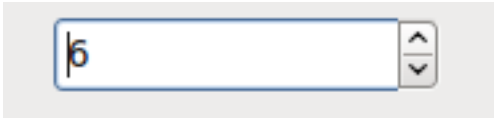
9.3.5.6 La boîte d'incrément (SpinButton)

La boîte d'incrément de HAL est dérivée de GtkSpinButton, elle a deux pins de sortie:

<nomwidget>-f
out FLOAT pin

<nomwidget>-s
out S32 pin

Pour être fonctionnelle, Spinbutton doit avoir une valeur d'ajustement comme l'échelle, vue précédemment.

Exemple de boîte d'incrément: 

9.3.5.7 Les labels

Le Label HAL est un simple widget basé sur GtkLabel qui représente la valeur d'une pin de HAL dans un format défini par l'utilisateur.

HAL pin type
Les pins de HAL sont des types (0:S32, 1:float ou 2:U32), voir aussi l'infobulle d'info sur Général → HAL pin type, (noter que c'est différent de PyVCP qui lui, a trois widgets label, un pour chaque type).

text template
Détermine le texte à afficher, une chaine au format Python pour convertir la valeur de la pin en texte. Par défauts, à `%s` (les valeurs sont converties par la fonction str()), mais peut contenir n'importe quel argument légal pour la méthode format() de Python. Exemple: `Distance: %.03f` va afficher le texte et la valeur de la pin avec 3 digits fractionnaires remplis avec des zéros pour une pin FLOAT.

9.3.5.8 Les conteneurs: HAL_HBox et HAL_Table

Comparés à leurs contreparties Gtk ils ont une pin d'entrée BIT qui contrôle si les enfants des widgets sont sensitifs ou non. Si la pin est basse, alors les widgets enfants sont inactifs, ce qui est le comportement par défaut.

Tip
Si vous trouvez que certaines parties de votre application GladeVCP sont grisées (insensible), vérifiez que les pins d'un conteneur ne soient pas inutilisées.

9.3.5.9 Les Leds

La Led hal simule un vrai indicateur à Led. Elle a une seule pin d'entrée BIT qui contrôle son état: ON ou OFF. Les Leds ont quelques propriétés pour contrôler leur aspect:

on_color
Une chaine définissant la couleur ON de la Led. Peut être tout nom valide de gtk.gdk.Color. Ne fonctionne pas sous Ubuntu 8.04.

off_color
Un chaine définissant la couleur OFF de la Led. Peut être tout nom valide de gtk.gdk.Color ou la valeur spéciale dark. dark signifie que la couleur OFF sera fixée à 0.4 valeur de la couleur ON. Ne fonctionne pas sous Ubuntu 8.04.

pick_color_on, pick_color_off
Couleurs pour les états ON et OFF peuvent être représentées par une chaine comme #RRRRGGGGBBBB. Ces propriétés optionnelles ont la précédence sur on_color et off_color.

led_size
Rayon de la Led (pour une Led carrée, 1/2 côté)

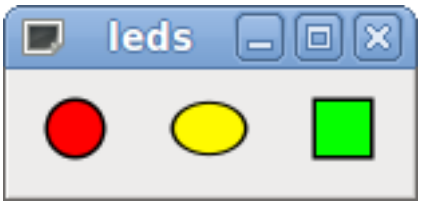
led_shape
Forme de la Led Shape. Les valeurs permises sont 0 pour ronde, 1 pour ovale et 2 pour carrée.

led_blink_rate
Si utilisée et que la Led est ON, alors la Led clignotera. La fréquence du clignotement est égal à la valeur de "led_blink_rate", spécifiée en millisecondes.

Comme un widget d'entrée, la Led aussi supporte le `hal-pin-changed signal`. Si vous voulez avoir une notification dans votre code quand les pins des Leds HAL ont changé d'état, alors connectez ce signal au gestionnaire, par exemple `on_led_pin_changed` et passez ce qui suit au gestionnaire:

```
def on_led_pin_changed(self,hal_led,data=None):
    print "on_led_pin_changed() - HAL pin value:",hal_led.hal_pin.get()
```

Ce code sera appelé à chaque front du signal et également au démarrage du programme pour reporter la valeur courante.

Exemple de Leds: 

9.3.5.10 La barre de progression (ProgressBar)

Note
Ce widget pourrait disparaître. Utilisez les widgets HAL_HBar et HAL_VBar à sa place.

La HAL_ProgressBar est dérivée de gtk.ProgressBar et a deux pins d'entrée de HAL float:

<nomwidget>
la valeur courante à afficher.

<nomwidget>-scale
la valeur maximum absolue en entrée.

Elle a les propriétés suivantes:

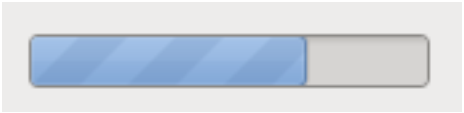
scale
Valeur d'échelle. fixe la valeur maximum absolue en entrée. Pareil que la configuration de la pin <nomwidget>.scale. Un flottant, compris entre -2^{24} et $+2^{24}$.

green_limit
Limite basse de la zone verte

yellow_limit
Limite basse de la zone jaune

red_limit
Limite basse de la zone rouge

text_template
Texte modèle pour afficher la valeur courante de la pin `<nomwidget>`. Formaté pour Python, peut être utilisé pour dict `{"valeur":valeur}`.

Exemple de barre de progression: 

9.3.5.11 La boîte combinée (ComboBox)

La comboBox HAL est dérivée de gtk.ComboBox. Elle valide le choix d'une valeur dans une liste déroulante.

Elle exporte deux pins de HAL:

<nomwidget>-f
La valeur courante, de type FLOAT

<nomwidget>-s
La valeur courante, de type S32

Elle a la propriété suivante, qui est configurable dans Glade:

column
L'index de colonne, type S32, défaut à -1, échelle de -1 à 100.

En mode par défaut, ces réglages du widget mettent les pins à la valeur d'index de l'entrée choisie dans la liste. Aussi, si le widget a trois labels, il peut seulement assumer les valeurs 0, 1 et 2.

En mode colonne (colonne > -1), la valeur reportée est choisie dans le tableau de stockage de liste défini dans Glade. Ainsi, typiquement la définition du widget devrait comprendre deux colonnes dans le tableau de stockage, une avec le texte affiché dans la liste déroulante, l'autre une valeur entière ou flottante correspondante au choix.

Il y a un exemple dans `configs/gladevcp/by-widget/combobox/combobox.{py,ui}` qui utilise le mode colonne pour prendre une valeur flottante dans un stockage de liste.

Si comme moi, vous êtes désorienté pour éditer une liste de stockage de ComboBox ou de CellRenderer, voyez http://www.youtube.com/watch?v=Z5_F-rW2cL8.

9.3.5.12 Les barres

Les widgets HAL, HBar et VBar pour barres Horizontale et Verticale, représentent des valeurs flottantes. Elles ont une pin d'entrée de HAL FLOAT. Chaque barre a les propriétés suivantes:

invert
Inverse les directions min avec max. Une HBar inversée croît de la droite vers la gauche, un VBar inversée croît du haut vers le bas.

min, max
Valeurs minimum et maximum de l'étendue souhaitée. Ce n'est pas une erreur si la valeur courante dépasse cette étendue.

zero
Point le plus bas de l'étendue. Si il est entre min et max, alors la barre croît à partir de cette valeur et non de la gauche du widget (ou de sa droite). Utile pour représenter des valeurs qui peuvent être à la fois, positives ou négatives.

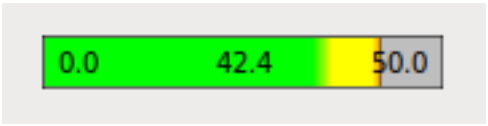
force_width, force_height
Force la largeur ou la hauteur du widget. Si inutilisés, la taille sera déduite du conteneur ou de la taille des widgets et des barres qui remplissent la zone.

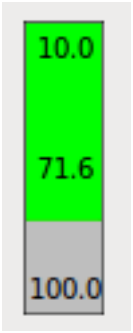
text_template
Détermine le texte à afficher, comme pour le Label, pour les valeurs min/max/courante. Peut être utilisé pour arrêter l'affichage de la valeur.

bg_color
Couleur de fond pour la barre (inactive).

z0_color, z1_color, z2_color
Couleurs des zones des différentes valeurs. Par défaut, green, yellow et red. Pour une description des zones voir propriétés des z _border.

z0_border, z1_border
Définissent les limites des zones de couleur. Par défaut, seule une zone est validée. Pour en activer plus d'une, fixer z0_border et z1_border aux valeurs souhaitées. Ainsi, zone 0 va remplir depuis 0 à la première bordure, zone 1 va remplir de la première à la seconde bordure et zone 2 depuis la dernière bordure jusqu'à 1. Les bordures se règlent comme des fractions, les valeurs vont de 0 à 1.

Barre horizontale: 

.Barre verticale: 

9.3.5.13 L'indicateur (HAL Meter)

L'indicateur est un widget similaire à celui de PyVCP, il représente une valeur flottante et a une pin d'entrée de HAL FLOAT. L'indicateur a les deux propriétés suivantes:

min, max
Valeurs minimum et maximum de l'étendue souhaitée. Ce n'est pas une erreur si la valeur courante dépasse cette étendue.

force_size
Force le diamètre du widget. Si inutilisé, alors la taille sera déduite du conteneur ou des dimensions d'un widget à taille fixe. L'indicateur occupera alors l'espace le plus grand disponible, tout en respectant les proportions.

text_template
Détermine le texte à afficher, comme pour le Label, pour la valeur courante. Peut être utilisé pour arrêter l'affichage de la valeur.

label
Label large au dessus du centre de l'indicateur.

sublabel
Petit label, sous le centre de l'indicateur.

bg_color
Couleur de fond de l'indicateur.

z0_color, z1_color, z2_color
Valeurs des couleurs des différentes zones. Par défaut, green, yellow et red. For description of zones see z _border properties.

z0_border, z1_border
Définissent les limites externes des zones de couleur. Par défaut, une seule zone de couleur est définie. Pour en activer plus d'une, fixer z0_border et z1_border aux valeurs souhaitées. Ainsi, zone 0 va remplir depuis min à la première bordure, zone 1 va remplir de la première à la seconde bordure et zone 2 depuis la dernière bordure jusqu'à max. Les bordures se règlent sur une étendue comprise en min et max.

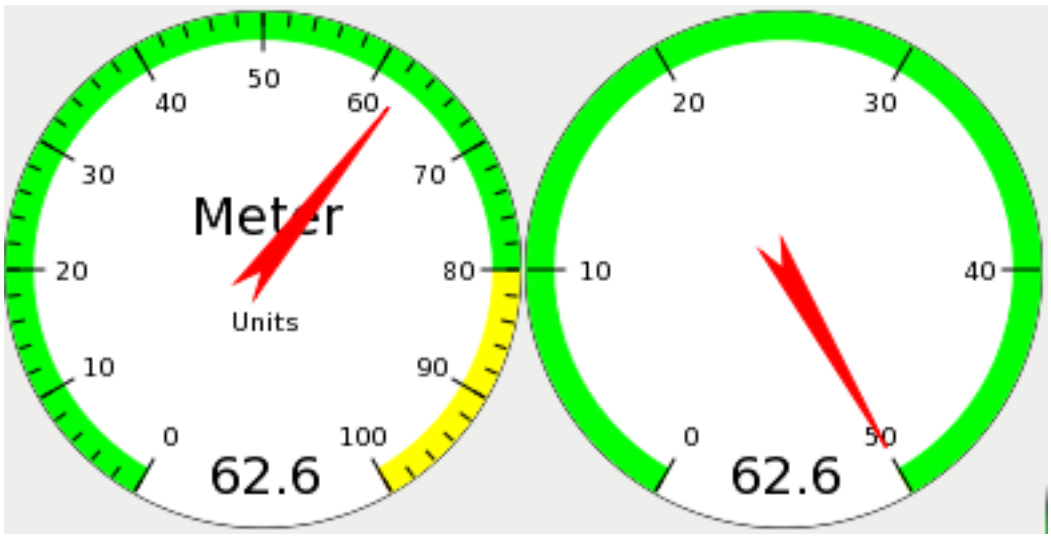


Figure 9.11: Exemples d'indicateurs:

9.3.5.14 Gremlin, visualiseur de parcours d'outil pour fichiers .ngc

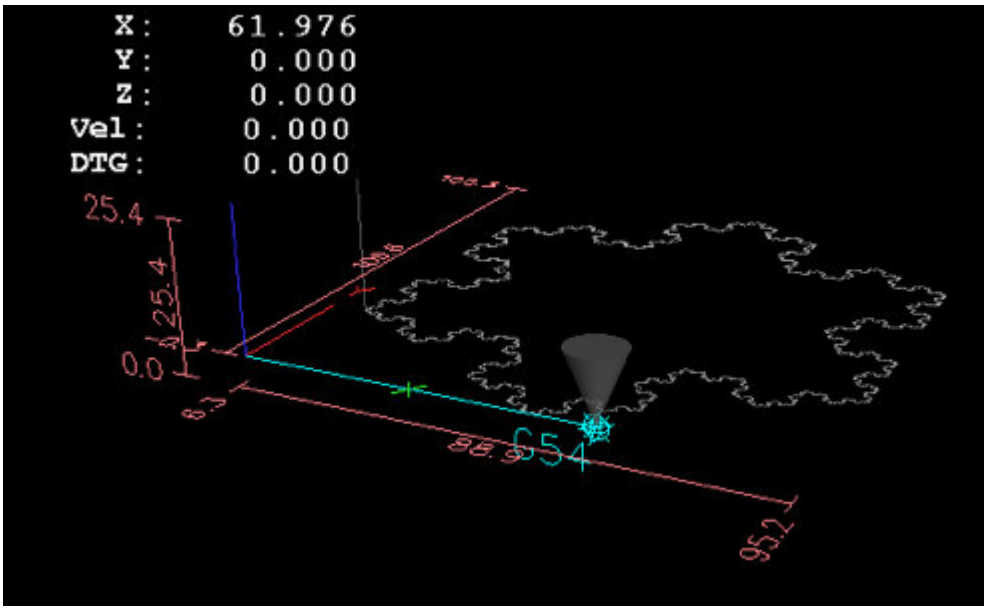
Gremlin est un traceur de parcours d'outil similaire à celui d'Axis. Il demande un environnement LinuxCNC en fonctionnement, comme Axis ou Touchy. Pour se connecter à lui, inspecter la variable d'environnement INI_FILE_NAME. Gremlin affiche le fichiers .ngc courant. Si le fichier ngc est modifié, il doit être rechargé pour actualiser le tracé. Si il est lancé dans une application GladeVCP quand LinuxCNC n'est pas en marche, un message va être affiché parce-que le widget Gremlin ne trouve pas le statut de LinuxCNC, comme le nom du fichier courant.

Gremlin n'exporte aucune pin de HAL. Il a les propriétés suivantes:

view
: Peut être la vue en x, y, z, p (perspective) . Par défaut, vue en z.

enable_dro
: Booléen; afficher une visu sur le tracé ou non. Par défaut,à True.

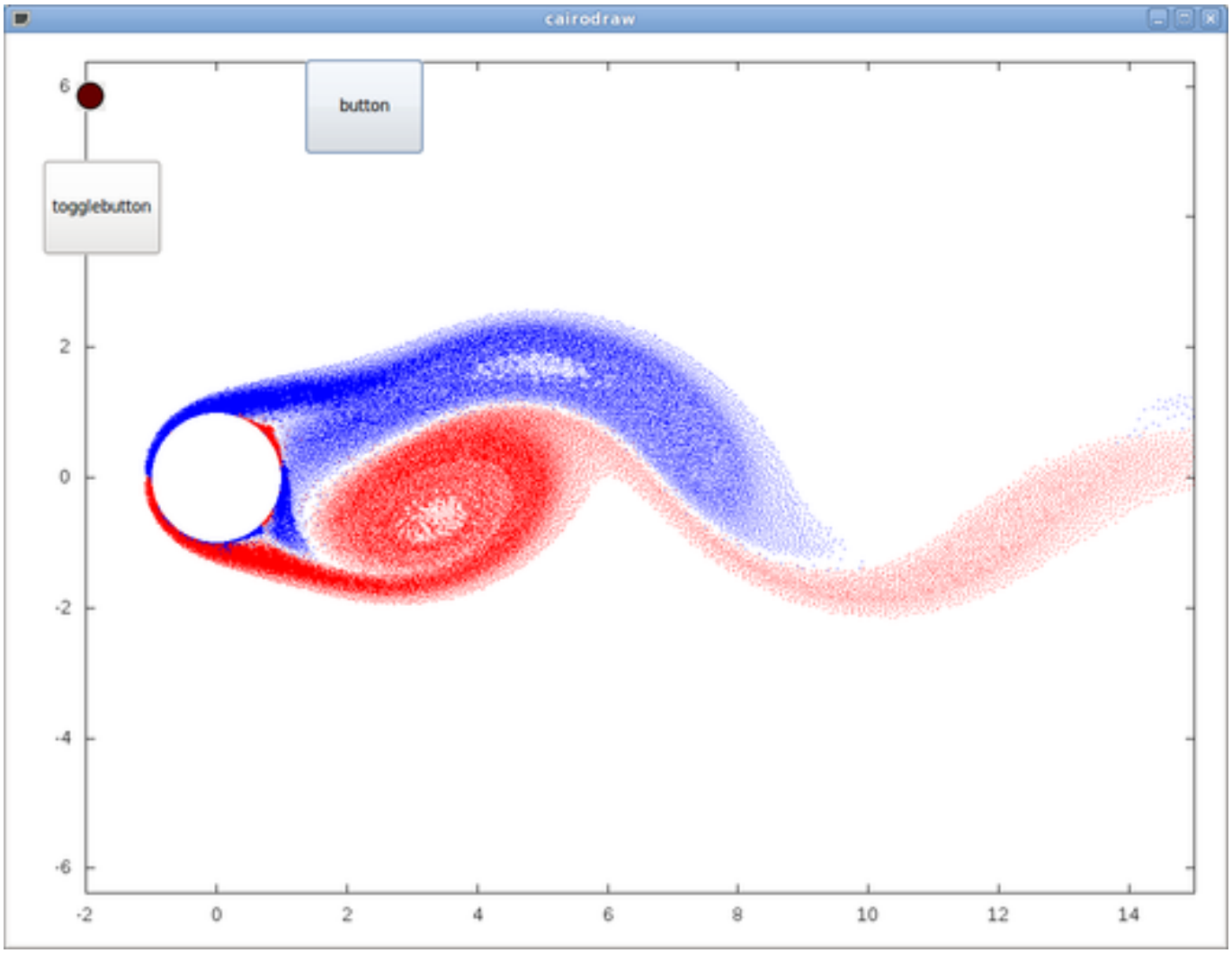
Exemple:



9.3.5.15 Fonction de diagrammes animés: Widgets HAL dans un bitmap

Pour certaines applications, il est intéressant d'avoir une image de fond, comme un diagramme fonctionnel et positionner les widgets aux endroits appropriés dans le diagramme. Une bonne combinaison consiste à placer une image de fond comme un fichier .png, mettre la fenêtre GladeVCP en taille fixe, et utiliser Glade pour fixer la position du widget sur cette image.

Le code pour l'exemple ci-dessus peut être trouvé dans `configs/gladevcp/animated-backdrop`:



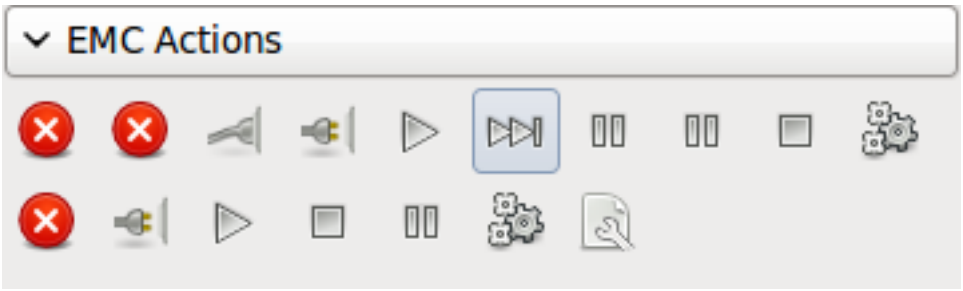
9.3.6 Références des Widgets LinuxCNC Action

GladeVcp inclus une collection d'actions préprogrammées appelées widgets LinuxCNC Action qui sont des Widgets pour l'éditeur Glade. À la différence des widgets HAL, qui interagissent avec les pins de HAL, les widgets LinuxCNC Actions, interagissent avec LinuxCNC et son interpréteur de G-code.

Les widgets LinuxCNC Action sont dérivés du widget Gtk.Action. Le widget LinuxCNC Action en quelques mots:

- C'est un objet disponible dans l'éditeur Glade.
- Il n'a pas d'apparence visuelle par lui-même.
- Son but: associer à un composant d'interface visible, à un composant d'interface sensitif, comme un menu, un bouton outil, un bouton avec une commande. Voir les propriétés des widgets Action dans Général → Related Action de l'éditeur.
- L'action préprogrammée sera exécutée quand l'état du composant associé basculera (bouton pressé, menu cliqué…)
- Ils fournissent une voie facile pour exécuter des commandes sans avoir à faire appel à la programmation en Python.

L'apparence des LinuxCNC Actions dans Glade est approximativement la suivante:



Le survol de la souris donne une infobulle.

9.3.6.1 Les widgets LinuxCNC Action

Les widgets LinuxCNC Action sont des widgets de type simple état. Ils implémentent une seule action par l'usage, d'un seul bouton, d'une option de menu, d'un bouton radio ou d'une case à cocher.

9.3.6.2 Les widgets LinuxCNC bascule action (ToggleAction)

Ce sont des widgets double état. Ils implémentent deux actions ou utilisent un second état (habituellement, pressé) pour indiquer qu'une action est actuellement en cours. Les bascules action sont prévues pour être utilisées avec les boutons à bascule (ToggleButtons) et les boutons à bascule d'outil (ToggleToolButtons) ou encore, pour basculer les items de menu. Un exemple simple est le bouton à bascule d'Arrêt d'Urgence (EStop).

Actuellement, les widgets suivants sont disponibles:

- La bascule d'Arrêt d'Urgence (ESTOP) envoie la commande ESTOP ou ESTOP_RESET à LinuxCNC, selon l'état courant.
- La bascule ON/OFF envoie la commande STATE_ON ou STATE_OFF.
- La bascule Pause/Reprise envoie la commande AUTO_PAUSE ou AUTO_RESUME.

Les bascules action suivantes ont seulement une commande associée et utilisent l'état pressé pour indiquer que l'opération demandée est lancée:

- La bascule Run envoie la commande AUTO_RUN et attends dans l'état pressé jusqu'à ce que l'interpréteur soit de nouveau au repos.
- La bascule Stop est inactive jusqu'à ce que l'interpréteur passe à l'état actif (Un G-code est lancé) et permet alors à l'utilisateur d'envoyer la commande AUTO_ABORT.
- La bascule MDI envoie la commande passée dans le MDI et attends sa complétion dans l'état inactif pressé.

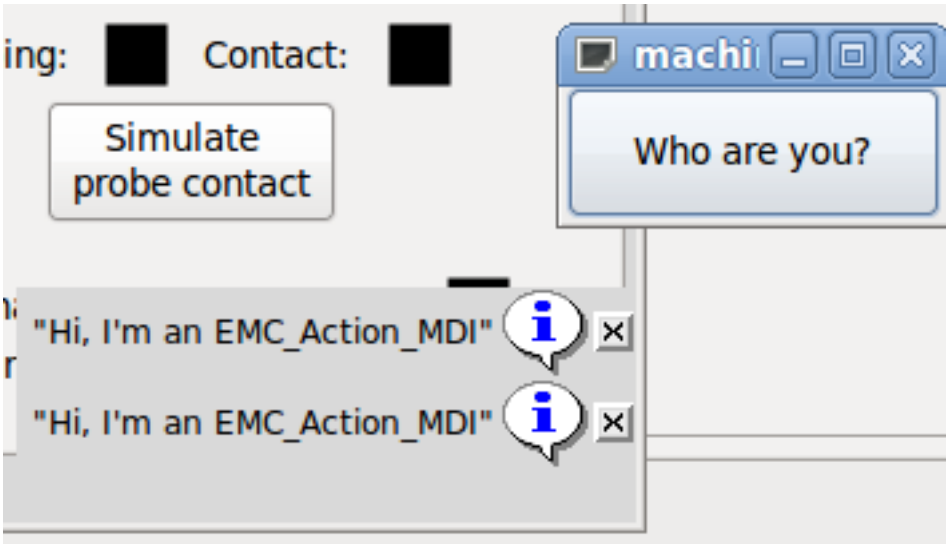
9.3.6.3 La bascule Action_MDI et les widgets Action_MDI

Ces widgets fournissent le moyen d'exécuter des commandes MDI. Le widget Action_MDI n'attends pas la complétion de la commande, comme le fait la bascule Action_MDI, qui reste elle, désactivée tant que la commande n'est pas terminée.

9.3.6.4 Un exemple simple: Exécuter une commande MDI lors de l'appui sur un bouton.

`configs/gladevcp/mdi-command-example/whoareyou.ui` est un fichier UI Glade qui transmet cette action basique:

L'ouvrir dans Glade et étudier comment il est fait. Lancer Axis puis dans un terminal faire: +gladevcp whoareyou.ui+. Voir l'action `hal_action_mdi1` et les propriétés de `MDI command` qui exécute juste `(MSG, "Hi, I'm an LinuxCNC_Action_MDI")` ce qui ouvre un popup de message dans Axis, comme ci-dessous:



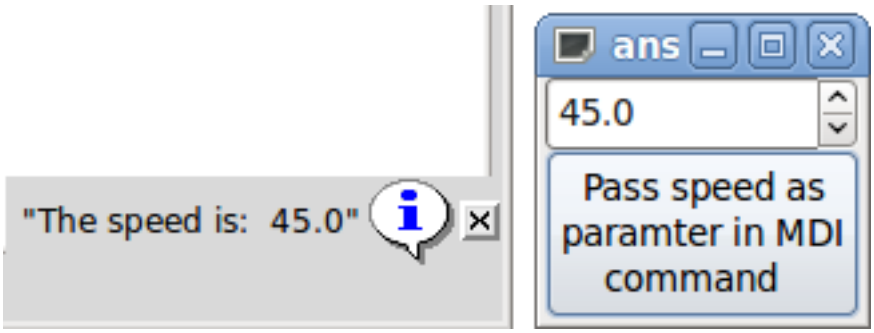
Noter que le bouton, associé à l'Action_MDI, est grisé si la machine est arrêtée, en A/U ou si l'interpréteur est déjà en marche. Il deviendra automatiquement actif quand la machine sera mise en marche donc, sortie de l'A/U (E-Stop), et que le programme est au repos.

9.3.6.5 Paramètres passés avec les widgets Action_MDI et ToggleAction_MDI

Optionnellement, la chaine MDI command peut avoir des paramètres substitués avant d'être passée à l'interpréteur. Ces paramètres sont actuellement les noms des pins de HAL dans les composants GladeVCP. Voici comment cela fonctionne:

- Supposons que nous avons une SpinBox HAL nommée `speed`, nous voulons passer sa valeur courante comme paramètre dans une commande MDI.
- La SpinBox HAL aura une pin de HAL de type flottant, nommée speed-f (voir la description des Widgets Hal).
- Pour substituer cette valeur dans la commande MDI, insérons le nom de la pin de HAL
- Pour la spinbox HAL précédente, il aurait été possible d'utiliser

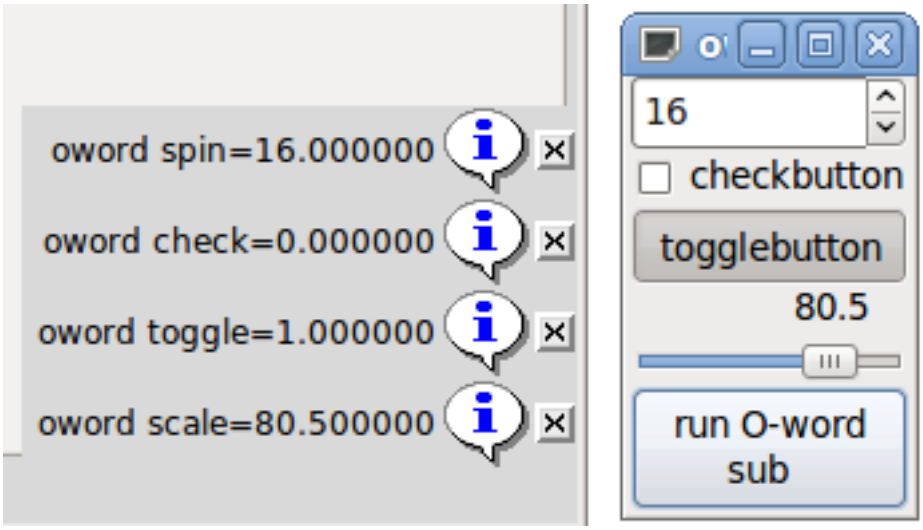
L'exemple de fichier UI est `configs/gladevcp/mdi-command-example/speed.ui`. Voici ce qui ce qui est obtenu en le lançant:



9.3.6.6 Un exemple plus avancé: Passer des paramètres à un sous-programme O-word

Il est parfaitement permis d'appeler un sous-programme O-word dans une commande MDI et passer la valeur des pins de HAL comme paramètres actuels. Un exemple de fichier UI est dans `configs/gladevcp/md`

Placer `configs/gladevcp/nc_files/oword.ngc` de sorte qu'Axis puisse le trouver, et lancer gladevcp owordsub.ui depuis un terminal. Ce qui devrait ressembler à celà:



9.3.6.7 Préparation d'une Action_MDI

L'interpréteur de G-code de LinuxCNC dispose d'un simple jeu de variables globales, comme la vitesse travail, la vitesse broche, le mode relatif/absolu et autres. Si on utilise des commandes G-code ou des sous-programmes O-word, certaines de ces variables doivent être modifiées par la commande ou le sous-programme. Par exemple, un sous-programme de sonde a très probablement besoin de définir la vitesse d'avance à une valeur très faible. Sans autres précautions, le réglage de vitesse précédent serait écrasé par la valeur du sous-programme de sonde.

Pour faire avec ce surprenant, autant qu'indésirable effet de bord produit par un sous-programme O-word ou un G-code exécuté avec une bascule Action MDI, le gestionnaire pré-MDI et post-MDI doit être associé avec une bascule Action_MDI donnée. Ces gestionnaires sont optionnels et fournissent une voie pour sauver tous les états avant d'exécuter l'action MDI et pour les restaurer ensuite aux valeurs précédentes. Les noms de signaux sont `mdi-command-start` et `mdi-command-stop`, les noms de gestionnaire peuvent être fixés dans Glade comme tout autre gestionnaire.

Voici un exemple, montrant comment la valeur de la vitesse d'avance est sauvée puis restaurée par de tels gestionnaires, noter que la commande LinuxCNC et le statut des voies sont disponibles comme `self.emc` et `self.stat` à travers la classe LinuxCNC_ActionBase:

```
    def on_mdi_command_start(self, action, userdata=None):
        action.stat.poll()
        self.start_feed = action.stat.settings[1]

    def on_mdi_command_stop(self, action, userdata=None):
        action.emc.mdi('F%.1f' % (self.start_feed))
        while action.emc.wait_complete() == -1:
            pass
```

Seule le widget de la bascule Action_MDI, supporte ces signaux.

Note
Dans une prochaine version de LinuxCNC, les nouveaux M-codes M70 à M72 seront disponibles, ils enregistreront l'état avant l'appel du sous-programme, la restauration de l'état au retour sera plus aisée.

9.3.6.8 Utiliser l'objet LinuxCNC Stat pour traiter les changements de statut

Beaucoup d'actions dépendent du statut de LinuxCNC, est-il en mode manuel, en mode MDI ou en mode auto ? Un programme est-il en cours d'exécution, est-il en pause ou au repos ? Il est impossible de lancer une commande MDI tant qu'un programme G-code est en cours d'exécution, cela doit donc être pris en compte. Beaucoup d'actions LinuxCNC prennent cela en compte d'elle même, les boutons et les options de menu sont désactivés quand leurs actions sont rendues impossibles.

Avec l'utilisation des gestionnaires d'événements Python, qui sont à un niveau inférieur aux Actions, on doit prendre soin de traiter les dépendances de statut soit-même. À cette fin, existe le widget LinuxCNC Stat, il associe les changements de statut de LinuxCNC avec les gestionnaires d'événements.

LinuxCNC Stat n'a pas de composant visible, il suffi de l'ajouter dans l'éditeur Glade. Une fois ajouté, vous pouvez associer des gestionnaires avec les signaux suivants:

- relatif au statut: émis quand l'arrêt d'urgence est activé, ou désactivé,
  - `state-estop` la machine est totalement arrêtée, puissance coupée.
  - `state-estop-reset` la machine passe à l'arrêt.
  - `state-on`, la machine est mise en marche
  - `state-off` la machine passe à l'arrêt.
- relatif au mode: émis quand LinuxCNC entre dans un de ces modes particuliers
  - `mode-manual`
  - `mode-mdi`
  - `mode-auto`
- relatif à l'interpréteur: émis quand l'interpréteur de G-code passe dans un de ces modes
  - `interp-run`
  - `interp-idle`
  - `interp-paused`
  - `interp-reading`
  - `interp-waiting`

9.3.7 Programmation de GladeVCP

9.3.7.1 Actions définies par l'utilisateur

La plupart des jeux de widgets, par le biais de l'éditeur Glade, supportent le concept de fonction de rappel, fonctions écrites par l'utilisateur, qui sont exécutées quand quelque chose arrive dans l'UI, événements tels que clics de souris, caractère tapé, mouvement de souris, événements d'horloge, fenêtre iconisée ou agrandie et ainsi de suite.

Les widgets de sortie HAL, typiquement, scrutent les événements de type entrée, tels qu'un bouton pressé, provoquant un changement de la valeur d'une pin HAL associée par le biais d'une telle fonction de rappel prédéfinie. Dans PyVCP, c'est réellement le seul type d'événement qui peut être défini à la main. Faire quelque chose de plus complexe, comme exécuter une commande MDI pour appeler un sous-programme G-code, n'est pas supporté.

Dans GladeVCP, les changement sur les pins de HAL sont juste un type de la classe générale d'événements (appelés signaux) dans GTK+. La plupart des widgets peuvent générer de tels signaux et l'éditeur de Glade supporte l'association de ces signaux avec une méthode Python ou nom de fonction.

Si vous décidez d'utiliser les actions définies par l'utilisateur, votre travail consistera à écrire un module Python dont la méthode, une fonction suffit dans les cas simples, peut être référencée à un gestionnaire d'événements dans Glade. GladeVCP fournit un moyen d'importer votre module au démarrage, il sera alors lié automatiquement au gestionnaire d'événements avec les signaux de widget comme un ensemble dans la description de l'éditeur Glade.

9.3.7.2 Un exemple: ajouter une fonction de rappel en Python

Ceci est juste un exemple minimal pour exprimer l'idée, les détails sont donnés dans le reste de cette section.

GladeVCP peut, non seulement manipuler ou afficher les pins de HAL, il est possible aussi d'écrire des gestionnaires d'événements en Python. Ce qui peut être utilisé, entre autre, pour exécuter des commandes MDI. Voici comment faire:

Écrire un module Python comme le suivant, et l'enregistrer sous le nom handlers.py

```
nhits = 0
def on_button_press(gtkobj,data=None):
    global nhits nhits += 1 gtkobj.set_label("hits: %d" % nhits)
```

Dans Glade, définir un bouton ou un bouton HAL, sélectionner l'onglet Signal, et dans les propriétés GtkButton sélectionner la ligne pressed. Entrer on_button_press ici, puis enregistrer le fichier Glade.

Ensuite, ajouter l'option -u handlers.py à la ligne de commande de gladevcp. Si les gestionnaires d'événements son répartis sur plusieurs fichiers, ajouter de multiples options -u <pynomfichier>.

Maintenant, presser le bouton devrait modifier son label car il est défini dans la fonction de rappel.

Que fait le drapeau `-u`: toutes les fonctions Python dans ce fichier sont collectées et configurées comme des gestionnaires de fonction de rappel potentiels pour les widgets Gtk, ils peuvent être référencés depuis l'onglet Signaux de Glade. Le gestionnaire de fonction de rappel est appelé avec l'instance de l'objet particulier comme paramètre, comme l'instance du GtkButton précédente, ainsi, il est possible d'appliquer n'importe quelle méthode GtkButton depuis ici.

Ou faire des choses plus utiles, par exemple, appeler une commande MDI!

9.3.7.3 L'événement valeur de HAL modifiée

Les widgets d'entrée HAL, comme la Led, ont l'état de leur pin de HAL (on/off), automatiquement associé avec l'apparence optique du widget (Led allumée/éteinte).

Au delà de cette fonctionnalité primitive, on peut associer n'importe quelle pin de HAL avec une fonction de rappel, y compris les widgets de HAL prédéfinis. Cela correspond bien avec la structure événementielle de l'application typique du widget: chaque activité, qu'elle soit un simple clic de souris, une touche pressée, une horloge expirée ou le changement de valeur d'une pin de HAL, générera une fonction de rappel et sera gérée par le même mécanisme.

Pour les pins de HAL définies par l'utilisateur, non associées à un widget de HAL particulier, le nom du signal est value-changed. Voir la section Ajouter des pins de HAL pour plus de détails.

Les widgets HAL sont fournis avec un signal prédéfini appelé hal-pin-changed. Voir la section sur les Widgets HAL pour d'autres détails.

9.3.7.4 Modèle de programmation

L'approche globale est la suivante:

- Concevoir l'interface graphique avec Glade, fixer les gestionnaires de signaux associés aux widgets action.
- Écrire un module Python qui contient des objets appelables (voir 'gestionnaire de modèles, plus loin)
- Passer le chemin du modules à gladevcp avec l'option -u <module>.
- gladevcp importe le module, inspecte les gestionnaires de signaux et les connecte à l'arbre des widgets.
- La boucle principale d'événements est exécutée.

Pour des tâches simple, il est suffisant de définir des fonctions nommées après les gestionnaires de signaux de Glade. Elles seront appelées quand l'événement correspondant se produira dans l'arbre des widgets. Voici un exemple très simple, il suppose que le signal pressed d'un bouton Gtk ou d'un bouton HAL est lié à une fonction de rappel appelée on_button_press:

```
nhits = 0
def on_button_press(gtkobj,data=None):
    global nhits
    nhits += 1
    gtkobj.set_label("hits: %d" % nhits)
```

Ajouter cette fonction dans un fichier Python et le lancer avec:

```
gladevcp -u <myhandler>.py mygui.ui
```

Noter que la communication entre les gestionnaires doit passer par des variables globales, qui s'adaptent mal est ne sont pas très "pythonique". C'est pourquoi nous en arrivons au gestionnaire de classes.

L'idée ici est la suivante: les gestionnaires sont liés aux méthodes de classe. La classe sous-jacente est instanciée et inspectée durant le démarrage de GladeVCP et liée à l'arbre des widgets comme gestionnaire de signaux. Donc, la tâche est maintenant d'écrire:

- Une ou plusieurs définitions de classe avec une ou plusieurs méthodes, dans un module ou répartis sur plusieurs modules.
- Une fonction get_handlers dans chaque module, qui retournera la liste des instances de classe à GladeVCP, leurs noms de méthode seront liés aux gestionnaires de signaux.

Voici un exemple minimaliste de module de gestionnaire définit par l'utilisateur:

```
class MyCallbacks :
    def on_this_signal(self,obj,data=None):
        print "this_signal happened, obj=",obj
    def get_handlers(halcomp,builder,useropts):
        return [MyCallbacks ()]
```

Maintenant, on_this_signal est disponible comme gestionnaire de signal dans l'arbre des widgets.

Si durant l'inspection du module GladeVCP trouve une fonction get_handlers, Il l'appelle de la manière suivante:

`get_handlers(halcomp,builder,useropts)`

Les arguments sont:

- halcomp - Se réfère au composant de HAL en construction.
- builder - arbre du widget - résulte de la lecture de la définition de l'UI (soit, en référence à un objet de type GtkBuilder ou de type libglade).
- useropts - Une liste de chaines collectée par l'option de la ligne de commande de gladevcp -U <useropts>.

GladeVCP inspecte alors la liste des instances de classe et récupère leurs noms. Les noms de méthode sont connectés à l'arbre des widgets comme gestionnaire de signaux. Seuls, les noms de méthode ne commençant pas par un _ (tiret bas) sont considérés.

Noter que peu importe si la libglade ou le nouveau format GtkBuilder est utilisé pour l'UI Glade, les widgets peuvent toujours être soumis au builder.get_object(<nomwidget>). En outre, la liste complète des widgets est disponible par builder.get_objects(), indépendamment du format de l'UI.

9.3.7.5 Séquence d'initialisation

Il est important de connaitre pour quoi faire, la fonction get_handlers() est appelée, et connaitre ce qui est sûr et ce qui ne l'est pas. Tout d'abord, les modules sont importés et initialisés dans leur ordre d'apparition sur la ligne de commande. Après le succès de l'importation, get_handlers() est appelé selon les étapes suivantes:

- L'arbre du widget est créé, mais pas encore réalisé (pas tant que le niveau supérieur window.show() n'aura pas été exécuté)
- Le composant de HAL, halcomp, est configuré et toutes les pins de HAL des widgets lui sont ajoutées.
- Il est sûr d'ajouter plus de pins de HAL parce-que halcomp.ready() n'a pas encore été appelé à ce point, ainsi, on peut ajouter ses propres pins, par exemple, dans la méthode de classe init().

Après que tous les modules ont été importés et que les noms des méthodes ont été extraits, les étapes suivantes se produisent:

- Tous les noms de méthode qualifiés seront connectés à l'arbre du widget avec connect_signals() ou signal_autoconnect() (selon le type de l'UI importée, format GtkBuilder ou l'ancien libglade).
- Le composant de HAL est finalisé avec halcomp.ready().
- Si un ID de fenêtre est passé comme argument, l'arbre du widget est re-apparenté pour démarrer dans cette fenêtre, et la fenêtre de niveau supérieur de Glade, window1 est abandonnée (voir la FAQ)
- Si un fichier de commandes de HAL, est passé avec -H halfile, il est exécuté avec halcmd.

- La boucle principal de Gtk est lancée.

Ainsi, lorsque le gestionnaire de classe est initialisé, tous les widgets sont existants mais pas encore réalisés (affichés à l'écran). Et le composant de HAL n'est pas prêt non plus, de sorte qu'il n'est pas sûr d'accéder aux valeurs des pins dans la méthode init().

Si on doit avoir une fonction de rappel à exécuter au démarrage du programme mais, après qu'il soit sûr d'accéder aux pins de HAL, alors connecter un gestionnaire au signal de la fenêtre de niveau supérieur réalisée, window1 (qui pourrait être sa seule raison d'être). A ce point, GladeVCP en a terminé avec toutes les configurations, le halfile a bien été lancé et GladeVCP est sur le point d'entrer dans la boucle principale Gtk.

9.3.7.6 Multiple fonctions de rappel avec le même nom

Dans une classe, les noms de méthode doivent être unique. Cependant, il est permis d'avoir de multiples instances de classe passées à GladeVCP par get_handlers() avec des méthodes portant le même nom. Lorsque le signal correspondant survient, les méthodes sont appelées dans l'ordre dans lequel elles ont été définies, module par module et dans un module, dans l'ordre des instances de classe retourné get_handlers().

9.3.7.7 Le drapeau GladeVCP -U <useropts>

Au lieu d'étendre GladeVCP à toutes les options concevables qui pourraient potentiellement être utilisées par un gestionnaire de classe, on peut utiliser le drapeau -U<useroption> (répétitivement si nécessaire). Ce drapeau collecte la liste des chaines de <useroption>. Cette liste est passée à la fonction get_handlers() (argument useropts). Le code est libre d'interpréter ces chaines comme bon lui semble. Un utilisation possible serait de les passer à la fonction exec de Python dans le get_handlers(), comme suit:

```
debug = 0
...
def get_handlers(halcomp,builder,useropts):
    ...
    global debug # suppose qu'il y a une variable globale
    pour cmd dans useropts:
        exec cmd in globals()
```

De cette façon, on peut passer des déclarations Python arbitraires au module grâce à l'option gladevcp -U, Par exemple:

```
gladevcp -U debug=42 -U "print 'debug=%d' % debug" ...
```

Debug devrait être mis à 2, et confirmer ce que le module fait actuellement.

9.3.7.8 Variables persistantes dans GladeVCP

Un aspect gênant de GladeVCP dans sa forme initiale avec pyvcp est le fait qu'on peut changer les valeurs des pins de HAL au travers du texte saisi, curseurs, bouton tournant, bouton à bascule etc, mais leurs paramètres ne sont pas enregistrés ni restaurés à la prochaine exécution de LinuxCNC. Ils commencent aux valeurs par défaut fixées dans le panneau ou la définition du widget.

GladeVCP dispose d'un mécanisme facile à utiliser pour enregistrer et restaurer l'état des widgets de HAL, ainsi que les variables du programme (en fait, n'importe quel attribut d'instance de type int, float, bool ou string).

Ce mécanisme utilise le format du populaire fichier .ini pour enregistrer et recharger les attributs persistants.

Imaginons renommer, ajouter ou supprimer des widgets dans Glade: un fichier .ini qui traîne depuis une version précédente du programme, ou une interface utilisateur entièrement différente, ne serait pas en mesure de restaurer correctement l'état des variables et des types puisqu'ils ont changé depuis.

GladeVCP détecte cette situation par la signature qui dépends de tous les noms d'objets et de types qui ont été enregistrés et qui doivent être restaurés. Dans le cas de signatures incompatibles, un nouveau fichier .ini avec la configuration pas défaut est généré.

9.3.7.9 Utilisation des variables persistantes

Pour que tous les états des widgets Gtk, que toutes les valeurs des pins de sortie des widget HAL et/ou que tous les attributs de classe du gestionnaire de classe soient conservés entre les invocations, procéder comme suit:

- Importer le module `gladevcp.persistence`.
- Décider quels attributs d'instance et leurs valeurs par défaut doivent être conservés, le cas échéant,
- décider quels widgets doivent avoir leur état conservé.
- Décrire ces décisions dans le gestionnaire de classe par la méthode `init()` grâce à un dictionnaire imbriqué comme suit:

```
def __init__(self, halcomp,builder,useropts):
    self.halcomp = halcomp
    self.builder = builder
    self.useropts = useropts
    self.defaults = {
        # les noms suivants seront enregistrés/restaurés comme attributs de méthode,
        # le mécanisme d'enregistrement/restauration est fortement typé,
        # les types de variables sont dérivés depuis le type de la valeur initiale.
        # les types couramment supportées sont: int, float, bool, string
        IniFile.vars : { 'nhits' : 0, 'a': 1.67, 'd': True ,'c' : "a string"},
        # pour enregistrer/restaurer l'état de tous les widgets pour lesquels
        # c'est sensé, ajouter cela:
        IniFile.widgets : widget_defaults(builder.get_objects())
        # une alternative sensée pourrait être de ne retenir que l'état de
        # tous les widgets de sortie HAL:
        # IniFile.widgets: widget_defaults(select_widgets(self.builder.get_objects(),
hal_only=True,output_only = True)),
    }
```

Puis associer un fichier .ini avec ce descripteur:

```
self.ini_filename = __name__ + '.ini'
self.ini = IniFile(self.ini_filename,self.defaults,self.builder)
self.ini.restore_state(self)
```

Ensuite restore_state(), aura automatiquement les attributs définis si ce qui suit a été exécuté:

```
self.nhits = 0
self.a = 1.67
self.d = True
self.c = "a string"
```

Noter que les types sont enregistrés et conservés lors de la restauration. Cet exemple suppose que le fichier .ini n'existe pas ou qu'il contient les valeurs par défaut depuis self.defaults.

Après cette incantation, on peut utiliser les méthodes IniFil suivantes:

ini.save_state(obj)
enregistre les attributs des objets depuis le dictionnaire IniFil.vars l'état du widget comme décrit par IniFile.widgets dans self.defaults

ini.create_default_ini()
crée un fichier .ini avec les valeurs par défaut

ini.restore_state(obj)
restaure les pins de HAL et les attributs des objets enregistrés/initialisés par défaut comme précédemment

Pour enregistrer le widget et/ou l'état des variables en quittant, connecter un gestionnaire de signal à la fenêtre de niveau supérieur `window1`, détruire l'événement:

```
def on_destroy(self,obj,data=None):
    self.ini.save_state(self)
```

La prochaine fois que l'application GladeVCP démarrera, les widgets doivent retrouver l'état qu'ils avaient à la fermeture de l'application.

9.3.7.10 Édition manuelle des fichiers .ini

Il est possible de faire cela, mais noter que les valeurs dans self.defaults écraseront votre édition si il y a erreur de frappe ou de syntaxe. Une erreur détectée, un message émis dans la console, donneront des indices sur ce qui s'est passé et le mauvais fichier ini sera renommé avec le suffixe .BAD. Après une mauvaise initialisation, les fichiers .BAD les plus anciens seront écrasés.

9.3.7.11 Ajouter des pins de HAL

Si il faut des pins de HAL non associées avec un widget HAL, les ajouter comme ci-dessous:

```
import hal_glib
...
# dans le gestionnaire de classe __init__():
self.example_trigger = hal_glib.GPin(halcomp.newpin('example-trigger', hal.HAL_BIT, hal. ↩
    HAL_IN))
```

Pour appeler une fonction de rappel quand la valeur de cette pin change il faut associer une fonction de rappel `value-changed` avec cette pin, ajouter pour cela:

```
self.example_trigger.connect('value-changed', self._on_example_trigger_change)
```

et définir une méthode de fonction de rappel (ou une fonction, dans ce cas laisser tomber le paramètre `self`):

```
# noter *_* - cette méthode n'est pas visible dans l'arbre du widget
def _on_example_trigger_change(self,pin,userdata=None):
    print "pin value changed to:" % (pin.get())
```

9.3.7.12 Ajout de timers

Depuis que GladeVCP utilise les widgets Gtk qui se rattachent sur les classes de base GObject, la totalité des fonctionnalités de la glib est disponible. Voici un exemple d' horloge de fonction de rappel:

```
def _on_timer_tick(self,userdata=None):
    ...
    return True # pour relancer l'horloge; return False pour un monostable
...
# démonstration d'une horloge lente en tâche de fond - la granularité est de une seconde
# pour une horloge rapide (granularité 1 ms), utiliser cela:
# glib.timeout_add(100, self._on_timer_tick,userdata) # 10Hz
glib.timeout_add_seconds(1, self._on_timer_tick)
```

9.3.7.13 Exemples, et lancez votre propre application GladeVCP

Visiter `linuxcnc/configs/gladevcp` pour des exemples prêt à l'emploi et points de départ de vos propres projets.

9.3.8 Questions & réponses

1. Je reçois un événement unmap inattendu dans ma fonction de gestionnaire juste après le démarrage, qu'est-ce que c'est?

   C'est la conséquence d'avoir dans votre fichier d'UI Glade la propriété de la fenêtre window1 visible fixée à True, il y a changement de parents de la fenêtre GladeVCP dans Axis ou touchy. L'arbre de widget de GladeVCP est créé, incluant une fenêtre de niveau supérieur puis re-aparenté dans Axis, laissant trainer les orphelins de la fenêtre de niveau supérieur. Pour éviter d'avoir cette fenêtre vide qui traine, elle est unmapped (rendue invisible) et la cause du signal unmap que vous avez eux. Suggestion pour fixer le problème: fixer window1.visible à False et ignorer le message initial d'événement unmap.

2. Mon programme GladeVCP démarre, mais aucune fenêtre n'apparait alors qu'elle devrait.

   La fenêtre allouée par Axis pour GladeVCP obtient la taille naturelle de tous ses enfants combinés. C'est au widget enfant a réclamer une taille (largeur et/ou hauteur). Cependant, toutes le fenêtres ne demandent pas une plus grande que 0, par exemple, le widget Graph dans sa forme courante. Si il y a un tel widget dans votre fichier Glade et que c'est lui qui défini la disposition vous devrez fixer sa largeur explicitement. Noter que la largeur et la hauteur de la fenêtre window1 dans Glade n'a pas de sens puisque cette fenêtre sera orpheline lors du changement de parent et donc sa géométrie n'aura aucun impact sur les mise en page (voir ci-dessus). La règle générale est la suivante: si vous exécutez manuellement un fichier UI avec gladevcp <fichierui> et que sa fenêtre a une géométrie raisonnable, elle devrait apparaitre correctement dans Axis.

3. Je veux une Led clignotante, alors j'ai coché une case pour la laisser clignoter avec un intervalle de 100ms. Elle devrait clignoter, mais je reçois un :Warning: value 0 le type gint est invalide ou hors de l'étendue pour les propriétés de led-blink-rate, c'est quoi le type gint?

   Il semble qu'il s'agisse d'un bug de Glade. Il faut re-saisir une valeur sur le champ de la fréquence de clignotement et enregistrer à nouveau. Ça a marché pour moi.

4. Mon panneau gladevcp ne marche pas dans Axis, il n'enregistre pas les états quand je ferme Axis, j'ai pourtant défini un gestionnaire on_destroy attaché au signal destroy de la fenêtre.

   Ce gestionnaire est très probablement lié à window1, qui en raison du changement de parent ne peux pas assurer cette fonction. Attachez le gestionnaire on_destroy handler au signal destroy d'une fenêtre intérieure. Par exemple: J'ai un notebook dans window1, attaché on_destroy au signal destroy de notebooks et ça marche bien. Il ne marcherait pas pour window1.

9.3.9 Troubleshooting

- make sure your have the development version of LinuxCNC installed. You don't need the axisrc file any more, this was mentioned in the old GladeVcp wiki page.
- run GladeVCP or Axis from a terminal window. If you get Python errors, check whether there's still a `/usr/lib/python2.6/dist-packages/hal.so` file lying around besides the newer `/usr/lib/python2.6/d` (note underscore); if yes, remove the `hal.so` file. It has been superseded by hal.py in the same directory and confuses the import mechanism.
- if you're using run-in-place, do a make clean to remove any accidentally left over hal.so file, then make.
- if you're using HAL_table or HAL_HBox widgets, be aware they have an HAL pin associated with it which is off by default. This pin controls whether these container's children are active or not.

9.3.10 Notes d'implémentation: la gestion des touches dans Axis

Nous pensons que la gestion des touches fonctionne bien, mais comme c'est un nouveau code, nous devons vous informer à ce propos pour que vous puissiez surveiller ces problèmes; S'il vous plaît, faites nous savoir si vous connaissez des erreurs ou des choses bizarres. Voici l'histoire:

Axis utilise le jeu de widget de TkInter. L'application GladeVCP utilise les widgets Gtk et démarre dans un contexte de processus différent. Ils sont attachés dans Axis avec le protocole Xembed. Ce qui permet à une application enfant comme GladeVCP de bien tenir proprement dans la fenêtre d'un parent et, en théorie, d'être intégrée au gestionnaire d'événements.

Toutefois, cela suppose que parent et enfant supportent tous les deux proprement le protocole Xembed, c'est le cas avec Gtk, pas avec TkInter. Une conséquence de cela, c'est que certaines touches ne sont pas transmises correctement dans toutes les circonstances depuis un panneau GladeVCP vers Axis. Une d'elle est la touche Entrée. Ou quand le widget SpinButton a le focus, dans ce cas, par exemple la touche Échap n'est pas bien transmise à Axis et cause un abandon avec des conséquences potentiellement désastreuses.

Par conséquent, les événements touches dans GladeVCP, sont traités explicitement, et sélectivement transmises à Axis, pour assurer que de telles situations ne puissent pas survenir. Pour des détails, voir la fonction keyboard_forward() dans la lib/python/gladevcp/xembed.py.

Chapter 10

User Interfaces

10.1 Halui Examples

For any Halui examples to work you need to add the following line to the [HAL] section of the ini file.

```
HALUI = halui
```

10.1.1 Remote Start

To connect a remote program start button to LinuxCNC you use the `halui.program.run` pin and the `halui.mode.auto` pin. You have to insure that it is OK to run first by using the `halui.mode.is-auto` pin. You do this with an `and2` component. The following figure shows how this is done. When the Remote Run Button is pressed it is connected to both `halui.mode.auto` and `and2.0.in0`. If it is OK for auto mode the pin `halui.mode.is-auto` will be on. If both the inputs to the `and2.0` component are on the `and2.0.out` will be on and this will start the program.

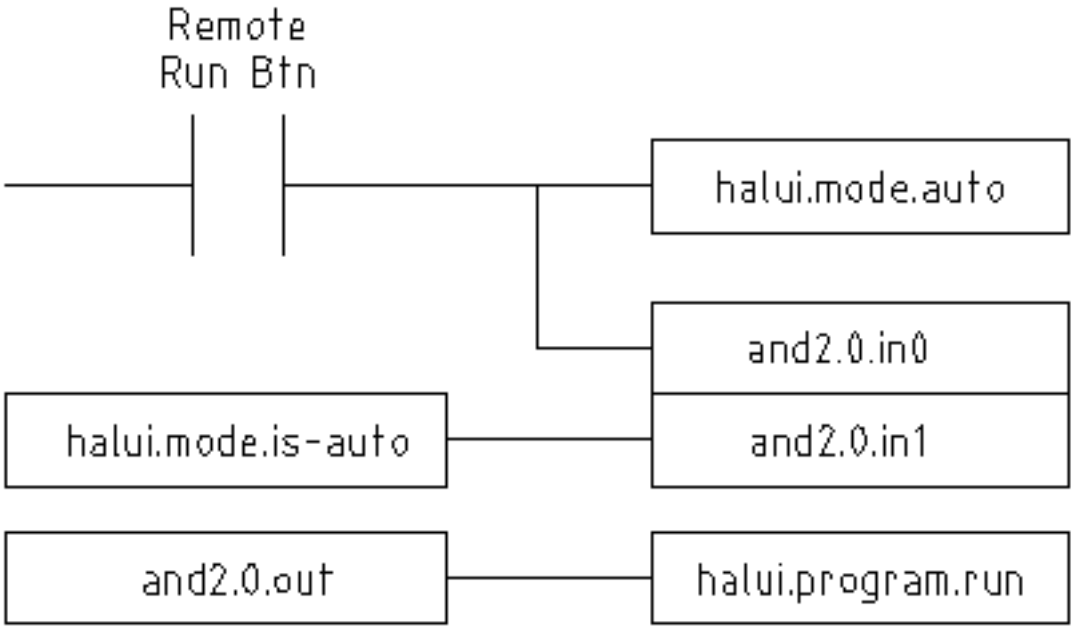


Figure 10.1: Remote Start Example

The hal commands needed to accomplish the above are:

```
net program-start-btn halui.mode.auto and2.0.in0 <= <your input pin>
net program-run-ok and2.0.in1 <= halui.mode.is-auto
net remote-program-run halui.program.run <= and2.0.out
```

Notice on line one that there are two reader pins, this can also be split up to two lines like this:

```
net program-start-btn halui.mode.auto <= <your input pin>
net program-start-btn and2.0.in0
```

10.1.2 Pause & Resume

This example was developed to allow LinuxCNC to move a rotary axis on a signal from an external machine. The coordination between the two systems will be provided by two Halui components:

- halui.program.is-paused
- halui.program.resume

In your customized hal file, add the following two lines that will be connected to your I/O to turn on the program pause or to resume when the external system wants LinuxCNC to continue.

```
net ispaused halui.program.is paused => ”your output pin”
net resume halui.program.resume <= ”your input pin”
```

Your input and output pins are connected to the pins wired to the other controller. They may be parallel port pins or any other I/O pins that you have access to.

This system works in the following way. When an M0 is encountered in your G-code, the `halui.program.is-p` signal goes true. This turns on your output pin so that the external controller knows that LinuxCNC is paused.

To resume the LinuxCNC G-code program, when the external controller is ready it will make its output true. This will signal LinuxCNC that it should resume executing G-code.

Difficulties in timing

- The ”resume” input return signal should not be longer than the time required to get the G-code running again.
- The ”is-paused” output should no longer be active by the time the ”resume” signal ends.

These timing problems could be avoided by using ClassicLadder to activate the ”is-paused” output via a monostable timer to deliver one narrow output pulse. The ”resume” pulse could also be received via a monostable timer.

Chapter 11

Drivers

11.1 Port parallèle

11.1.1 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, contrôles et états. Le groupe données consiste en 8 broches de sortie, le groupe contrôles consiste en 4 broches et le groupe états consiste en 5 broches d'entrée.

Au début des années 1990, le port parallèle bidirectionnel est arrivé, ce qui a permis 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 entrées ou en sorties. Si il est configuré en sorties, un port fournit un total de 12 sorties et 5 entrées. Si il est configuré en entrées, 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 collecteurs ouverts, le mode x de HAL permet un usage plus flexible avec 8 sorties dédiées, 5 entrées dédiées et 4 broches en collecteurs ouverts. Dans d'autres ports parallèles, les broches du groupe contrôles sont en push-pull et ne peuvent pas être utilisées comme des entrées.

HAL et les collecteurs ouverts
HAL ne peut pas déterminer automatiquement si les broches en mode bidirectionnel x sont effectivement en collecteurs ouverts. 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 un 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.

Aucune autre combinaison n'est supportée. Un port ne peut plus être modifié pour passer d'entrées en sorties une fois le pilote installé. La figure des diagrammes blocs affiche deux diagrammes, un

montre le pilote quand le groupe de données est configuré en sorties et le second le montre configuré en entrées.

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

11.1.1.1 Chargement de hal_parport

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

11.1.1.2 Utiliser l'index du port

Les adresses d'E/S inférieures à 16 sont traitées comme les index de port. C'est la manière la plus simple d'installer le pilote parport en coopération avec le pilote de Linux parport_pc si il est chargé.

```
loadrt hal_parport cfg="0"
```

Utilisera l'adresse que Linux a détecté pour parport0.

11.1.1.3 Utiliser l'adresse du port

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 défaut configuré en sorties. 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 0x278 ou 0x378 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 noayau 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, c'est une erreur.

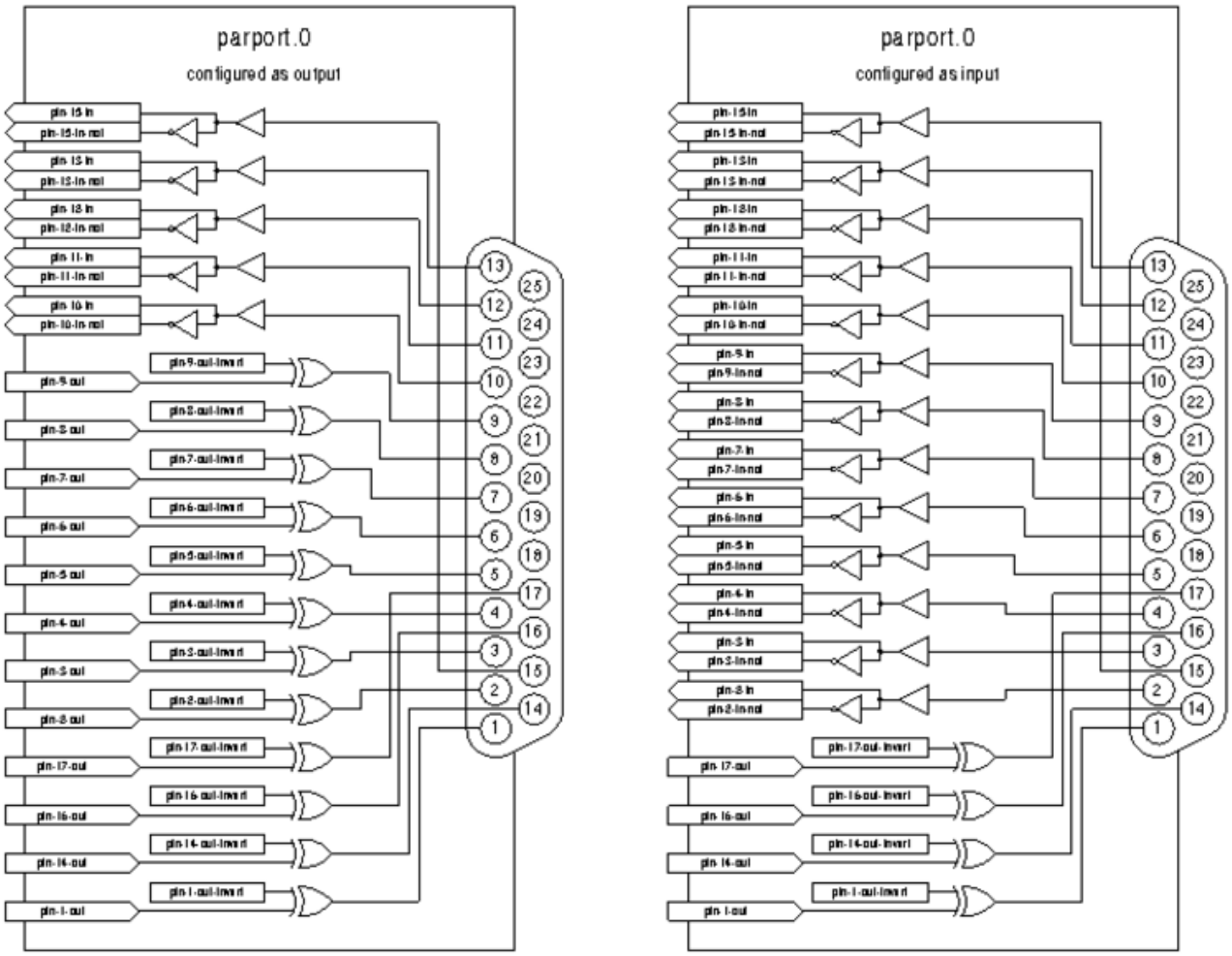


Figure 11.1: Diagrammes blocs de parport

11.1.1.4 Pins

- (bit) parport.<portnum>.pin-<pinnum>-out — Pilote une broche de sortie physique. output pin.
- (bit) parport.<portnum>.pin-<pinnum>-in — Suit une broche d'entrée physique. pin.
- (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-out. Les pins 2 jusqu'à 9 font partie du groupe données, elles sont des pins de sortie si le port est défini comme un port de sortie (par défaut, port de sortie). Les broches 1, 14, 16 et 17 sont des sorties dans tous les modes. Ces pins 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 inversée, elle est FAUSSE si la pin physique est haute. En connectant un signal à l'une ou à l'autre,

l'utilisateur peut décider de la logique de l'entrée. En mode x, les pins 1, 14, 16 et 17 sont également des pins d'entrée.

11.1.1.5 Paramètres

- (bit) parport.<portnum>.pin-<pinnum>-out-invert — Inverse une pin de sortie.
- (bit) parport.<portnum>.pin-<pinnum>-out-reset — (seulement pour 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 réinitialisée 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.

11.1.1.6 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 reset-time soit écoulé depuis la dernière écriture associée write puis 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 individuelles 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.

11.1.1.7 Problème courant

Si, au chargement du module un message du genre suivant apparait:

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

s'assurer que le module du kernel standard, parport_pc, n'est pas chargé et qu'aucun périphérique dans le système ne revendique les ports concernés. [1]

Si le module est chargé mais ne semble pas fonctionner, l'adresse du port est incorrecte ou le module probe_parport est revendiqué par un autre périphérique.

[1] Dans le paquetage LinuxCNC pour Ubuntu, le fichier /etc/modprobe.d/linuxcnc empêche normalement que parport_pc soit chargé automatiquement.

11.1.1.8 Utiliser DoubleStep

Pour activer DoubleStep sur un port parallèle, il faut ajouter la fonction parport.n.reset après parport.n.write et configurer stepspace à 0 ainsi que le reset-time souhaité. Alors ce pas pourra être positionné à chaque période dans HAL, puis voir son état basculé par parport après été positionné pendant le temps spécifié par parport.n.reset-time.

Par exemple:

```
loadrt hal_parport cfg="0x378 out"
setp parport.0.reset-time 5000
loadrt stepgen step_type=0,0,0
addf parport.0.read base-thread
addf stepgen.make-pulses base-thread
addf parport.0.write base-thread
addf parport.0.reset base-thread
addf stepgen.capture-position servo-thread
...
setp stepgen.0.steplen 1
setp stepgen.0.stepspace 0
```

11.1.2 probe_parport

Dans les PC actuels, les ports parallèles peuvent requérir une configuration plug and play (PNP) avant qu'ils ne puissent être utilisés. Le module de noyau probe_parport effectue la configuration de tous les port PNP présents. Il doit être chargé avant hal_parport. Sur les machines sans port PNP, il peut être chargé mais restera sans effet.

11.1.2.1 Installer probe_parport

```
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.

11.2 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. 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.

11.2.1 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.

11.2.2 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.

11.2.3 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.

11.2.4 Functions

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

11.3 Variateur de fréquence GS2

Composant de HAL pour la série de variateurs de fréquence GS2 fournie par la société Automation Direct. [2]

[2] En Europe on trouve l’équivalent sous la marque Omron.

11.3.1 Chargement du composant

- Ce composant est chargé en utilisant la commande suivante:

```
loadusr -Wn spindle-vfd gs2_vfd -n spindle-vfd
```

La commande de HAL loadusr est détaillée au chapitre: loadusr.

11.3.2 Options spécifiques au chargement

Les options spécifiques au chargement du composant gs2_vfd:

- -b ou --bits <n> (défaut 8) Fixe le nombre de bits de donnée à <n>, dans lequel <n> doit être compris entre 5 et 8 inclus.
- -d ou --device <path> (défaut /dev/ttyS0) Fixe le nom de la liaison série à utiliser.
- -g ou --debug Active les messages de débogage. Le drapeau du mode verbeux pourra être activé. Le débogage affichera tous les messages modbus en hexadécimal sur terminal.
- -n ou --name <string> (défaut gs2_vfd) Fixe le nom du composant de HAL à <string>, les noms de toutes ses pins et paramètres commenceront également par <string>.
- -p ou --parity {even, odd, none} (défaut odd) Fixe la parité de la liaison série à parité paire, parité impaire ou sans parité.
- -r ou --rate <n> (défaut 38400) Fixe le débit de la liaisons à <n>. C'est une erreur si le débit n'est pas une des valeurs suivantes: 110, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200.
- -s ou --stopbits {1,2} (défaut 1) Fixe le nombre de bits de stop de la liaison série à 1 ou 2.
- -t ou --target <n> (défaut 1) Fixe le nombre de cibles MODBUS (esclaves). Doit correspondre au nombre de périphériques réglé dans le GS2.
- -v ou --verbose Active les messages de débogage. Noter qu'en cas d'erreurs série, cela ne fera pas beaucoup de différence ce qui peut être gênant.

11.3.3 Consignes de dialogue avec le variateur

Les valeurs <name> sont les noms donnés par l'option -n durant la phase de chargement du composant.

- <name>.DC-bus-volts (float, out) La tension du bus DC sur le variateur.
- <name>.at-speed (bit, out) Quand la consigne vitesse est atteinte.
- <name>.err-reset (bit, in) Envoi d'un reset errors au variateur.
- <name>.firmware-revision (s32, out) envoyé par le variateur.
- <name>.frequency-command (float, out) envoyé par le variateur.
- <name>.frequency-out (float, out) envoyé par le variateur.
- <name>.is-stopped (bit, out) when the VFD reports 0 Hz output.
- <name>.load-percentage (float, out) envoyé par le variateur.

- <name>.motor-RPM (float, out) envoyé par le variateur.
- <name>.output-current (float, out) envoyé par le variateur.
- <name>.output-voltage (float, out) envoyé par le variateur.
- <name>.power-factor (float, out) envoyé par le variateur.
- <name>.scale-frequency (float, out) envoyé par le variateur.
- <name>.speed-command (float, in) Consigne vitesse envoyée. au variateur en $tr.mn^{-1}$. C'est une erreur d'envoyer une consigne de vitesse supérieure à la valeur maximum réglée dans le variateur.
- <name>.spindle-fwd (bit, in) Sens de rotation envoyé au variateur, 1 pour le sens horaire et 0 pour le sens anti-horaire.
- <name>.spindle-rev (bit, in) 1 pour marche en sens anti-horaire et 0 pour ARRÊT.
- <name>.spindle-on (bit, in) 1 pour MARCHE et 0 pour ARRÊT du variateur.
- <name>.status-1 (s32, out) Drive Status du VFD (voir le manuel du GS2).
- <name>.status-2 (s32, out) Drive Status du VFD (voir le manuel du GS2). Note: la valeur est la somme de tous les bits à 1. Ainsi, 163 signifie que le pilote est dans le mode de marche qui est la somme de:
  - 3 (marche)
  - + 32 (fréquence fixée par liaison série)
  - +128 (opération fixée par liaison série).

11.3.4 Paramètres de réglage du variateur

Les valeurs <name> sont les noms donnés par l'option -n durant la phase de chargement du composant.

- <name>.error-count (s32, RW)
- <name>.loop-time (float, RW) Nombre d'interrogation d modbus (défaut 0.1).
- <name>.nameplate-HZ (float, RW) Vitesse plaquée du moteur en Hz (défaut 50).
- <name>.nameplate-RPM (float, RW) Vitesse plaquée du moteur en $tr.mn^{-1}$ (défaut 1500).
- <name>.retval (s32, RW) la valeur de retour d'une erreur dans HAL.
- <name>.tolerance (s32, RW) Tolérance en vitesse (défaut 0.01).

Un exemple d'utilisation d'un variateur de fréquence pour piloter une broche est donné dans le manuel de l'intégrateur au chapitre Exemples: utiliser un GS2.

11.4 Mesa HostMot2

11.4.1 Introduction

HostMot2 is an FPGA configuration developed by Mesa Electronics for their line of "Anything I/O" motion control cards. The firmware is open source, portable and flexible. It can be configured (at compile-time) with zero or more instances (an object created at runtime) of each of several Modules: encoders (quadrature counters), PWM generators, and step/dir generators. The firmware can be configured (at run-time) to connect each of these instances to pins on the I/O headers. I/O pins not driven by a Module instance revert to general-purpose bi-directional digital I/O.

11.4.2 Firmware Binaries

Several pre-compiled HostMot2 firmware binaries are available for the different Anything I/O boards. (This list is incomplete, check the hostmot2-firmware distribution for up-to-date firmware lists.)

- 3x20 (144 I/O pins): using hm2_pci module
  - 24-channel servo
  - 16-channel servo plus 24 step/dir generators
- 5i22 (96 I/O pins): using hm2_pci module
  - 16-channel servo
  - 8-channel servo plus 24 step/dir generators
- 5i20, 5i23, 4i65, 4i68 (72 I/O pins): using hm2_pci module
  - 12-channel servo
  - 8-channel servo plus 4 step/dir generators
  - 4-channel servo plus 8 step/dir generators
- 7i43 (48 I/O pins): using hm2_7i43 module
  - 8-channel servo (8 PWM generators & 8 encoders)
  - 4-channel servo plus 4 step/dir generators

11.4.3 Installing Firmware

Depending on how you installed LinuxCNC you may have to open the Synaptic Package Manager from the System menu and install the package for your Mesa card. The quickest way to find them is to do a search for ”hostmot2” in the Synaptic Package Manager. Mark the firmware for installation, then apply.

11.4.4 Loading HostMot2

The LinuxCNC support for the HostMot2 firmware is split into a generic driver called ”hostmot2” and two low-level I/O drivers for the Anything I/O boards. The low-level I/O drivers are ”hm2_7i43” and ”hm2_pci” (for all the PCI- and PC-104/Plus-based Anything I/O boards). The hostmot2 driver must be loaded first, using a HAL command like this:

```
loadrt hostmot2
```

See the hostmot2(9) man page for details.

The hostmot2 driver by itself does nothing, it needs access to actual boards running the HostMot2 firmware. The low-level I/O drivers provide this access. The low-level I/O drivers are loaded with commands like this:

```
loadrt hm2_pci config=”firmware=hm2/5i20/SVST8_4.BIT num_encoders=3 num_pwmgens=3  ↩
    num_stepgens=1”
```

The config parameters are described in the hostmot2 man page.

11.4.5 Watchdog

The HostMot2 firmware may include a watchdog Module; if it does, the hostmot2 driver will use it.

The watchdog must be petted by LinuxCNC periodically or it will bite.

When the watchdog bites, all the board's I/O pins are disconnected from their Module instances and become high-impedance inputs (pulled high), and all communication with the board stops. The state of the HostMot2 firmware modules is not disturbed (except for the configuration of the I/O Pins). Encoder instances keep counting quadrature pulses, and pwm- and step-generators keep generating signals (which are not relayed to the motors, because the I/O Pins have become inputs).

Resetting the watchdog resumes communication and resets the I/O pins to the configuration chosen at load-time.

If the firmware includes a watchdog, the following HAL objects will be exported:

11.4.5.1 Pins:

has_bit
(bit i/o) True if the watchdog has bit, False if the watchdog has not bit. If the watchdog has bit and the has_bit bit is True, the user can reset it to False to resume operation.

11.4.5.2 Parameters:

timeout_ns
(u32 read/write) Watchdog timeout, in nanoseconds. This is initialized to 1,000,000,000 (1 second) at module load time. If more than this amount of time passes between calls to the hm2 write function, the watchdog will bite.

11.4.6 HostMot2 Functions

hm2_<BoardType>.<BoardNum>.read
Read all inputs, update input HAL pins.

hm2_<BoardType>.<BoardNum>.write
Write all outputs.

hm2_<BoardType>.<BoardNum>.pet-watchdog
Pet the watchdog to keep it from biting us for a while.

hm2_<BoardType>.<BoardNum>.read_gpio
Read the GPIO input pins only. (This function is not available on the 7i43 due to limitations of the EPP bus.)

hm2_<BoardType>.<BoardNum>.write_gpio
Write the GPIO control registers and output pins only. (This function is not available on the 7i43 due to limitations of the EPP bus.)

Note
The above **read_gpio** and **write_gpio** functions should not normally be needed, since the GPIO bits are read and written along with everything else in the standard **read** and **write** functions above, which are normally run in the servo thread.
The **read_gpio** and **write_gpio** functions were provided in case some very fast (frequently updated) I/O is needed. These functions should be run in the base thread. If you have need for this, please send an email and tell us about it, and what your application is.

11.4.7 Pinouts

The hostmot2 driver does not have a particular pinout. The pinout comes from the firmware that the hostmot2 driver sends to the Anything I/O board. Each firmware has different pinout, and the pinout depends on how many of the available encoders, pwmgens, and stepgens are used. To get a pinout list for your configuration after loading LinuxCNC in the terminal window type:

```
dmesg > hm2.txt
```

The resulting text file will contain lots of information as well as the pinout for the HostMot2 and any error and warning messages.

To reduce the clutter by clearing the message buffer before loading LinuxCNC type the following in the terminal window:

```
sudo dmesg -c
```

Now when you run LinuxCNC and then do a ”dmesg > hm2.txt” in the terminal only the info from the time you loaded LinuxCNC will be in your file along with your pinout. The file will be in the current directory of the terminal window. Each line will contain the card name, the card number, the I/O Pin number, the connector and pin, and the usage. From this printout you will know the physical connections to your card based on your configuration.

An example of a 5i20 configuration:

```
[HOSTMOT2]
DRIVER=hm2_pci
BOARD=5i20
CONFIG=”firmware=hm2/5i20/SVST8_4.BIT num_encoders=1 num_pwmgens=1 num_stepgens=3”
```

The above configuration produced this printout.

```
[ 1141.053386] hm2/hm2_5i20.0: 72 I/O Pins used:
[ 1141.053394] hm2/hm2_5i20.0: IO Pin 000 (P2-01): IOPort
[ 1141.053397] hm2/hm2_5i20.0: IO Pin 001 (P2-03): IOPort
[ 1141.053401] hm2/hm2_5i20.0: IO Pin 002 (P2-05): Encoder #0, pin B (Input)
[ 1141.053405] hm2/hm2_5i20.0: IO Pin 003 (P2-07): Encoder #0, pin A (Input)
[ 1141.053408] hm2/hm2_5i20.0: IO Pin 004 (P2-09): IOPort
[ 1141.053411] hm2/hm2_5i20.0: IO Pin 005 (P2-11): Encoder #0, pin Index (Input)
[ 1141.053415] hm2/hm2_5i20.0: IO Pin 006 (P2-13): IOPort
[ 1141.053418] hm2/hm2_5i20.0: IO Pin 007 (P2-15): PWMGen #0, pin Out0
(PWM or Up) (Output)
[ 1141.053422] hm2/hm2_5i20.0: IO Pin 008 (P2-17): IOPort
[ 1141.053425] hm2/hm2_5i20.0: IO Pin 009 (P2-19): PWMGen #0, pin Out1
(Dir or Down) (Output)
[ 1141.053429] hm2/hm2_5i20.0: IO Pin 010 (P2-21): IOPort
[ 1141.053432] hm2/hm2_5i20.0: IO Pin 011 (P2-23): PWMGen #0, pin
Not-Enable (Output)
    <snip>...
[ 1141.053589] hm2/hm2_5i20.0: IO Pin 060 (P4-25): StepGen #2, pin Step (Output)
[ 1141.053593] hm2/hm2_5i20.0: IO Pin 061 (P4-27): StepGen #2, pin Direction (Output)
[ 1141.053597] hm2/hm2_5i20.0: IO Pin 062 (P4-29): StepGen #2, pin (unused) (Output)
[ 1141.053601] hm2/hm2_5i20.0: IO Pin 063 (P4-31): StepGen #2, pin (unused) (Output)
[ 1141.053605] hm2/hm2_5i20.0: IO Pin 064 (P4-33): StepGen #2, pin (unused) (Output)
[ 1141.053609] hm2/hm2_5i20.0: IO Pin 065 (P4-35): StepGen #2, pin (unused) (Output)
[ 1141.053613] hm2/hm2_5i20.0: IO Pin 066 (P4-37): IOPort
[ 1141.053616] hm2/hm2_5i20.0: IO Pin 067 (P4-39): IOPort
[ 1141.053619] hm2/hm2_5i20.0: IO Pin 068 (P4-41): IOPort
[ 1141.053621] hm2/hm2_5i20.0: IO Pin 069 (P4-43): IOPort
[ 1141.053624] hm2/hm2_5i20.0: IO Pin 070 (P4-45): IOPort
[ 1141.053627] hm2/hm2_5i20.0: IO Pin 071 (P4-47): IOPort
[ 1141.053811] hm2/hm2_5i20.0: registered
[ 1141.053815] hm2_5i20.0: initialized AnyIO board at 0000:02:02.0
```

Note that the I/O Pin nnn will correspond to the pin number shown on the HAL Configuration screen for GPIOs. Some of the Stepgen, Encoder and PWMGen will also show up as GPIOs in the HAL Configuration screen.

11.4.8 PIN Files

The default pinout is described in a .PIN file (human-readable text). When you install a firmware package .deb, the .PIN files are installed in

`/usr/share/doc/hostmot2-firmware-<board>/`

11.4.9 Firmware

The selected firmware (.BIT file) and configuration is uploaded from the PC motherboard to the Mesa mothercard on LinuxCNC startup. If you are using Run In Place, you must still install a hostmot2-firmware-<board> package. There is more information about firmware and configuration in the ”Configurations” section.

11.4.10 HAL Pins

The HAL pins for each configuration can be seen by opening up ”Show HAL Configuration” from the Machine menu. All the HAL pins and parameters can be found there. The following figure is of the 5i20 configuration used above.

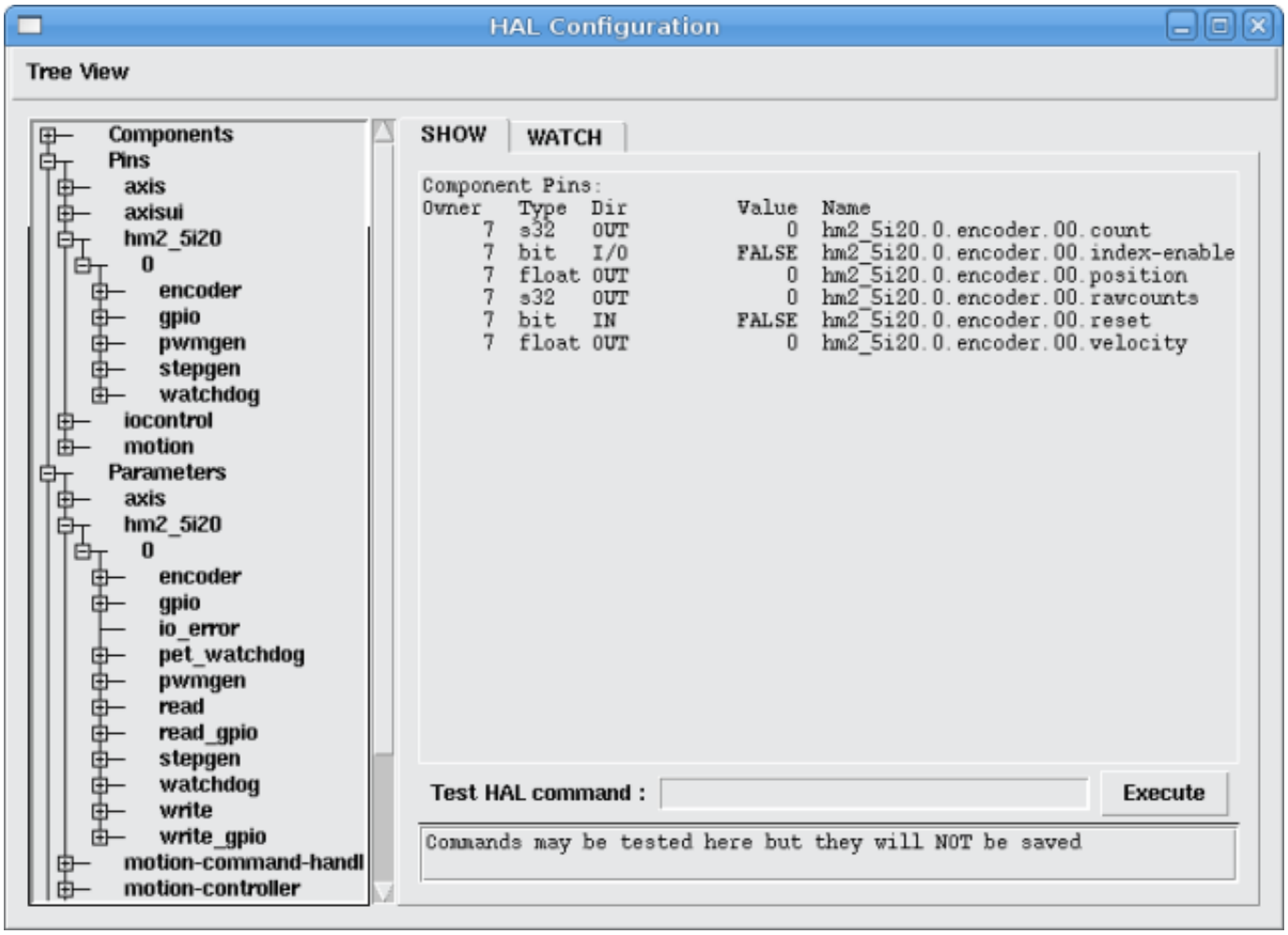


Figure 11.2: 5i20 HAL Pins

11.4.11 Configurations

The Hostmot2 firmware is available in several versions, depending on what you are trying to accomplish. You can get a reminder of what a particular firmware is for by looking at the name. Let's look at a couple of examples.

In the 7i43 (two ports), SV8 ("Servo 8") would be for having 8 servos or fewer, using the "classic" 7i33 4-axis (per port) servo board. So 8 servos would use up all 48 signals in the two ports. But if you only needed 3 servos, you could say `num_encoders=3` and `num_pwmgens=3` and recover 5 servos at 6 signals each, thus gaining 30 bits of GPIO.

Or, in the 5i22 (four ports), SVST8_24 ("Servo 8, Stepper 24") would be for having 8 servos or fewer (7i33 x2 again), and 24 steppers or fewer (7i47 x2). This would use up all four ports. If you only needed 4 servos you could say `num_encoders=4` and `num_pwmgens=4` and recover 1 port (and save a 7i33). And if you only needed 12 steppers you could say `num_stepgens=12` and free up one port (and save a 7i47). So in this way we can save two ports (48 bits) for GPIO.

Here are tables of the firmwares available in the official packages. There may be additional firmwares available at the Mesanet.com website that have not yet made it into the LinuxCNC official firmware packages, so check there too.

3x20 (6-port various) Default Configurations (The 3x20 comes in 1M, 1.5M, and 2M gate versions. So far, all firmware is available in all gate sizes.)

Firmware	Encoder	PWMGen	StepGen	GPIO
SV24	24	24	0	0
SVST16_24	16	16	24	0

5i22 (4-port PCI) Default Configurations (The 5i22 comes in 1M and 1.5M gate versions. So far, all firmware is available in all gate sizes.)

Firmware	Encoder	PWM	StepGen	GPIO
SV16	16	16	0	0
SVST2_4_7I47	4	2	4	72
SVST8_8	8	8	8	0
SVST8_24	8	8	24	0

5i23 (3-port PCI) Default Configurations (The 5i23 has 400k gates.)

Firmware	Encoder	PWM	StepGen	GPIO
SV12	12	12	0	0
SVST2_8	2	2	8 (tbl5)	12
SVST2_4_7I47	4	2	4	48
SV12_2X7I48_72	12	12	0	24
SV12IM_2X7I48_72	12 (+IM)	12	0	12
SVST4_8	4	4	8 (tbl5)	0
SVST8_4	8	8	4 (tbl5)	0
SVST8_4IM2	8 (+IM)	8	4	8
SVST8_8IM2	8 (+IM)	8	8	0
SVTP6_7I39	6	0 (6 BLDC)	0	0

5i20 (3-port PCI) Default Configurations (The 5i20 has 200k gates.)

Firmware	Encoder	PWM	StepGen	GPIO
SV12	12	12	0	0
SVST2_8	2	2	8 (tbl5)	12
SVST2_4_7I47	4	2	4	48
SV12_2X7I48_72	12	12	0	24
SV12IM_2X7I48_72	12 (+IM)	12	0	12
SVST8_4	8	8	4 (tbl5)	0
SVST8_4IM2	8 (+IM)	8	4	8

4i68 (3-port PC/104) Default Configurations (The 4i68 has 400k gates.)

Firmware	Encoder	PWM	StepGen	GPIO
SV12	12	12	0	0
SVST2_4_7I47	4	2	4	48
SVST4_8	4	4	8	0
SVST8_4	8	8	4	0
SVST8_4IM2	8 (+IM)	8	4	8
SVST8_8IM2	8 (+IM)	8	8	0

.

.

4i65 (3-port PC/104) Default Configurations (The 4i65 has 200k gates.)

Firmware	Encoder	PWM	StepGen	GPIO
SV12	12	12	0	0
SVST8_4	8	8	4	0
SVST8_4IM2	8 (+IM)	8	4	8

7i43 (2-port parallel) 400k gate versions, Default Configurations

Firmware	Encoder	PWM	StepGen	GPIO
SV8	8	8	0	0
SVST4_4	4	4	4 (tbl5)	0
SVST4_6	4	4	6 (tbl3)	0
SVST4_12	4	4	12	0
SVST2_4_7I47	4	2	4	24

7i43 (2-port parallel) 200k gate versions, Default Configurations

Firmware	Encoder	PWM	StepGen	GPIO
SV8	8	8	0	0
SVST4_4	4	4	4 (tbl5)	0
SVST4_6	4	4	6 (tbl3)	0
SVST2_4_7I47	4	2	4	24

Even though several cards may have the same named .BIT file you cannot use a .BIT file that is not for that card. Different cards have different clock frequencies so make sure you load the proper .BIT file for your card. Custom hm2 firmwares can be created for special applications and you may see some custom hm2 firmwares in the directories with the default ones.

When you load the board-driver (hm2_pci or hm2_7i43), you can tell it to disable instances of the three primary modules (pwmgen, stepgen, and encoder) by setting the count lower. Any I/O pins belonging to disabled module instances become GPIOs.

11.4.12 GPIO

General Purpose I/O pins on the board which are not used by a module instance are exported to HAL as ”full” GPIO pins. Full GPIO pins can be configured at run-time to be inputs, outputs, or open drains, and have a HAL interface that exposes this flexibility. I/O pins that are owned by an active module instance are constrained by the requirements of the owning module, and have a restricted HAL interface.

GPIOs have names like ”hm2_<BoardType>.<BoardNum>.gpio.<IONum>.” IONum. is a three-digit number. The mapping from IONum to connector and pin-on-that-connector is written to the syslog when the driver loads, and it’s documented in Mesa’s manual for the Anything I/O boards.

The hm2 GPIO representation is modeled after the Digital Inputs and Digital Outputs described in the Canonical Device Interface (part of the HAL General Reference document).

GPIO pins default to input.

11.4.12.1 Pins

in

(Bit, Out) Normal state of the hardware input pin. Both full GPIO pins and I/O pins used as inputs by active module instances have this pin.

in_not
(Bit, Out) Inverted state of the hardware input pin. Both full GPIO pins and I/O pins used as inputs by active module instances have this pin.

out
(Bit, In) Value to be written (possibly inverted) to the hardware output pin. Only full GPIO pins have this pin.

11.4.12.2 Parameters

invert_output
(Bit, RW) This parameter only has an effect if the ”is_output” parameter is true. If this parameter is true, the output value of the GPIO will be the inverse of the value on the ”out” HAL pin. Only full GPIO pins and I/O pins used as outputs by active module instances have this parameter. To invert an active module pin you have to invert the GPIO pin not the module pin.

is_opendrain
(Bit, RW) This parameter only has an effect if the ”is_output” parameter is true. If this parameter is false, the GPIO behaves as a normal output pin: the I/O pin on the connector is driven to the value specified by the ”out” HAL pin (possibly inverted), and the value of the ”in” and ”in_not” HAL pins is undefined. If this parameter is true, the GPIO behaves as an open-drain pin. Writing 0 to the ”out” HAL pin drives the I/O pin low, writing 1 to the ”out” HAL pin puts the I/O pin in a high-impedance state. In this high-impedance state the I/O pin floats (weakly pulled high), and other devices can drive the value; the resulting value on the I/O pin is available on the ”in” and ”in_not” pins. Only full GPIO pins and I/O pins used as outputs by active module instances have this parameter.

is_output
(Bit, RW) If set to 0, the GPIO is an input. The I/O pin is put in a high-impedance state (weakly pulled high), to be driven by other devices. The logic value on the I/O pin is available in the ”in” and ”in_not” HAL pins. Writes to the ”out” HAL pin have no effect. If this parameter is set to 1, the GPIO is an output; its behavior then depends on the ”is_opendrain” parameter. Only full GPIO pins have this parameter.

11.4.13 StepGen

Stepgens have names like ”hm2_<BoardType>.<BoardNum>.stepgen.<Instance>.”. ”Instance” is a two-digit number that corresponds to the HostMot2 stepgen instance number. There are ”num_stepgens” instances, starting with 00.

Each stepgen allocates 2-6 I/O pins (selected at firmware compile time), but currently only uses two: Step and Direction outputs.[3]

The stepgen representation is modeled on the stepgen software component. Stepgen default is active high step output (high during step time low during step space). To invert a StepGen output pin you invert the corresponding GPIO pin that is being used by StepGen. To find the GPIO pin being used for the StepGen output run dmesg as shown above.

Each stepgen instance has the following pins and parameters:

11.4.13.1 Pins

[3] At present, the firmware supports multi-phase stepper outputs, but the driver doesn’t. Interested volunteers are solicited.

control-type
(Bit, In) Switches between position control mode (0) and velocity control mode (1). Defaults to position control (0).

counts
(s32, Out) Feedback position in counts (number of steps).

enable
(Bit, In) Enables output steps. When false, no steps are generated.

position-cmd
(Float, In) Target position of stepper motion, in user-defined position units.

position-fb
(Float, Out) Feedback position in user-defined position units (counts / position_scale).

velocity-cmd
(Float, In) Target velocity of stepper motion, in user-defined position units per second. This pin is only used when the stepgen is in velocity control mode (control-type=1).

velocity-fb
(Float, Out) Feedback velocity in user-defined position units per second.

11.4.13.2 Parameters

dirhold
(u32, RW) Minimum duration of stable Direction signal after a step ends, in nanoseconds.

dirsetup
(u32, RW) Minimum duration of stable Direction signal before a step begins, in nanoseconds.

maxaccel
(Float, RW) Maximum acceleration, in position units per second per second. If set to 0, the driver will not limit its acceleration.

maxvel
(Float, RW) Maximum speed, in position units per second. If set to 0, the driver will choose the maximum velocity based on the values of steplen and stepspace (at the time that maxvel was set to 0).

position-scale
(Float, RW) Converts from counts to position units. position = counts / position_scale

step_type
(u32, RW) Output format, like the step_type modparam to the software stegen(9) component. 0 = Step/Dir, 1 = Up/Down, 2 = Quadrature. In Quadrature mode (step_type=2), the stepgen outputs one complete Gray cycle (00 -> 01 -> 11 -> 10 -> 00) for each ”step” it takes.

steplen
(u32, RW) Duration of the step signal, in nanoseconds.

stepspace
(u32, RW) Minimum interval between step signals, in nanoseconds.

11.4.13.3 Output Parameters

The Step and Direction pins of each StepGen have two additional parameters. To find which I/O pin belongs to which step and direction output run dmesg as described above.

invert_output
(Bit, RW) This parameter only has an effect if the ”is_output” parameter is true. If this parameter is true, the output value of the GPIO will be the inverse of the value on the ”out” HAL pin.

is_opendrain
(Bit, RW) If this parameter is false, the GPIO behaves as a normal output pin: the I/O pin on the connector is driven to the value specified by the ”out” HAL pin (possibly inverted). If this parameter is true, the GPIO behaves as an open-drain pin. Writing 0 to the ”out” HAL pin drives the I/O pin low, writing 1 to the ”out” HAL pin puts the I/O pin in a high-impedance state. In this high-impedance state the I/O pin floats (weakly pulled high), and other devices can drive the value; the resulting value on the I/O pin is available on the ”in” and ”in_not” pins. Only full GPIO pins and I/O pins used as outputs by active module instances have this parameter.

11.4.14 PWMGen

PWMgens have names like ”hm2_<BoardType>.<BoardNum>.pwmgen.<Instance>.”. ”Instance” is a two-digit number that corresponds to the HostMot2 pwmgen instance number. There are ”num_pwmgens” instances, starting with 00.

In HM2, each pwmgen uses three output I/O pins: Not-Enable, Out0, and Out1. To invert a PWMGen output pin you invert the corresponding GPIO pin that is being used by PWMGen. To find the GPIO pin being used for the PWMGen output run dmesg as shown above.

The function of the Out0 and Out1 I/O pins varies with output-type parameter (see below).

The hm2 pwmgen representation is similar to the software pwmgen component. Each pwmgen instance has the following pins and parameters:

11.4.14.1 Pins

enable
(Bit, In) If true, the pwmgen will set its Not-Enable pin false and output its pulses. If ”enable” is false, pwmgen will set its Not-Enable pin true and not output any signals.

value
(Float, In) The current pwmgen command value, in arbitrary units.

11.4.14.2 Parameters

output-type
(s32, RW) This emulates the output_type load-time argument to the software pwmgen component. This parameter may be changed at runtime, but most of the time you probably want to set it at startup and then leave it alone. Accepted values are 1 (PWM on Out0 and Direction on Out1), 2 (Up on Out0 and Down on Out1), 3 (PDM mode, PDM on Out0 and Dir on Out1), and 4 (Direction on Out0 and PWM on Out1, ”for locked antiphase”).

scale
(Float, RW) Scaling factor to convert ”value” from arbitrary units to duty cycle: dc = value / scale. Duty cycle has an effective range of -1.0 to +1.0 inclusive, anything outside that range gets clipped.

pdm_frequency
(u32, RW) This specifies the PDM frequency, in Hz, of all the pwmgen instances running in PDM mode (mode 3). This is the ”pulse slot frequency”; the frequency at which the pdm generator in the Anything I/O board chooses whether to emit a pulse or a space. Each pulse (and space) in the PDM pulse train has a duration of 1/pdm_frequency seconds. For example, setting the pdm_frequency to 2e6 (2 MHz) and the duty cycle to 50% results in a 1 MHz square wave, identical to a 1 MHz PWM signal with 50% duty cycle. The effective range of this parameter is from about 1525 Hz up to just under 100 MHz. Note that the max frequency is determined by the ClockHigh frequency of the Anything I/O board; the 5i20 and 7i43 both have a 100 MHz clock, resulting in a 100 Mhz max PDM frequency. Other boards may have different clocks, resulting in different max PDM frequencies. If the user attempts to set the frequency too high, it will be clipped to the max supported frequency of the board.

pwm_frequency
(u32, RW) This specifies the PWM frequency, in Hz, of all the pwmgen instances running in the PWM modes (modes 1 and 2). This is the frequency of the variable-duty-cycle wave. Its effective range is from 1 Hz up to 193 KHz. Note that the max frequency is determined by the ClockHigh frequency of the Anything I/O board; the 5i20 and 7i43 both have a 100 MHz clock, resulting in a 193 KHz max PWM frequency. Other boards may have different clocks, resulting in different max PWM frequencies. If the user attempts to set the frequency too high, it will be clipped to the max supported frequency of the board. Frequencies below about 5 Hz are not terribly accurate, but above 5 Hz they’re pretty close.

11.4.14.3 Output Parameters

The output pins of each PWMGen have two additional parameters. To find which I/O pin belongs to which output run dmesg as described above.

invert_output
(Bit, RW) This parameter only has an effect if the ”is_output” parameter is true. If this parameter is true, the output value of the GPIO will be the inverse of the value on the ”out” HAL pin.

is_opendrain
(Bit, RW) If this parameter is false, the GPIO behaves as a normal output pin: the I/O pin on the connector is driven to the value specified by the ”out” HAL pin (possibly inverted). If this parameter is true, the GPIO behaves as an open-drain pin. Writing 0 to the ”out” HAL pin drives the I/O pin low, writing 1 to the ”out” HAL pin puts the I/O pin in a high-impedance state. In this high-impedance state the I/O pin floats (weakly pulled high), and other devices can drive the value; the resulting value on the I/O pin is available on the ”in” and ”in_not” pins. Only full GPIO pins and I/O pins used as outputs by active module instances have this parameter.

11.4.15 Encoder

Encoders have names like ”hm2_<BoardType>.<BoardNum>.encoder.<Instance>.”. ”Instance” is a two-digit number that corresponds to the HostMot2 encoder instance number. There are ”num_encoders” instances, starting with 00.

Each encoder uses three or four input I/O pins, depending on how the firmware was compiled. Three-pin encoders use A, B, and Index (sometimes also known as Z). Four-pin encoders use A, B, Index, and Index-mask.

The hm2 encoder representation is similar to the one described by the Canonical Device Interface (in the HAL General Reference document), and to the software encoder component. Each encoder instance has the following pins and parameters:

11.4.15.1 Pins

count
(s32, Out) Number of encoder counts since the previous reset.

index-enable
(Bit, I/O) When this pin is set to True, the count (and therefore also position) are reset to zero on the next Index (Phase-Z) pulse. At the same time, index-enable is reset to zero to indicate that the pulse has occurred.

position
(Float, Out) Encoder position in position units (count / scale).

rawcounts
(s32, Out) Total number of encoder counts since the start, not adjusted for index or reset.

reset
(Bit, In) When this pin is TRUE, the count and position pins are set to 0. (The value of the velocity pin is not affected by this.) The driver does not reset this pin to FALSE after resetting the count to 0, that is the user's job.

velocity
(Float, Out) Estimated encoder velocity in position units per second.

11.4.15.2 Parameters

counter-mode
(Bit, RW) Set to False (the default) for Quadrature. Set to True for Up/Down or for single input on Phase A. Can be used for a frequency to velocity converter with a single input on Phase A when set to true.

filter
(Bit, RW) If set to True (the default), the quadrature counter needs 15 clocks to register a change on any of the three input lines (any pulse shorter than this is rejected as noise). If set to False, the quadrature counter needs only 3 clocks to register a change. The encoder sample clock runs at 33 MHz on the PCI Anything I/O cards and 50 MHz on the 7i43.

index-invert
(Bit, RW) If set to True, the rising edge of the Index input pin triggers the Index event (if index-enable is True). If set to False, the falling edge triggers.

index-mask
(Bit, RW) If set to True, the Index input pin only has an effect if the Index-Mask input pin is True (or False, depending on the index-mask-invert pin below).

index-mask-invert
(Bit, RW) If set to True, Index-Mask must be False for Index to have an effect. If set to False, the Index-Mask pin must be True.

scale
(Float, RW) Converts from "count" units to "position" units. A quadrature encoder will normally have 4 counts per pulse so a 100 PPR encoder would be 400 counts per revolution. In ".counter-mode" a 100 PPR encoder would have 100 counts per revelution as it only uses the rising edge of A and direction is B.

vel-timeout
(Float, RW) When the encoder is moving slower than one pulse for each time that the driver reads the count from the FPGA (in the hm2_read() function), the velocity is harder to estimate. The driver can wait several iterations for the next pulse to arrive, all the while reporting the upper bound of the encoder velocity, which can be accurately guessed. This parameter specifies how long to wait for the next pulse, before reporting the encoder stopped. This parameter is in seconds.

11.4.16 Examples

Several example configurations are included with LinuxCNC for both stepper and servo applications. The configurations are located in the hm2-servo and hm2-stepper sections of the LinuxCNC Configuration Selector window. You will need the same board installed for the configuration you pick to load. The examples are a good place to start and will save you time. Just pick the proper example from the LinuxCNC Configuration Selector and save a copy to your computer so you can edit it. To see the exact pins and parameters that your configuration gave you, open the Show HAL Configuration window from the Machine menu, or do dmesg as outlined above.

11.5 Motenc

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.

Installing:

```
loadrt hal_motenc
```

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

Up to 4 boards may be used in one system.

11.5.1 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 sets the ID based on a pair of jumpers on the board, 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.

11.5.2 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
1	2	
2	4	Watchdog is enabled
3	8	
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?

11.5.3 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.

11.6 Opto22 PCI

PCI AC5 ADAPTER CARD / HAL DRIVER This page is considered current as of November 2008.

11.6.1 The Adapter Card

This is a card made by Opto22 for adapting the PCI port to solid state relay racks such as their standard or G4 series. It has 2 ports that can control up to 24 points each and has 4 on board LEDs. The ports use 50 pin connectors the same as Mesa boards. Any relay racks/breakout boards thats work with Mesa Cards should work with this card with the understanding any encoder counters, PWM, etc., would have to be done in software. The AC5 does not have any smart logic on board, it is just an adapter.

See the manufacturer's website for more info:

http://www.opto22.com

I would like to thank Opto22 for releasing info in their manual, easing the writing of this driver!

11.6.2 The Driver

This driver is for the PCI AC5 card and will not work with the ISA AC5 card. The HAL driver is a realtime module. It will support 4 cards as is (more cards are possible with a change in the source code). Load the basic driver like so:

```
loadrt opto_ac5
```

This will load the driver which will search for max 4 boards. It will set I/O of each board's 2 ports to a default setting. The default configuration is for 12 inputs then 12 outputs. The pin name numbers correspond to the position on the relay rack. For example the pin names for the default I/O setting of port 0 would be:

opto_ac5.0.port0.in-00 They would be numbered from 00 to 11

opto_ac5.0.port0.out-12 They would be numbered 12 to 23 port 1 would be the same.

11.6.3 PINS

opto_ac5.[BOARDNUMBER].port[PORTNUMBER].in-[PINNUMBER] OUT bit

opto_ac5.[BOARDNUMBER].port[PORTNUMBER].in-[PINNUMBER]-not OUT bit

Connect a HAL bit signal to this pin to read an I/O point from the card. The PINNUMBER represents the position in the relay rack. Eg. PINNUMBER 0 is position 0 in a Opto22 relay rack and would be pin 47 on the 50 pin header connector. The -not pin is inverted so that LOW gives TRUE and HIGH gives FALSE.

opto_ac5.[BOARDNUMBER].port[PORTNUMBER].out-[PINNUMBER] IN bit

Connect a HAL bit signal to this pin to write to an I/O point of the card. The PINNUMBER represents the position in the relay rack.Eg. PINNUMBER 23 is position 23 in a Opto22 relay rack and would be pin 1 on the 50 pin header connector.

opto_ac5.[BOARDNUMBER].led[NUMBER] OUT bit

Turns one of the 4 onboard LEDs on/off. LEDs are numbered 0 to 3.

BOARDNUMBER can be 0-3 PORTNUMBER can be 0 or 1. Port 0 is closest to the card bracket.

11.6.4 PARAMETERS

opto_ac5.[BOARDNUMBER].port[PORTNUMBER].out-[PINNUMBER]-invert W bit

When TRUE, invert the meaning of the corresponding -out pin so that TRUE gives LOW and FALSE gives HIGH.

11.6.5 FUNCTIONS

opto_ac5.0.digital-read Add this to a thread to read all the input points.

opto_ac5.0.digital-write Add this to a thread to write all the output points and LEDs.

For example the pin names for the default I/O setting of port 0 would be:

opto_ac5.0.port0.in-00

They would be numbered from 00 to 11

opto_ac5.0.port0.out-12

They would be numbered 12 to 23 port 1 would be the same.

11.6.6 Configuring I/O Ports

To change the default setting load the driver something like so:

```
loadrt opto_ac5 portconfig0=0xffff portconfig1=0xff0000
```

Of course changing the numbers to match the I/O you would like. Each port can be set up different.

Here's how to figure out the number: The configuration number represents a 32 bit long code to tell the card which I/O points are output vrs input. The lower 24 bits are the I/O points of one port. The 2 highest bits are for 2 of the on board LEDs. A one in any bit position makes the I/O point an output. The two highest bits must be output for the LEDs to work. The driver will automatically set the two highest bits for you, we won't talk about them.

The easiest way to do this is to fire up the calculator under APPLICATIONS/ACCESSORIES. Set it to scientific (click view). Set it BINARY (radio button Bin). Press 1 for every output you want and/or zero

for every input. Remember that HAL pin 00 corresponds to the rightmost bit. 24 numbers represent the 24 I/O points of one port. So for the default setting (12 inputs then 12 outputs) you would push 1 twelve times (thats the outputs) then 0 twelve times (thats the inputs). Notice the first I/O point is the lowest (rightmost) bit. (that bit corresponds to HAL pin 00 .looks backwards) You should have 24 digits on the screen. Now push the Hex radio button. The displayed number (fff000) is the configport number (put a 0x in front of it designating it as a HEX number).

Another example: To set the port for 8 outputs and 16 inputs (the same as a Mesa card). Here is the 24 bits represented in a BINARY number. Bit 1 is the rightmost number.

000000000000000011111111

16 zeros for the 16 inputs and 8 ones for the 8 outputs

Which converts to FF on the calculator so 0xff is the number to use for portconfig0 and/or portconfig1 when loading the driver.

11.6.7 Pin Numbering

HAL pin 00 corresponds to bit 1 (the rightmost) which represents position 0 on an Opto22 relay rack. HAL pin 01 corresponds to bit 2 (one spot to the left of the rightmost) which represents position 1 on an Opto22 relay rack. HAL pin 23 corresponds to bit 24 (the leftmost) which represents position 23 on an Opto22 relay rack.

HAL pin 00 connects to pin 47 on the 50 pin connector of each port. HAL pin 01 connects to pin 45 on the 50 pin connector of each port. HAL pin 23 connects to pin 1 on the 50 pin connector of each port.

Note that Opto22 and Mesa use opposite numbering systems: Opto22 position 23 = connector pin 1, and the position goes down as the connector pin number goes up. Mesa Hostmot2 position 1 = connector pin 1, and the position number goes up as the connector pin number goes up.

11.7 Pico PPMC

Le Pico Systems est une famille de cartes pour contrôler les servos analogiques, les moteurs pas à pas et les servos numériques pilotés en PWM. Les cartes se connectent sur le PC par le port parallèle configuré en mode EPP. Bien que la plupart des utilisateurs ne connectent qu'une seule carte par port parallèle, en théorie toutes les combinaisons de cartes entre 8 et 16 peuvent être utilisées sur un seul port parallèle. Un pilote servant pour tous les types de cartes. La combinaison finale d'entrées/sorties dépends du nombre de cartes installées. Le pilote ne distingue pas entre les cartes, il s'agit simplement d'un numéro de canal d'entrées/sorties (codeur, etc) commençant à 0 sur la première carte.

Installation:

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

Le paramètres port_addr indique au pilote quel port parallèle utiliser. Par défaut, <addr1> est en 0x0378, <addr2> et les suivantes ne sont pas utilisées. Le pilote cherche sur l'espace entier de l'adresse du port parallèle étendu (EPP) indiquée par port_addr, scrutant pour toute carte(s) de la famille PPMC. Il exporte ensuite les pins de HAL de tout ce qu'il a trouvé. Durant le chargement, ou la tentative de chargement, le pilote affiche tous les messages de débogage utiles dans le log du noyau, qui pourra être visualisé avec dmesg.

Un maximum de 3 bus parport peuvent êtres utilisés, et chaque bus peut recevoir un maximum de 8 périphériques.

11.7.1 Pins

Dans ce qui suit, pour les pins, les paramètres et les fonctions, <board> représente l'ID de la carte. Selon nos conventions de nommage, la première carte devrait toujours avoir l'ID zéro. Toutefois, le driver fixera l'ID en se basant sur les switches de la carte, de sorte qu'il peut être différent de zéro même si il n'y a qu'une seule carte.

(All s32 output) ppmc.<port>.encoder.<channel>.count
Position codeur, en nombre de top comptés.

(all s32 output) ppmc.<port>.encoder.<channel>.delta
Différence de top comptés depuis la dernière lecture, en unités brutes de comptage codeur.

(All float output) ppmc.<port>.encoder.<channel>.velocity
Vitesse mise à l'échelle en unités utilisateur par seconde. Sur PPMC et USC ces valeurs sont dérivées du nombre de top codeur par période servo, elle est donc affectée par la granularité du codeur. Sur les cartes UPC avec les micro-logiciels du 21/08/09 et suivants, la vitesse est estimée par timestamping sur le comptage du codeur, ce qui peut être utilisé pour accroitre la finesse de cette sortie vitesse. Cela peut être régulé par un composant PID de HAL pour produire une réponse servo plus stable. Cette fonction doit être validée dans la ligne de commande HAL qui démarre le pilote PPMC, avec une option timestamp=0x00.

(All float output) ppmc.<port>.encoder.<channel>.position
Position codeur, en unités utilisateur.

(All bit bidir) ppmc.<port>.encoder.<channel>.index-enable
Connecte l'index axis.#.index-enable pour home-to-index. C'et un signal de HAL bi-directionnel. Le fixer à TRUE, causera une remise à zéro hardware du codeur sur la prochaine impulsion d'index du codeur. Le pilote détectera cela et remettra le signal sur FALSE.

(UPC bit input) ppmc.<port>.pwm.<channel>.enable
Active un générateur de PWM.

(UPC float input) ppmc.<port>.pwm.<channel>.value
Valeur qui détermine le rapport cyclique de l'onde PWM. La valeur est divisée par pwm.<channel>.scale par exemple, si le résultat est 0.6, le rapport cyclique sera de 60%, et ainsi de suite. Les valeurs de rapport cyclique négatives finiront en valeurs absolues, la pin de direction sera positionnée pour indiquer ce négatif.

(USC bit input) ppmc.<port>.stepgen.<channel>.enable
Active un générateur d'impulsions de pas.

(USC float input) ppmc.<port>.stepgen.<channel>.velocity
Valeur qui détermine la fréquence des pas. La valeur est multipliée par stepgen.<channel>.scale et le résultat est la fréquence, en pas par seconde. Des valeurs négatives résultera une fréquence basée sur une valeur absolue, la pin de direction sera positionnée pour indiquer ce négatif.

(All bit output) ppmc.<port>.in-<channel>
État d'une pin d'entrée numérique, voir l'entrée numérique canonique.

(All bit input) ppmc.<port>.in.<channel>-not
État inversé d'une pin d'entrée numérique, voir l'entrée numérique canonique.

(All bit output) ppmc.<port>.out-<channel>
Valeur à écrire sur une sortie numérique, voir la sortie numérique canonique.

(Option float output) ppmc.<port>.DAC8-<channel>.value
Valeur à écrire sur une sortie analogique, étendue entre 0 et 255. Ce qui envoie 8 bits de sortie sur J8, sur lequel doit être connectée une carte DAC de broche. 0 corresponds à zéro Volts, 255 corresponds à 10 Volts. La polarité de la sortie peut être fixée toujours négative, toujours

positive, ou elle peut être contrôlée par l'état de SSR1 (positive quand on) et SSR2 (négative quand on). Vous devez spécifier extradac = 0x00 sur la ligne de commande HAL qui charge le pilote PPMC pour valider cette fonction sur la première carte USC ou UPC.

(Option bit input) ppmc.<port>.dout-<channel>.out
Valeur à écrire sur une des 8 pins de sorties extra numériques de J8. Vous devez spécifier extradout = 0x00 sur la ligne de commande HAL qui charge le pilote PPMC pour valider cette fonction sur la première carte USC ou UPC. extradac et extradout sont des caractéristiques mutuellement exclusives comme elles utilisent les mêmes lignes de signal à des fins différentes.Ces pins de sortie seront énumérées après les sorties numériques standards de la carte.

11.7.2 Paramètres

(All float) ppmc.<port>.enc.<channel>.scale
Nombre de tops codeur par unité utilisateur (pour les conversions depuis le nombre d'unités).

(UPC float) ppmc.<port>.pwm.<channel-range>.freq
Fréquence porteuse de la PWM, en Hz. S'applique à un groupe de quatre générateurs de PWM consécutifs, indiqués par <channel-range>. Le minimum est de 610Hz, le maximum est de 500KHz.

(UPC float) ppmc.<port>.pwm.<channel>.scale
Échelle pour générateur de PWM. Si scale vaut X, alors le rapport cyclique sera de 100% quand value de la pin vaudra X (ou -X).

(UPC float) ppmc.<port>.pwm.<channel>.max-dc
Rapport cyclique maximum, compris entre 0.0 et 1.0.

(UPC float) ppmc.<port>.pwm.<channel>.min-dc
Rapport cyclique minimum, compris entre 0.0 et 1.0.

(UPC float) ppmc.<port>.pwm.<channel>.duty-cycle
Rapport cyclique actuel (utilisé surtout pour la maintenance)

(UPC bit) ppmc.<port>.pwm.<channel>.bootstrap
Si true, le générateur de PWM générera une courte séquence d'impulsions dans les deux polarités quand l'Arrêt d'Urgence sera activé, pour charger les capacités de bootstrap utilisées par certains pilotes à portes MOSFET.

(USC u32) ppmc.<port>.stepgen.<channel-range>.setup-time
Fixe le temps minimum, entre l'impulsion de changement de direction et l'impulsion de pas, en unités de 100ns. S'applique à un groupe de quatre générateurs de PWM consécutifs, comme indiqué par <channel-range>.

(USC u32) ppmc.<port>.stepgen.<channel-range>.pulse-width
Fixe la largeur des impulsions de pas, en unité de 100ns. S'applique à un groupe de quatre générateurs de PWM consécutifs, comme indiqué par <channel-range>.

(USC u32) ppmc.<port>.stepgen.<channel-range>.pulse-space-min
Fixe le temps minimum entre les impulsions, en unité de 100ns. S'applique à un groupe de quatre générateurs de PWM consécutifs, comme indiqué par <channel-range>. Le ratio maximum est: **`1 / (100ns * (pulse-width + pulse-space-min))`**

(USC float) ppmc.<port>.stepgen.<channel>.scale
Échelle pour générateur d'impulsions de pas. La fréquence des pas est en Hz, c'est la valeur absolue de vitesse * échelle.

(USC float) ppmc.<port>.stepgen.<channel>.max-vel
La valeur maximum de velocity. Les consignes supérieures à max-vel, lui seront clampées. S'applique également aux valeurs négatives. (La valeur absolue est clampée.)

(USC float) ppmc.<port>.stepgen.<channel>.frequency
Fréquence de pas actuelle en Hz (utilisé principalement pour la maintenance)

(Option float) ppmc.<port>.DAC8.<channel>.scale
Fixe l'échelle d'une sortie extra DAC, de sorte qu'une valeur de sortie égale à l'échelle fournisse une amplitude de sortie de 10.0 V. (Le signe de la sortie est fixé par cavaliers et/ou une autre sortie numérique)

(Option bit) ppmc.<port>.out.<channel>-invert
Inverse une sortie numérique, voir la sortie numérique canonique.

(Option bit) ppmc.<port>.dout.<channel>-invert
Inverse une sortie numérique de J8, voir la sortie numérique canonique.

11.7.3 Fonctions

(All funct) ppmc.<port>.read
Lit toutes les entrées (entrées numériques et top de codeurs) sur un port. Ces lectures sont organisées par blocs de registres contigus, pour éviter au maximum de charger le CPU.

(All funct) ppmc.<port>.write
Écrit toutes les sorties (sorties numériques, générateurs de pas et de PWM) sur un port. Ces lectures sont organisées par blocs de registres contigus, pour éviter au maximum de charger le CPU.

11.8 Pluto-P

11.8.1 General Info

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

11.8.1.1 Requirements

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

11.8.1.2 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.

11.8.1.3 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 LinuxCNC, all Pluto-P pins are tristated with weak pull-ups (20k-ohms min, 50k-ohms max). If the watchdog timer is enabled (the default), these pins are also tristated after an interruption of communication between LinuxCNC and the board. The watchdog timer takes approximately 6.5ms to activate. However, software bugs in the pluto_servo firmware or LinuxCNC 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.

11.8.1.4 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**).

11.8.1.5 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.

11.8.1.6 PC interface

- Only a single pluto_servo or pluto_step board is supported.

11.8.1.7 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 LinuxCNC.

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.

11.8.1.8 For more information

Some additional information about it is available from http://www.fpga4fun.com/board_pluto-P.html and from the developer's blog.

11.8.2 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 LinuxCNC servo cycle, or about 8MHz for LinuxCNC'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 100 us on machines tested so far, enabling servo rates above 1kHz.

11.8.2.1 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

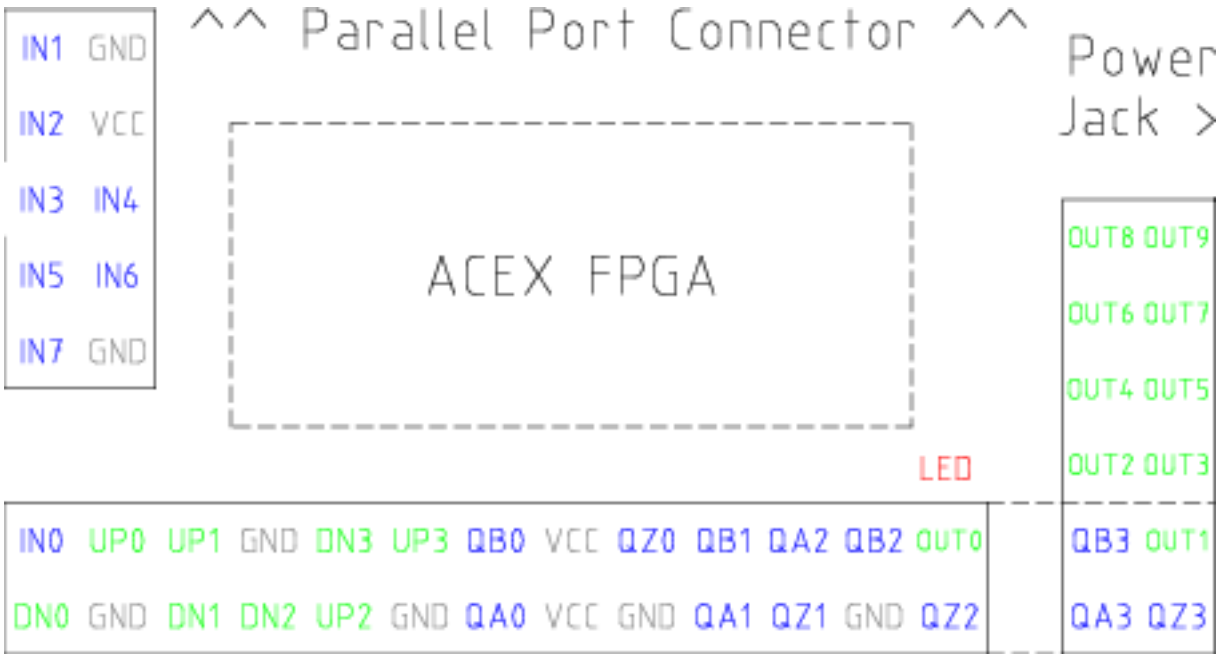


Figure 11.3: Pluto-Servo Pinout

Table 11.1: 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

Table 11.1: (continued)

Primary function	Alternate Function	Behavior if both functions used
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

11.8.2.2 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.

11.8.2.3 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.

11.8.2.4 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 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.

11.8.3 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 communication with the PC

11.8.3.1 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.

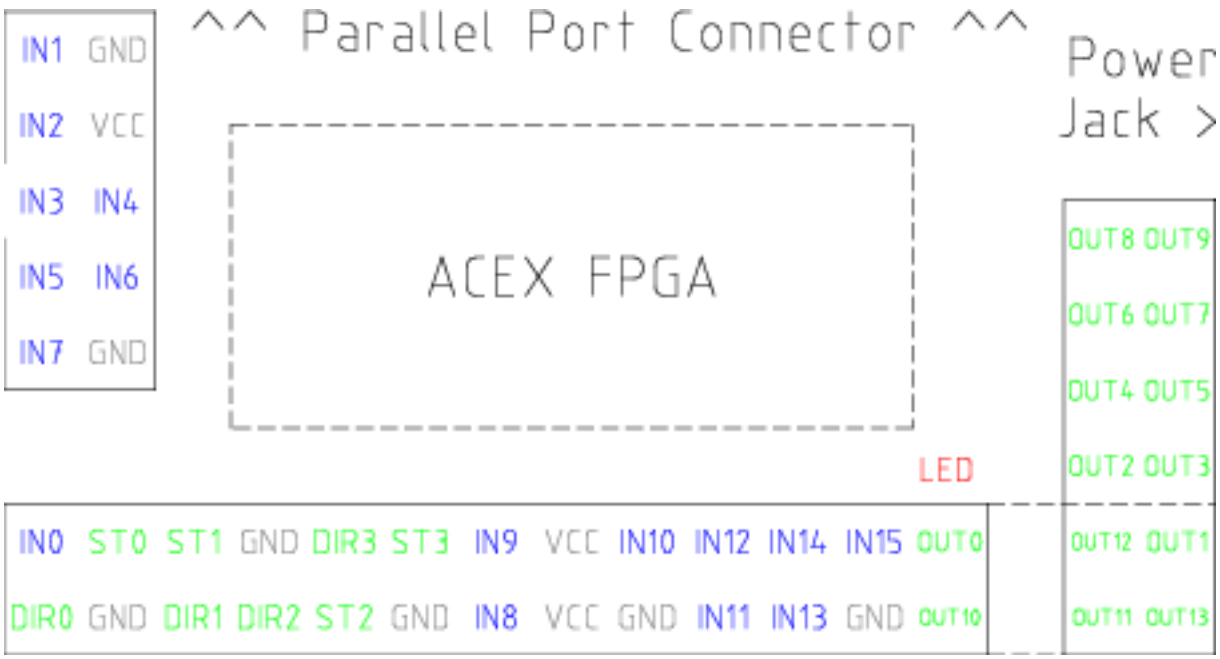


Figure 11.4: Pluto-Step Pinout

11.8.3.2 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.

11.8.3.3 Step Waveform Timings

The firmware and driver enforce step length, space, and direction change times. Timings are rounded up to the next multiple of **1.6µs**, with a maximum of **49.6µs**. 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.

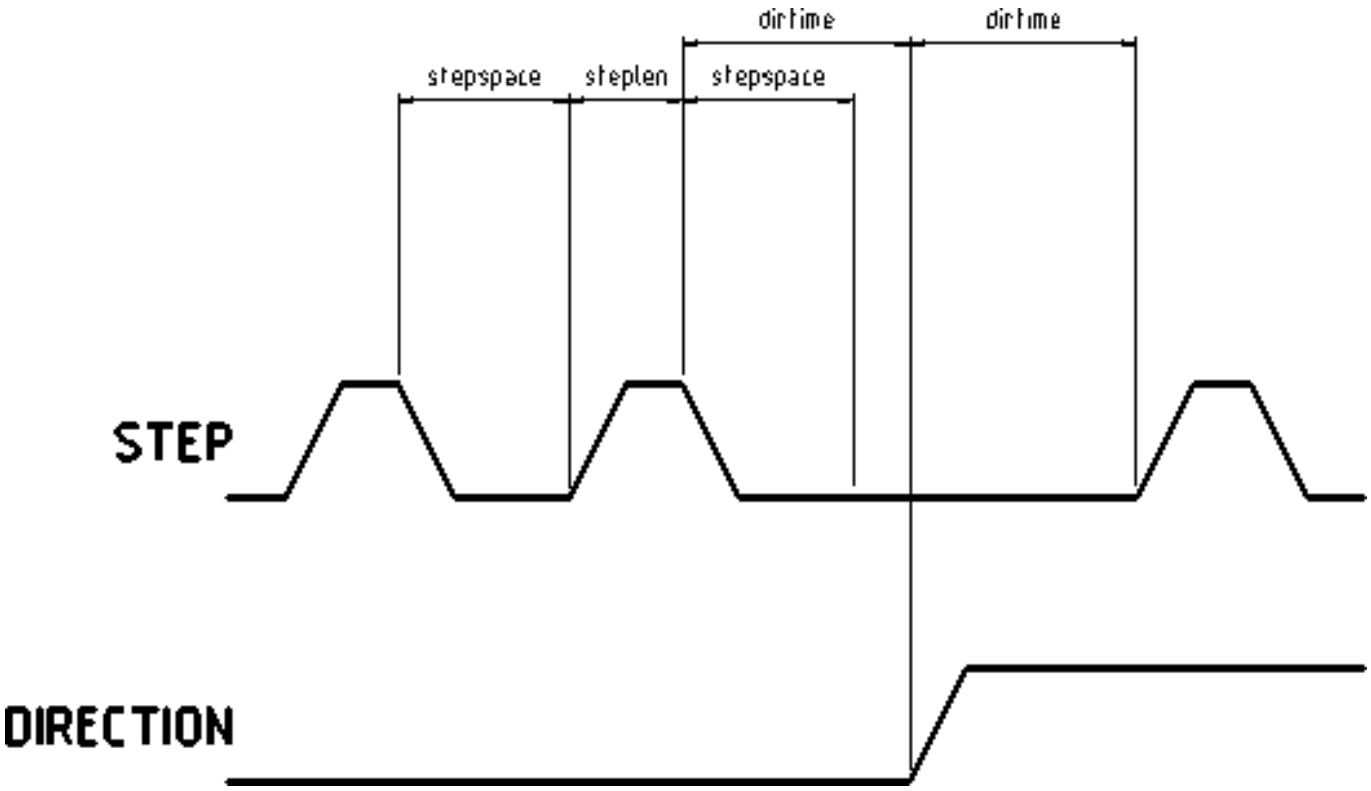


Figure 11.5: Pluto-Step Timings

11.8.3.4 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.

11.9 Servo-To-Go

The Servo-To-Go (STG) is one of the first PC motion control cards supported by LinuxCNC. It is an ISA card and it exists in different flavors (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.

11.9.1 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.

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

11.9.2 Pins

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

For each pin, <channel> is the axis number, and <pinnum> is the logic pin number of the STG if `II00` 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).

The in-<pinnum> 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.

11.9.3 Parameters

- stg.<channel>.position-scale (float) The number of counts / user unit (to convert from counts to units).
- stg.<channel>.dac-offset (float) Sets the offset for the corresponding DAC.
- stg.<channel>.dac-gain (float) Sets the gain of the corresponding DAC.
- stg.<channel>.adc-offset (float) Sets the offset of the corresponding ADC.
- stg.<channel>.adc-gain (float) Sets the gain of the corresponding ADC.
- stg.out-<pinnum>-invert (bit) 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.

11.9.3.1 Functions

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

Chapter 12

Driver Examples

12.1 Deuxième port parallèle sur port PCI

Lors de l'ajout d'un deuxième port parallèle placé dans un slot PCI il est indispensable de connaître son adresse avant de pouvoir l'utiliser avec LinuxCNC. Pour trouver l'adresse de ce port, ouvrez un terminal et tapez:

```
lspci -v
```

Vous devriez voir quelques choses comme ci-dessous parmi la liste du matériel installé en PCI:

```
Communication controller: NetMos Technology PCI 1 port parallel adapter (rev 01)
LSI Logic / Symbios Logic: Unknown device 0010
    medium devsel, IRQ 11
    ports at a800 [size=8]
    ports at ac00 [size=8]
    ports at b000 [size=8]
    ports at b400 [size=8]
    ports at b800 [size=8]
    ports at bc00 [size=16]
```

Dans notre cas, l'adresse était la première, nous avons donc modifié le fichier .hal pour passer de

```
loadrt hal_parport cfg=0x378
```

à

```
loadrt hal_parport cfg="0x378 0xa800 in"
```

Noter les guillemets obligatoires encadrant les adresses.

Nous avons également ajouté:

```
addf parport.1.read base-thread
addf parport.1.write base-thread
```

pour que le second port parallèle soit lu et écrit.

Par défaut les 8 premières broches des ports parallèles sont des sorties. Le drapeau in situé derrière l'adresse d'un port permet de les positionner comme étant 8 entrées sur ce port.

12.2 Contrôle de la broche

LinuxCNC puede controlar hasta 8 husillos. El número se establece en el archivo INI. Todos los ejemplos a continuación se refieren a una configuración de husillo único con pines de control del husillo con nombres como spindle.0... En el caso de una máquina de husillos múltiples, todo lo que cambia es que existen pines adicionales con nombres como spindle.6...

12.2.1 Vitesse broche en 0-10V

Si la vitesse de la broche est contrôlée par un variateur de fréquence avec une consigne vitesse en 0 à 10V et qu'une carte de conversion (DAC) comme la m5i20 est utilisée pour sortir le signal.

Premièrement il faut calculer le facteur d'échelle entre la vitesse broche et la tension de commande. Dans cet exemple, la vitesse maximale de la broche sera de 5000 tr/mn pour une tension de commande de 10V. 10 Volts/5000 tr/mn = 0.002 Volts par tr/mn notre facteur d'échelle sera donc de 0.002.

$$\frac{10\,Volts}{5000\,RPM} = \frac{0.002\,Volts}{1\,RPM}$$

Si la carte DAC ne dispose pas d'une fonction échelle, il est nécessaire d'ajouter un composant scale (echelle) au fichier hal pour calibrer spindle.N.speed-out entre 0 et 10 comme demandé par le variateur de fréquence.

```
loadrt scale count=1
addf scale.0 servo-thread
setp scale.0.gain 0.002
net spindle-speed-scale spindle.N.speed-out => scale.0.in
net spindle-speed-DAC scale.0.out => «le nom de la sortie de votre DAC»
```

12.2.2 Vitesse de broche en PWM

Si la vitesse de la broche peut être contrôlée par un signal de PWM, utiliser le composant pwmgen pour créer le signal:

```
loadrt pwmgen output_type=0
addf pwmgen.update servo-thread
addf pwmgen.make-pulses base-thread
net spindle-speed-cmd spindle.N.speed-out => pwmgen.0.value
net spindle-on spindle.N.on => pwmgen.0.enable
net spindle-pwm pwmgen.0.pwm => parport.0.pin-09-out
# Adapter selon la vitesse maximale de la broche en tr/mn
setp pwmgen.0.scale 1800
```

La réponse du contrôleur de PWM est simple: 0% donne 0tr/mn, 10% donnent 180 tr/mn... 100% donnent 1800 tr/mn. Si un minimum est nécessaire pour faire tourner la broche, suivre l'exemple nist-lathe fourni dans les exemples de configuration pour ajouter un composant d'échelle.

12.2.3 Marche broche

Si un signal de marche broche reliant spindle.N.on à une broche de sortie physique est envisagé. Pour relier ces pins à une broche du port parallèle, ajouter une ligne comme la suivante dans le fichier .hal, il faut bien sûr qu'elle soit câblée à l'interface de contrôle.

```
net spindle-enable spindle.N.on => parport.0.pin-14-out
```

12.2.4 Sens de rotation de la broche

Pour contrôler le sens de rotation de la broche, les pins de HAL spindle.N.forward et spindle.N.reverse étant contrôlées par M3 et M4, peuvent être mise à une valeur positive différente de zéro pour que M3/4 inverse le sens de la broche.

Pour relier ces pins à des broches du port parallèle utiliser, par exemple, les lignes suivantes dans le fichier .hal, bien sûr ces broches doivent être câblées à l'interface de contrôle.

```
net spindle-fwd spindle.N.forward -> parport.0.pin-16-out
net spindle-rev spindle.N.reverse => parport.0.pin-17-out
```

12.2.5 Démarrage en rampe

Si la broche doit démarrer en rampe et que le contrôleur n'a pas cette possibilité, HAL pourra le faire. Il faut premièrement détourner la sortie de spindle.N.speed-out et la faire transiter par un composant limit2 dont l'échelle est ajustée de sorte que la consigne suive une rampe entre spindle.N.speed-out et le périphérique recevant la consigne de vitesse. Ensuite, il faut faire connaitre à LinuxCNC le moment ou la broche a atteint sa vitesse pour que les mouvements puissent commencer.

Reprenant dans l'exemple en 0-10 V, la ligne:

```
net spindle-speed-scale spindle.N.speed-out => scale.0.in
```

sera modifiée, comme indiqué dans l'exemple ci-dessous:

Introduction aux composants de HAL limit2 et near:

Au cas ou vous ne les auriez jamais rencontré auparavant, voici une rapide introduction à ces deux composants de HAL utilisés dans l'exemple suivant.

- Le composant de HAL limit2 est un composant qui reçoit une valeur sur une entrée et fourni une valeur en sortie, limitée entre les seuils min et max et également, limitée pour ne pas dépasser l'amortissement spécifié. En d'autres termes, les fluctuations de la valeur de sortie seront toujours lentes et cette lenteur est ajustable.
- Le composant de HAL near est un composant à deux entrées et une sortie binaire qui indique quand les deux entrées sont approximativement égales.

Voir le manuel de HAL ou les man pages, taper juste man limit2 ou man near.

```
# charge un composant temps réel limit2 et un near avec des noms aisés à suivre
loadrt limit2 names=spindle-ramp
loadrt near names=spindle-at-speed

# ajoute les fonctions au thread
addf spindle-ramp servo-thread
addf spindle-at-speed servo-thread

# fixe le paramètre max pour l'amortissement
# (accélération/décélération de la broche en unités par seconde)
setp spindle-ramp.maxv 60

# détourne la sortie vitesse broche et l'envoie à l'entrée de la rampe
net spindle-cmd <= spindle.N.speed-out => spindle-ramp.in

# la sortie de la rampe est envoyée à l'entrée de l'échelle
net spindle-ramped <= spindle-ramp.out => scale.0.in
```

```
# pour connaitre quand commencer le mouvement on envoie la vitesse de broche
# commandée à une entrée du composant spindle-at-speed (qui est un composant near).
# on envoie également le signal de fin de rampe (vitesse actuelle)
# sur l'autre entrée de spindle-at-speed
net spindle-cmd => spindle-at-speed.in1
net spindle-ramped => spindle-at-speed.in2

# la sortie de spindle-at-speed est envoyée à spindle.N.at-speed
# et quand elle devient TRUE, les mouvements peuvent commencer
net spindle-ready <= spindle-at-speed.out => spindle.N.at-speed
```

12.2.6 Vitesse de broche avec signal de retour

Une information de retour est nécessaire pour que LinuxCNC puisse réaliser des mouvements synchronisés avec la broche comme le filetage ou la vitesse de coupe constante. L'assistant de configuration StepConf peut réaliser les connections lui même si les signaux Canal A codeur broche et Index codeur broche sont choisis parmi les entrées.

Matériel supposé présent:

- Un codeur est monté sur la broche et délivre 100 impulsions par tour sur son canal A.
- Ce canal A est raccordé à la broche 10 du port parallèle.
- L'index de ce codeur est connecté à la broche 11 du port parallèle.

Configuration de base pour ajouter ces composants:

```
loadrt encoder num_chan=1
addf encoder.update-counters base-thread
addf encoder.capture-position servo-thread
setp encoder.0.position-scale 100
setp encoder.0.counter-mode 1
net spindle-position encoder.0.position => spindle.N.revs
net spindle-velocity encoder.0.velocity => spindle.N.speed-in
net spindle-index-enable encoder.0.index-enable <=> spindle.N.index-enable
net spindle-phase-a encoder.0.phase-A
net spindle-phase-b encoder.0.phase-B
net spindle-index encoder.0.phase-Z
net spindle-phase-a <= parport.0.pin-10-in
net spindle-index <= parport.0.pin-11-in
```

12.2.7 Vitesse broche atteinte

Si le moteur de broche possède un retour d'information de vitesse provenant d'un codeur, il est alors possible d'utiliser la variable spindle.N.at-speed pour permettre à LinuxCNC d'attendre que la broche ait atteint sa vitesse de consigne avant d'effectuer tout mouvement. Cette variable passe à TRUE quand la vitesse commandée est atteinte. Comme le retour vitesse est la vitesse de consigne ne sont jamais exactement identiques, il faut utiliser le composant near qui indique quand les deux composantes sont suffisamment proches l'une de l'autre.

Il est nécessaire de connecter la commande de vitesse broche sur near.n.in1 et le signal de retour vitesse du codeur sur near.n.in2. La sortie near.n.out est connectée à spindle.N.at-speed. Le paramètre near.n.scale doit être ajusté pour indiquer dans quelle mesure les deux valeurs sont suffisamment proches pour passer activer la sortie. Selon le matériel utilisé, il pourra être utile d'ajuster l'échelle.

Les éléments suivants sont à ajouter au fichier HAL pour activer Spindle At Speed. Si near est déjà présent dans le fichier HAL, augmenter le numéro de composant et adapter le code suivant en conséquence. S'assurer que le nom du signal est bien le même dans le fichier HAL.

```
# charger un composant near et l'attacher à un thread
loadrt near
addf near.0 servo-thread

# connecter une entrée à la vitesse de broche commandée
net spindle-cmd => near.0.in1

# connecter une entrée à la mesure de vitesse broche du codeur
net spindle-velocity => near.0.in2

# connecter la sortie sur l'entrée spindle-at-speed
net spindle-at-speed spindle.N.at-speed <= near.0.out

# Ajuster les entrées de vitesse de broche pour être dans une fourchette de 1%
setp near.0.scale 1.01
```

12.3 Utilisation d'une manivelle

Cet exemple explique comment relier une manivelle, facile à trouver aujourd'hui sur le marché. Cet exemple utilisera la manivelle MPG3 avec une carte d'interface C22 de chez CNC4PC et un second port parallèle placé sur un slot PCI. Cet exemple fournira trois axes avec chacun trois incréments de pas: 0.1, 0.01, 0.001.

Dans le fichier custom.hal ou dans un fichier jog.hal, ajouter ce qui suit en vérifiant bien que les composants mux4 ou encoder ne soient pas déjà utilisés. Si c'était le cas il faudrait en augmenter le nombre en incrémentant la valeur du count de la commande loadrt. Ajuster également le numéro de référence. Les composants mux4 et encoder sont décrits dans le manuel de HAL et dans les man pages.

Manivelle de jog

```
loadrt encoder num_chan=1
loadrt mux4 count=1
addf encoder.capture-position servo-thread
addf encoder.update-counters base-thread
addf mux4.0 servo-thread

# Mode position
# Chaque cran de manivelle provoque un pas calibré,
# la durée du mouvement total peut dépasser la durée de rotation de la manivelle.
# C'est le mode par défaut.

setp axis.N.jog-vel-mode 0

# Mode vitesse
# L'axe s'arrête quand la manivelle s'arrête, même si la pas de jog est incomplet.
# Décommenter la ligne suivante pour obtenir ce mode de fonctionnement,
# et commenter le mode position.

setp axis.N.jog-vel-mode 1

# Chaque axe est ajusté indépendemment des autres.

# Tailles des pas de jog
```

```
setp mux4.0.in0 0.1
setp mux4.0.in1 0.01
setp mux4.0.in2 0.001

# Sélecteur de taille des pas du jog

net scale1 mux4.0.sel0 <= parport.1.pin-09-in
net scale2 mux4.0.sel1 <= parport.1.pin-10-in

net pend-scale axis.0.jog-scale <= mux4.0.out
net pend-scale axis.1.jog-scale
net pend-scale axis.2.jog-scale

# Signaux du codeur

net mpg-a encoder.0.phase-A <= parport.1.pin-02-in
net mpg-b encoder.0.phase-B <= parport.1.pin-03-in

# Sélecteur d'axe

net mpg-x axis.0.jog-enable <= parport.1.pin-04-in
net mpg-y axis.1.jog-enable <= parport.1.pin-05-in
net mpg-z axis.2.jog-enable <= parport.1.pin-06-in

net pend-counts axis.0.jog-counts <= encoder.0.counts
net pend-counts axis.1.jog-counts
net pend-counts axis.2.jog-counts
```

12.4 Broche avec variateur GS2

12.4.1 Exemple

Cet exemple montre les connexions demandées pour utiliser un variateur de fréquence fourni par la société Automation Direct pour piloter une broche. [1] La direction de la broche et sa vitesse seront contrôlées par LinuxCNC.

L'utilisation du composant GS2 est très simple à régler. Une configuration complète peut être réalisée par l'assistant Stepconf. Bien vérifier que les pins Spindle CW et Spindle PWM sont inutilisées sur la page de réglage du port parallèle.

Placer les lignes suivantes dans le fichier custom.hal pour connecter LinuxCNC au GS2 et avoir le contrôle de celui-ci depuis l'interface utilisateur.

```
# Charger le composant utilisateur pour le variateur variateur
loadusr -Wn spindle-vfd gs2_vfd -n spindle-vfd

# connecter la pin de direction au variateur
net gs2-fwd spindle-vfd.spindle-fwd <= spindle.0.forward

# connecter la pin de Marche/Arrêt au variateur
net gs2-run spindle-vfd.spindle-on <= spindle.0.on

# connecter la pin indiquant que la consigne vitesse est atteinte
net gs2-at-speed spindle.N.at-speed <= spindle-vfd.at-speed

# connecter la commande de vitesse au variateur
net gs2-RPM spindle-vfd.speed-command <= spindle.0.speed-out
```

[1] En Europe on trouve une gamme de produits identiques sous la marque Omron.

Sur le variateur de fréquence lui même, il est nécessaire de fixer quelques paramètres avant de pouvoir communiquer avec lui par Modbus. D'autres seront ajustés selon les caractéristiques demandées par le système. Ces réglages sortent, pour la plupart, du cadre de cet exemple, ils sont tous expliqués dans le manuel du variateur qu'il est impératif de consulter.

- Les switches de communication doivent être positionnés sur RS-232C.
- Les paramètres moteur doivent être ajustés en fonction du moteur.
- P3.00 (Source des commandes de marche) doit être ajusté sur Opérations déterminées par l'interface RS-485, 03 ou 04
- P4.00 (Source des commandes de fréquence) doit être ajusté sur Fréquence déterminée par l'interface RS232C/RS485, 05
- P9.02 (Protocole de communication) doit être ajusté sur Modbus RTU mode, 8 bits de donnée, pas de parity, 2 bits de stop, 03

Un panneau de commande virtuel PyVCP, basé sur cet exemple, est donné dans l'exemple avec un variateur.

Chapter 13

PLC

13.1 La programmation en Ladder

13.1.1 Introduction

La logique Ladder ou langage de programmation Ladder est une méthode pour tracer les schémas en logique électrique. Il s'agit maintenant d'un langage graphique vraiment populaire pour la programmation des automates programmables industriels (API). Il a été à l'origine inventé pour décrire la logique à relais. Son nom est fondé sur la constatation que les programmes dans cette langue ressemblent à une échelle (ladder), avec deux «rails» verticaux et, entre eux, une série «d'échelons». En Allemagne et ailleurs en Europe, le style consiste à placer les rails horizontaux, un en haut de la page et l'autre en bas avec les échelons verticaux dessinés séquentiellement de la gauche vers la droite.

Un programme en logique Ladder, également appelé schéma Ladder, est ressemblant au schéma d'un ensemble de circuits électriques à relais. C'est l'intérêt majeur du schéma Ladder de permettre à une large variété de personnels techniques, ingénieurs, techniciens électriciens, etc de le comprendre et de l'utiliser sans formation complémentaire grâce à cette ressemblance.

La logique Ladder est largement utilisée pour programmer les API, avec lesquels le contrôle séquentiel des processus de fabrication est requis. Le Ladder est utile pour les systèmes de contrôle simples mais critiques, ou pour reprendre d'anciens circuits à relais câblés. Comme les contrôleurs à logique programmable sont devenus plus sophistiqués, ils ont aussi été utilisés avec succès dans des systèmes d'automatisation très complexes.

Le langage Ladder peut être considéré comme un langage basé sur les règles, plutôt que comme un langage procédural. Un «échelon» en Ladder représente une règle. Quand elles sont mises en application avec des éléments électromécaniques, les diverses règles «s'exécutent» toutes simultanément et immédiatement. Quand elle sont mises en application dans la logique d'un automate programmable, les règles sont exécutées séquentiellement par le logiciel, dans une boucle. En exécutant la boucle assez rapidement, typiquement plusieurs fois par seconde, l'effet d'une exécution simultanée et immédiate est obtenu.

13.1.2 Exemple

Les composants les plus communs du Ladder sont les contacts (entrées), ceux-ci sont habituellement NC (normalement clos) ou NO (normalement ouvert) et les bobines (sorties).

- Le contact NO 

- Le contact NC 
- La bobine (sortie) 

Bien sûr, il y a beaucoup plus de composants dans le langage Ladder complet, mais la compréhension de ceux-ci aidera à appréhender le concept global du langage.

L'échelle se compose d'un ou plusieurs échelons. Ces échelons sont tracés horizontalement, avec les composants placés sur eux (entrées, sorties et autres), les composants sont évalués de la gauche vers la droite.

Cet exemple est un simple échelon:

L'entrée B0 sur la gauche et un contact normalement ouvert, il est connecté sur la sortie Q0 sur la droite. Imaginez maintenant qu'une tension soit appliquée à l'extrême gauche, dès que B0 devient vraie (par exemple: l'entrée est activée, ou l'utilisateur a pressé le contact NO), la tension atteint l'extrême droite en traversant la bobine Q0. Avec comme conséquence que la sortie Q0 passe de 0 à 1.

Chapter 14

HAL

14.1 Introduction à 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 de LinuxCNC 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.

14.1.1 HAL est basé sur le système 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 contacts de fin de course, les panneaux de boutons de commande, les manivelles électroniques, 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.

14.1.1.1 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.

14.1.1.2 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.

14.1.1.3 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 interconnexion 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.

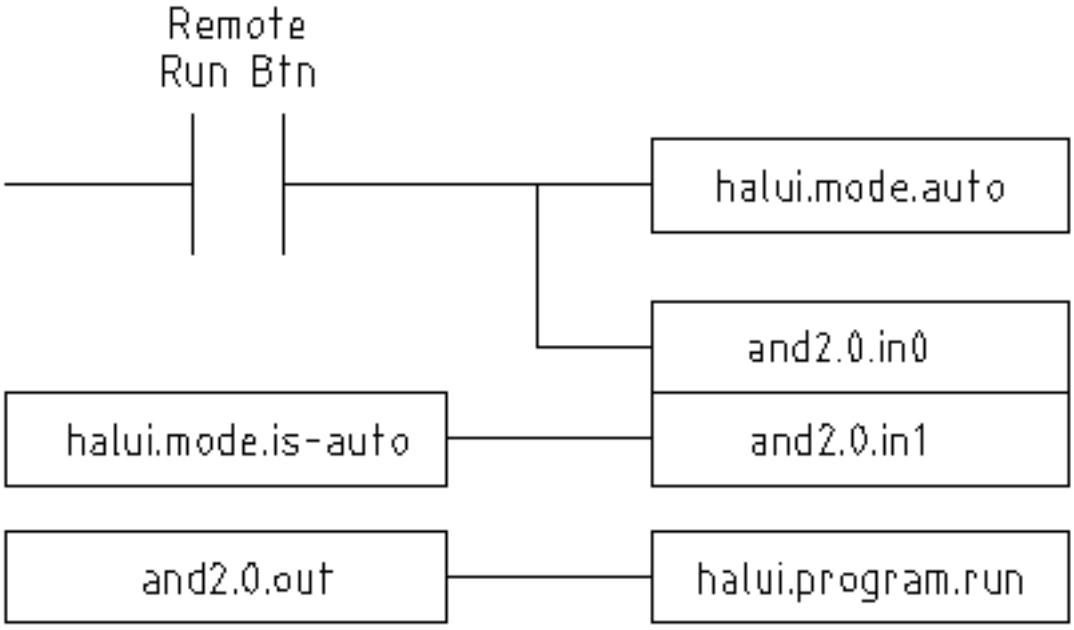
14.1.1.4 Mise au point

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 re-câblées 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.

14.1.1.5 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 LinuxCNC, par exemple pour une Remote Start Example telle que décrite dans la documentation de Halui.



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 indispensable 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 connexion 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.

14.1.2 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 connexion 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. Vous vous demandez peut être quelle relation il y a entre les HAL_pins, les Physical_pins et les éléments extérieurs comme les codeurs ou une carte stg. Nous avons ici, affaire à des interfaces de type translation/conversion de données.

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 +4,294,967,295
- s32 - un entier signé de 32 bits, les valeurs légales vont de -2,147,483,648 à +2,147,483,647

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 très 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 œuvre. 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.

14.1.3 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.

14.1.4 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.

14.1.5 Composants internes

stepgen
Générateur d'impulsions de pas avec boucle de position. Plus de détails sur stepgen.

encoder
Codeur/compteur logiciel. Plus de détails sur le codeur.

pid
Boucle de contrôle Proportionnelle/Intégrale/Dérivée. Plus de détails sur le PID.

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

supply
Une simple alimentation, pour la mise au point

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

14.1.6 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). Plus de détails sur les ports parallèles.

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

14.1.7 Outils-Utilitaires

halcmd
Ligne de commande pour la configuration et les réglages.

halmeter
Un multimètre pour les signaux HAL. Plus de détails pour utiliser halmeter.

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

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

14.1.8 Les réflexions qui ont abouti à la création de 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.

14.1.8.1 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.

14.1.8.2 Erector Sets [2]

C'était une grande série de boites de construction en métal, des tôles perforées, plates ou en cornières, toutes avaient des trous régulièrement 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 âgés 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.

14.1.8.3 Tinkertoys [3]

Le jouet en bois Tinkertoys est plus humain que l'acier froid de l'Erector. Le cœur 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 spatiale 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 ascenseur 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 connectent le multimètre et l'oscilloscope

[2] Le jeu Erector Set est une invention de AC Gilbert (Meccano en France)
[3] Tinkertoy est maintenant registered trademark of the Hasbro company.

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.

Tous les détails dans le tutoriel de HAL.

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.

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.

14.1.9 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.

14.1.10 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.

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

Dans HAL, c'est le code qui évalue. En fait, la version Ladder HAL temps réel de Classic Ladder 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.

Dans LinuxCNC 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 LinuxCNC. 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.

14.2 Commandes et composants de base

14.2.1 Commandes de Hal

Des informations plus détaillées peuvent être trouvées dans la man page en tapant man halcmd dans une console. Pour voir la configuration de HAL ainsi que le statut de ses pins et paramètres utiliser la fenêtre HAL Configuration dans le menu Machine d'AXIS. Pour visualiser le statut des pins, ouvrir l'onglet Watch puis cliquer dans l'arborescence sur les pins qui doivent être visualisées dans la fenêtre watch.

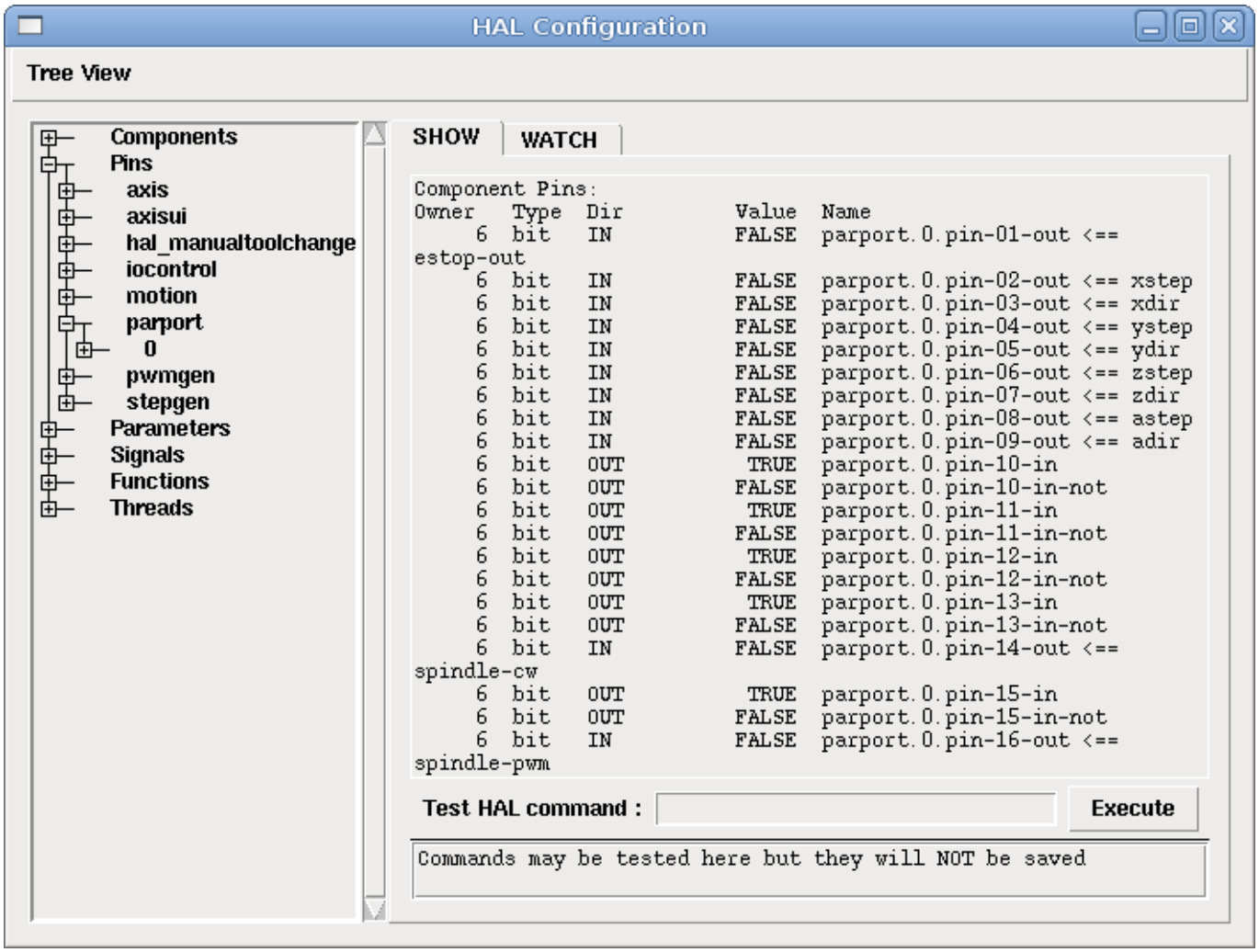


Figure 14.1: Fenêtre de configuration de HAL

14.2.1.1 loadrt

La commande loadrt charge un composant temps réel de HAL. Les composants temps réel doivent être ajoutés au thread temps réel pour être fonctionnels. Il n'est pas possible de charger un composant de l'espace utilisateur dans l'espace temps réel.

Syntaxe et exemple:

```
loadrt <component> <options>

loadrt mux4 count=1
```

14.2.1.2 addf

La commande addf ajoute une fonction à un thread temps réel. Si l'assistant StepConf a été utilisé pour créer la configuration, deux threads ont été créés.

- base-thread (le thread haute vitesse) ce thread prends en main les items nécessitant une réponse très rapide comme la génération d'impulsions, la lecture et l'écriture sur le port parallèle.

- servo-thread (le thread basse vitesse) ce thread prends en main les items n'étant pas influencés par la vitesse comme le contrôleur de mouvement, l'API Classic Ladder et les commandes manuelles.

Syntaxe et exemple:

```
addf <component> <thread>

addf mux4 servo-thread
```

14.2.1.3 loadusr

La commande loadusr charge un composant de HAL de l'espace utilisateur. Les programmes de l'espace utilisateur ont leur propre processus séparé qui optionnellement communique avec les autres composants de HAL via leurs pins et paramètres. Il n'est pas possible de charger un composant temps réel dans l'espace utilisateur.

Les drapeaux peuvent être un ou plusieurs parmi les suivants:

-W
: pour attendre que le composant soit prêt. Le composant est supposé avoir le même nom que le premier argument de la commande.

-Wn <nom>
: pour attendre un composant, qui porte le nom donné sous la forme <nom>.

-w
: pour attendre la fin du programme

-i
: pour ignorer la valeur retournée par le programme (avec -w)

Syntaxe et exemple:

```
loadusr <component> <options>

loadusr halui
loadusr -Wn spindle gs2_vfd -n spindle
```

En anglais ça donne loadusr wait for name spindle component gs2_vfd name spindle. Le -n spindle est une partie du composant gs2_vfd et non de la commande loadusr.

14.2.1.4 net

La commande net crée une connexion entre un signal et une ou plusieurs pins. Si le signal n'existe pas, net le crée. Les flèches de direction <=, => et <=> sont seulement là pour aider à la lecture de la logique, ils ne sont pas utilisés par la commande net. Un espace doit séparer les flèches de direction des noms de pin.

Syntaxe et exemple:

```
net signal-name pin-name <direction optionnelle> (<second pin-name optionnel>)

net home-x axis.0.home-sw-in <= parport.0.pin-11-in
```

Dans l'exemple ci-dessus, home-x est le nom du signal, axis.0.home-sw-in est une pin de direction IN, <= est une flèche de direction optionnelle et parport.0.pin-11-in est une pin de direction OUT. Cela peut paraître déroutant mais les labels in et out, pour une broche de port parallèle, indiquent la direction physique dans laquelle travaille la broche et non comment elle est traitée dans HAL.

Une pin peut être connectée à un signal si elle obéit aux règles suivantes:

- Une pin IN peut toujours être connectée à un signal.
- Une pin IO peut être connectée à moins qu'une pin OUT soit présente sur le signal.
- Une pin OUT peut être connectée seulement si il n'y a pas d'autre pin OUT ou IO sur le signal.

Le même signal-name peut être utilisé dans de multiples commandes net pour connecter des pins additionnelles, tant que les règles précédentes sont observées.

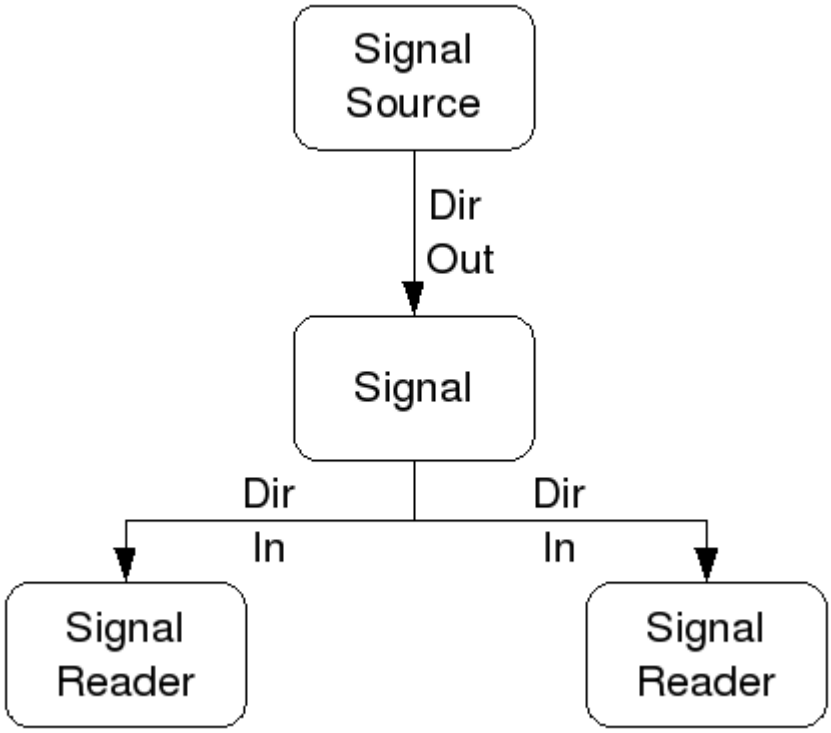


Figure 14.2: Direction du signal

Voici un exemple qui montre le signal xStep avec la source qui est stepgen.0.out et avec deux lecteurs, parport.0.pin-02-out etparport.0.pin-08-out. Simplement la valeur de stepgen.0.out est envoyée au signal xStep et cette valeur est alors envoyée sur parport.0.pin-02-out.

```
#   signal    source            destination
net xStep stepgen.0out => parport.0.pin-02-out
```

Puisque le signal xStep contient la valeur de stepgen.0.out (la source) il est possible de ré-utiliser le même signal pour envoyer la valeur à d'autres lecteurs, utiliser simplement le signal avec les autres lecteurs sur de nouvelles lignes:

```
#   signal       destination2
net xStep => parport.0.pin-08-out
```

Ce qui peut également s'écrire en une seule ligne:

```
#   signal    source            destination1          destination2
net xStep stepgen.0.out => parport.0.pin-02-out parport.0.pin-08-out
```

Pins I/O Les pins appelées I/O pins comme index-enable, ne suivent pas cette règle.

14.2.1.5 setp

La commande setp ajuste la valeur d'une pin ou d'un paramètre. Les valeurs valides dépendront du type de la pin ou du paramètre.

C'est une erreur si les types de donnée ne correspondent pas.

Certains composants ont des paramètres qui doivent être positionnés avant utilisation. Il n'est pas possible d'utiliser setp sur une pin connectée à un signal.

Syntaxe et exemple:

```
setp <pin/parameter-name> <value>

setp parport.0.pin-08-out TRUE
```

14.2.1.6 sets

La commande sets positionne la valeur d'un signal.

Syntaxe et exemple:

```
sets <signal-name> <value>

net mysignal and2.0.in0 pyvcp.my-led
sets mysignal 1
```

C'est une erreur si:

- Le nom de signal n'existe pas
- Le signal à déjà été écrit
- La valeur n'est pas du type correct pour le signal

14.2.1.7 unlinkp

La commande unlinkp déconnecte la pin du signal auquel elle est connectée. Si aucun signal n'a été connecté à la pin avant de lancer cette commande, rien ne se passe.

Syntaxe et exemple:

```
unlinkp <pin-name>

unlinkp parport.0.pin-02-out
```

14.2.1.8 Commandes obsolètes

Les commandes suivantes sont dépréciées et seront retirées dans les futures versions. Toute nouvelle configuration doit utiliser la commande net.

14.2.1.9 linksp

La commande linksp a été remplacée par la commande net.

La commande linksp créait une connexion entre un signal et une pin.

Syntaxe et exemple:

```
linksp <signal-name> <pin-name>

linksp X-step parport.0.pin-02-out
```

14.2.1.10 linkps

La commande linkps a été remplacée par la commande net.

La commande linksp créait une connexion entre une pin et un signal. C'est la même chose que linksp mais les arguments sont inversés.

Syntaxe et exemple:

```
linkps <pin-name> <signal-name>

linkps parport.0.pin-02-out X-Step
```

14.2.1.11 newsig

the command newsig creates a new HAL signal by the name <signame> and the data type of <type>. Type must be bit, s32, u32 or float. Error if <signame> already exists.

Syntaxe et exemple:

```
newsig <signame> <type>

newsig Xstep bit
```

D'autres informations peuvent être trouvées dans le manuel de HAL ou la man page de halrun.

14.2.2 HAL Data

[5]

14.2.2.1 Bit

A bit value is an on or off.

- bit values = true or 1 and false or 0 (True, TRUE, true are all valid)

14.2.2.2 Float

A float is a floating point number. In other words the decimal point can move as needed.

- float values = a 64 bit floating point value, with approximately 53 bits of resolution and over 1000 bits of dynamic range.

For more information on floating point numbers see:

http://fr.wikipedia.org/wiki/Nombre_flottant

14.2.2.3 s32

An s32 number is a whole number that can have a negative or positive value.

- s32 values = integer numbers -2147483648 to 2147483647

[5] NDT la description des données de HAL reste en Anglais, elle sont suffisamment simples pour être comprises.

14.2.2.4 u32

A u32 number is a whole number that is positive only.

- u32 values = integer numbers 0 to 4294967295

14.2.3 Fichiers Hal

Si l'assistant StepConf a été utilisé pour générer la configuration trois fichiers HAL ont dû être créés dans le répertoire de la configuration.

- ma-fraiseuse.hal (si ne nom de la config est "ma-fraiseuse") Ce fichier est chargé en premier, il ne doit pas être modifié sous peine de ne plus pouvoir l'utiliser avec l'assistant StepConf.
- custom.hal Ce fichier est le deuxième à être chargé et il l'est avant l'interface utilisateur graphique (GUI). C'est dans ce fichier que ce trouvent les commandes personnalisées de l'utilisateur devant être chargées avant la GUI.
- custom_postgui.hal Ce fichier est chargé après la GUI. C'est dans ce fichier que se trouvent les commandes personnalisées de l'utilisateur devant être chargées après la GUI. Toutes les commandes relatives aux widgets de pyVCP doivent être placées ici.

14.2.4 Composants de HAL

Deux paramètres sont automatiquement ajoutés à chaque composants HAL quand il est créé. Ces paramètres permettent d'encadrer le temps d'exécution d'un composant.

`.time`

`.tmax`

time est le nombre de cycles du CPU qu'il a fallu pour exécuter la fonction.

tmax est le nombre maximum de cycles du CPU qu'il a fallu pour exécuter la fonction. tmax est un paramètre en lecture/écriture, de sorte que l'utilisateur peut le mettre à 0 pour se débarrasser du premier temps d'initialisation de la fonction.

14.2.5 Composants de logiques combinatoire

Hal contient plusieurs composants logiques temps réel. Les composants logiques suivent une tables de vérité montrant les états logiques des sorties en fonction de l'état des entrées. Typiquement, la manipulation des bits d'entrée détermine l'état électrique des sorties selon la table de vérité des portes.

14.2.5.1 and2

Le composant and2 est une porte and à deux entrées. Sa table de vérité montre la sortie pour chaque combinaison des entrées.

Syntaxe

```
and2 [count=N] or [names=name1[,name2...]]
```

Fonctions

`and2.n`

Pins

```
and2.N.in0 (bit, in)
and2.N.in1 (bit, in)
and2.N.out (bit, out)
```

Table de vérité

in0	**in1**	**out**
False	False	False
True	False	False
False	True	False
True	True	True

14.2.5.2 not

Le composant not est un simple inverseur d'état.

Syntaxe

```
not [count=n] or [names=name1[,name2...]]
```

Fonctions

```
not.all
not.n
```

Pins

```
not.n.in (bit, in)
not.n.out (bit, out)
```

Table de vérité

in	**out**
True	False
False	True

14.2.5.3 or2

Le composant or2 est une porte OR à deux entrées.

Syntaxe

```
or2[count=n] or [names=name1[,name2...]]
```

Functions

```
or2.n
```

Pins

```
or2.n.in0 (bit, in)
or2.n.in1 (bit, in)
or2.n.out (bit, out)
```

Table de vérité

in0	in1	out
True	False	True
True	True	True
False	True	True
False	False	False

14.2.5.4 xor2

Le composant xor2 est une porte XOR à deux entrées (OU exclusif).

Syntaxe

```
xor2[count=n] or [names=name1[,name2...]]
```

Fonctions

`xor2.n`

Pins

```
xor2.n.in0 (bit, in)
xor2.n.in1 (bit, in)
xor2.n.out (bit, out)
```

Table de vérité

in0	in1	out
True	False	True
True	True	False
False	True	True
False	False	False

14.2.5.5 Exemples en logique combinatoire

Un exemple de connexion avec un ”and2”, deux entrées vers une sortie.

```
loadrt and2 count=1
addf and2.0 servo-thread
net my-sigin1 and2.0.in0 <= parport.0.pin-11-in
net my-sigin2 and2.0.in1 <= parport.0.pin-12-in
net both-on parport.0.pin-14-out <= and2.0.out
```

Dans cet exemple un and2 est chargé dans l’espace temps réel, puis ajouté à servo thread. Ensuite la broche d’entrée 11 du port parallèle est connectée à l’entrée in0 de la porte. Puis la broche d’entrée 12 du port est connectée à l’entrée in1 de la porte. Enfin la sortie and2.0.out de la porte est connectée à la broche de sortie 14 du port parallèle. Ainsi en suivant la table de vérité du and2, si les broches 11 et 12 du port sont à 1, alors sa sortie 14 est à 1 aussi.

14.2.6 Composants de conversion

14.2.6.1 Somme pondérée (weighted_sum)

La somme pondérée converti un groupe de bits en un entier. La conversion est la somme des poids des bits présents plus n’importe quel offset. C’est similaire au binaire codé décimal mais avec plus d’options. Le bit hold interrompt le traitement des entrées, de sorte que la valeur sum ne change plus.

La syntaxe suivante est utilisée pour charger le composant weighted_sum.

```
loadrt weighted_sum wsum_sizes=size[,size,...]
```

Crée des groupes de weighted_sum, chacun avec le nombre donné de bits d'entrée (size).

Pour mettre à jour la weighted_sum, le process_wsums doit être attaché à un thread.

```
addf process_wsums servo-thread
```

Ce qui met à jour le composant weighted_sum.

Dans l'exemple suivant, une copie de la fenêtre de configuration de HAL d'Axis, les bits 0 et 2 sont TRUE, ils n'ont pas d'offset. Le poids (weight) du bit 0 est 1, celui du bit 2 est 4, la somme est donc 5.

weighted_sum (somme pondérée)

```
Component Pins:
Owner   Type  Dir         Value  Name
    10  bit   In           TRUE  wsum.0.bit.0.in
    10  s32   I/O             1  wsum.0.bit.0.weight
    10  bit   In          FALSE  wsum.0.bit.1.in
    10  s32   I/O             2  wsum.0.bit.1.weight
    10  bit   In           TRUE  wsum.0.bit.2.in
    10  s32   I/O             4  wsum.0.bit.2.weight
    10  bit   In          FALSE  wsum.0.bit.3.in
    10  s32   I/O             8  wsum.0.bit.3.weight
    10  bit   In          FALSE  wsum.0.hold
    10  s32   I/O             0  wsum.0.offset
    10  s32   Out             5  wsum.0.sum
```

14.3 Le tutoriel de HAL

14.3.1 Introduction

Halrun peut être utilisé pour créer un système complet et fonctionnel. Il s'agit d'un outil de configuration et de mise au point très puissant, en ligne de commande ou en fichier texte. Les exemples suivants illustrent son installation et son fonctionnement.

14.3.2 Halcmd

Halcmd est un outil en ligne de commande pour manipuler HAL. Il existe une man page plus complète pour halcmd, elle sera installée en même temps qu' LinuxCNC depuis ses sources ou depuis un paquet. Si LinuxCNC a été compilé en run-in-place, la man page n'est pas installée, mais elle est accessible, dans le répertoire principal de LinuxCNC, taper:

```
$ man -M docs/man halcmd
```

14.3.2.1 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: loa<TAB>

halcmd: load

halcmd: loadrt

halcmd: loadrt deb<TAB>

halcmd: loadrt debounce
```

14.3.2.2 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 de LinuxCNC 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 de LinuxCNC, vous pouvez étudier linuxcnc/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.

14.3.3 Tutoriel simple

14.3.3.1 Charger un composant temps réel

Pour ce tutoriel, nous allons supposer que vous avez installé avec succès le CD-Live ou que vous avez compilé correctement l'arborescence linuxcnc/src. Si nécessaire, invoquez le script rip-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:

```
$cd linuxcnc

$linuxcnc halrun

$halcmd:
```

Avec l'OS temps réel et RTAPI chargés, vous pouvez passer 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.

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

```
halcmd: loadrt siggen
```

14.3.3.2 Examiner HAL

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

```
halcmd: show comp

    Loaded HAL Components:
    ID      Type  Name          PID   State
    3       RT    siggen              ready
    2       User  halcmd2177    2177  ready
```

Puisque halcmd lui même est un composant HAL, il sera toujours présent dans la liste. 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. 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
        3  float   IN             1  siggen.0.amplitude
        3  float  OUT             0  siggen.0.cosine
        3  float   IN             1  siggen.0.frequency
        3  float   IN             0  siggen.0.offset
        3  float  OUT             0  siggen.0.sawtooth
        3  float  OUT             0  siggen.0.sine
        3  float  OUT             0  siggen.0.square
        3  float  OUT             0  siggen.0.triangle
```

Cette commande affiche toutes les pins présentes dans HAL. Un système complexe peut avoir plusieurs dizaines ou centaines de pins. Mais pour le moment il y a seulement huit pins. Toutes ces huit 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, certaines pins ont une valeur de zéro.

L'étape suivante consiste à examiner les paramètres:

```
halcmd: show param

    Parameters:
    Owner   Type  Dir        Value   Name
        3    s32   RO            0   siggen.0.update.time
        3    s32   RW            0   siggen.0.update.tmax
```

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 paramètres marqués -W sont en écriture possible, pour ceux qui ne sont jamais modifiés par le composant lui-même, mais qui sont modifiables 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 débogage, ils ne sont pas couverts par cette documentation. Les paramètres thread.time et thread.tmax sont associés avec le thread créé quand le composant a été chargé. Quand la réécriture de HAL sera terminée, le thread ne sera plus créé à ce stade, de sorte que ces paramètres ne seront plus visibles.

Il n'y a pas de thread créé ici, mais il y a quand même les paramètres siggen.0.update.time et siggen.0.update

Les paramètres de thread sont ceux du composant 02, le module siggen. C'est incorrect, ils devraient être ceux du module hal_lib, parce que le thread lui même n'est plus la propriété de siggen, et si siggen est retiré, les paramètres devraient rester.

Et bien finalement, fixer les paramètres de thread aura pris plus de temps que je ne pensais. Donc, je les ai éliminés pour l'instant. Quand la réécriture de HAL sera terminée, je les remettrai.

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
    00003  f801b000  fae820b8  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 zéro. [6]

14.3.3.3 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 1 ms (1000 µs ou 1000000 ns):

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

Voyons si il fonctionne:

```
halcmd: show thread

    Realtime Threads:
         Period  FP     Name               (     Time, Max-Time )
         999855  YES           test-thread (        0,        0 )
```

Il fonctionne. La période n'est pas exactement de 1000000 ns 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 HAL. Mais cette fois-ci, nous avons utilisé la commande addf (add function) pour changer quelque chose dans 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 )
         999855  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 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:

[6] Les champs CodeAddr et Arg ont été utilisés pendant le développement et devraient probablement disparaître.

```
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
         3  float IN              1  siggen.0.amplitude
         3  float OUT    -0.1640929  siggen.0.cosine
         3  float IN              1  siggen.0.frequency
         3  float IN              0  siggen.0.offset
         3  float OUT    -0.4475303  siggen.0.sawtooth
         3  float OUT     0.9864449  siggen.0.sine
         3  float OUT            -1  siggen.0.square
         3  float OUT    -0.1049393  siggen.0.triangle
```

Regardons encore une fois:

```
halcmd: show pin

    Component Pins:
    Owner   Type  Dir         Value  Name
         3  float IN              1  siggen.0.amplitude
         3  float OUT     0.0507619  siggen.0.cosine
         3  float IN              1  siggen.0.frequency
         3  float IN              0  siggen.0.offset
         3  float OUT     -0.516165  siggen.0.sawtooth
         3  float OUT     0.9987108  siggen.0.sine
         3  float OUT            -1  siggen.0.square
         3  float OUT    0.03232994  siggen.0.triangle
```

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.

14.3.3.4 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
```

Voyons encore une fois les paramètres et les pins:

```
halcmd: show param

    Parameters:
    Owner   Type  Dir         Value  Name
         3  s32   RO           1754  siggen.0.update.time
         3  s32   RW          16997  siggen.0.update.tmax

halcmd: show pin

    Component Pins:
    Owner   Type  Dir         Value  Name
         3  float IN              5  siggen.0.amplitude
         3  float OUT     0.8515425  siggen.0.cosine
```

```
         3  float IN              1  siggen.0.frequency
         3  float IN              0  siggen.0.offset
         3  float OUT      2.772382  siggen.0.sawtooth
         3  float OUT     -4.926954  siggen.0.sine
         3  float OUT             5  siggen.0.square
         3  float OUT      0.544764  siggen.0.triangle
```

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.

14.3.3.5 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 réellement 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
    # pin aliases
    # signals
    # nets
    # parameter values
    setp siggen.0.update.tmax 14687
    # 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
```

14.3.3.6 Quitter halrun

Pour quitter halrun, ne pas fermez simplement la fenêtre de terminal sans avoir arrêté la session de HAL, pour l'arrêter correctement tapez:

```
halcmd: exit

~/linuxcnc$
```

14.3.3.7 Restaurer la configuration de HAL

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

```
~/linuxcnc$ halrun -I -f saved.hal
```

Noter 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

halcmd: exit

~/linuxcnc$
```

14.3.3.8 Suppression de la mémoire de HAL

Si un arrêt inattendu d'une session de HAL survient, il sera peut être nécessaire de décharger HAL de la mémoire avant de pouvoir lancer une autre session. Pour cela, taper la commande suivante dans une fenêtre de terminal:

```
~/linuxcnc$ halrun -U
```

14.3.4 Visualiser HAL avec halmeter

Il est possible de 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 HAL, est halmeter. C'est un programme très simple qui s'utilise comme un multimètre. Il permet d'observer les pins, signaux ou paramètres en affichant la valeur courante de ces items. Il est très simple à utiliser. Dans une console taper halmeter. halmeter est une application pour environnement graphique. Deux fenêtres vont apparaître, la fenêtre de sélection est la plus grande. Elle comprend trois onglets. Un onglet liste toutes les pins actuellement définies dans HAL. Le suivant, liste tous les signaux et le dernier onglet, liste tous les paramètres. Cliquer sur un onglet, puis cliquer sur un des items pour le sélectionner. La petite fenêtre affichera le nom et la valeur de l'item sélectionné. L'affichage est mis à jour environ 10 fois par seconde. Pour libérer de la place sur l'écran, la fenêtre de sélection peut être fermée avec le bouton Fermer. Sur la petite fenêtre, cachée sous la grande à l'ouverture, le bouton Sélectionner, ré-ouvre la fenêtre de sélection et le bouton Quitter arrête le programme et ferme les fenêtres.

Il est possible d'ouvrir et de faire fonctionner simultanément plusieurs halmeter, ce qui permet de visualiser plusieurs items en même temps. Pour ouvrir un halmeter en libérant la console, taper halmeter & pour le lancer en tâche de fond. Il est possible de lancer halmeter en lui faisant afficher immédiatement un item, pour cela, ajouter les arguments sur la ligne de commande pin|sig|par[am] nom. Il affichera le signal, la pin, ou le paramètre nom dès qu'il démarrera. Si l'item indiqué n'existe pas, il démarrera normalement. Finalement, si un item est spécifié pour l'affichage, il est possible d'ajouter -s devant pin|sig|param pour indiquer à halmeter d'utiliser une fenêtre encore plus réduite. Le nom de l'item sera affiché dans la barre de titre au lieu de sous la valeur et il n'y aura pas de bouton. Utile pour afficher beaucoup de halmeter dans un petit espace de l'écran.

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:

```
~/linuxcnc$ halrun

halcmd: loadrt siggen

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

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

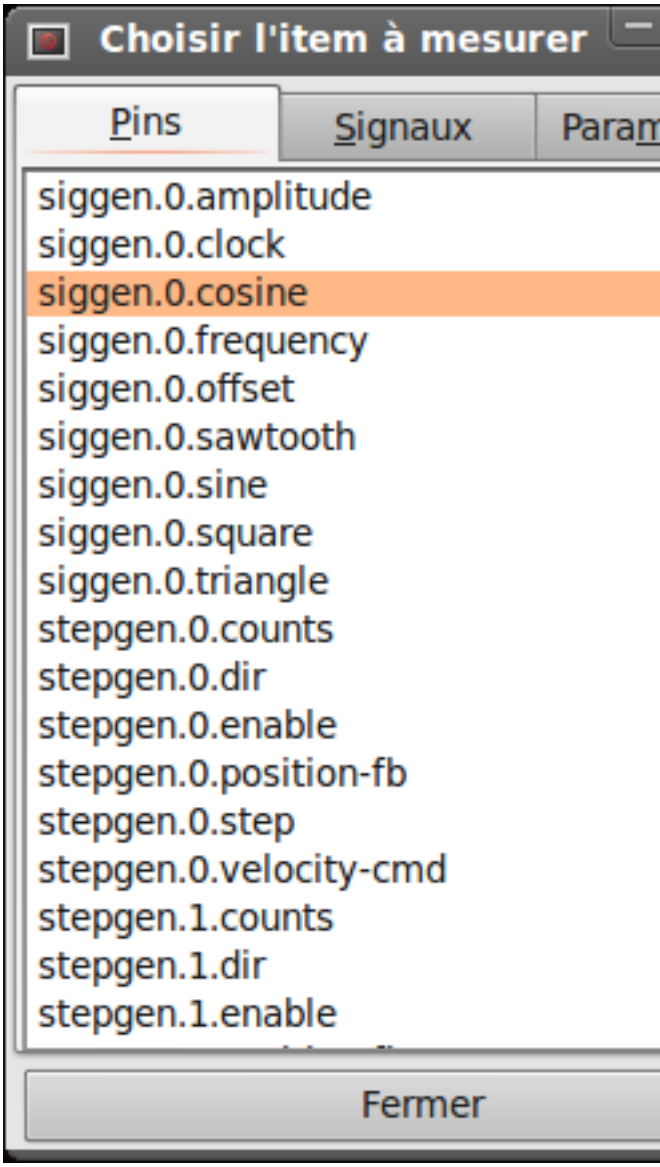
```
halcmd: start

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

14.3.4.1 Lancement de halmeter

À ce stade, nous avons chargé le composant siggen, il 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, demandant de sélectionner l'item à observer.



Fenêtre de sélection de halmeterFenêtre de sélection

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.cosine en premier, il suffit de cliquer sur elle puis sur le bouton Fermer. Le dialogue de sélection se ferme et la mesure s'affiche dans une fenêtre semblable à la figure ci-dessous.

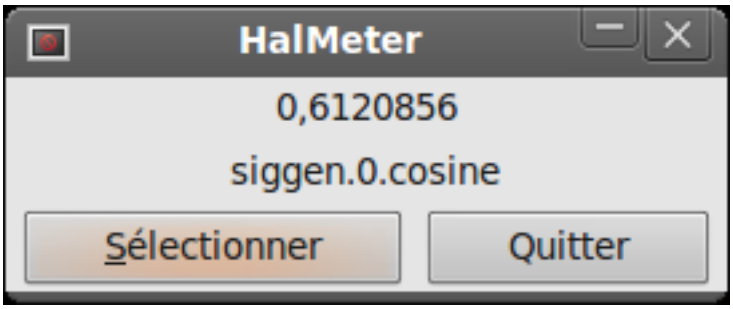


Figure 14.3: Affichage de la valeur

Pour modifier ce qui est affiché sur halmeter pressez le bouton Sélectionner qui vous ramènera à la fenêtre de sélection précédente.

Vous devriez voir la valeur évoluer puisque siggen génère une onde cosinusoïdale. halmeter rafraîchi son affichage environ 5 fois par seconde.

Pour éteindre halmeter, cliquer sur le bouton Quitter.

Pour visualiser plusieurs pins, signaux ou paramètres en même temps, il est possible d'ouvrir plusieurs halmeter. La fenêtre de halmeter est intentionnellement petite justement pour permettre d'en ouvrir un grand nombre sur le même écran.

14.3.5 Tutoriel plus complexe avec stepgen

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. L'exemple suivant 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

~/linuxcnc$ halrun
```

14.3.5.1 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:[7]

Dans cet exemple nous utiliserons le type de contrôle velocity du composant stepgen.

```
halrun: loadrt stepgen step_type=0,0 ctrl_type=v,v

halcmd: loadrt siggen

halcmd: loadrt threads name1=fast fp1=0 period1=50000 name2=slow period2=1000000
```

[7] 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.

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 50 µs 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 à HAL. Cette fois, nous aurons beaucoup plus de pins et de paramètres que précédemment:

```
halcmd: show pin

    Component Pins:
    Owner   Type  Dir         Value  Name
         4  float IN              1  siggen.0.amplitude
         4  float OUT             0  siggen.0.cosine
         4  float IN              1  siggen.0.frequency
         4  float IN              0  siggen.0.offset
         4  float OUT             0  siggen.0.sawtooth
         4  float OUT             0  siggen.0.sine
         4  float OUT             0  siggen.0.square
         4  float OUT             0  siggen.0.triangle
         3  s32   OUT             0  stepgen.0.counts
         3  bit   OUT         FALSE  stepgen.0.dir
         3  bit   IN          FALSE  stepgen.0.enable
         3  float OUT             0  stepgen.0.position-fb
         3  bit   OUT         FALSE  stepgen.0.step
         3  float IN              0  stepgen.0.velocity-cmd
         3  s32   OUT             0  stepgen.1.counts
         3  bit   OUT         FALSE  stepgen.1.dir
         3  bit   IN          FALSE  stepgen.1.enable
         3  float OUT             0  stepgen.1.position-fb
         3  bit   OUT         FALSE  stepgen.1.step
         3  float IN              0  stepgen.1.velocity-cmd


halcmd: show param

    Parameters:
    Owner   Type  Dir         Value  Name
         4  s32   RO              0  siggen.0.update.time
         4  s32   RW              0  siggen.0.update.tmax
         3  u32   RW     0x00000001  stepgen.0.dirhold
         3  u32   RW     0x00000001  stepgen.0.dirsetup
         3  float RO              0  stepgen.0.frequency
         3  float RW              0  stepgen.0.maxaccel
         3  float RW              0  stepgen.0.maxvel
         3  float RW              1  stepgen.0.position-scale
         3  s32   RO              0  stepgen.0.rawcounts
         3  u32   RW     0x00000001  stepgen.0.steplen
         3  u32   RW     0x00000001  stepgen.0.stepspace
         3  u32   RW     0x00000001  stepgen.1.dirhold
         3  u32   RW     0x00000001  stepgen.1.dirsetup
         3  float RO              0  stepgen.1.frequency
         3  float RW              0  stepgen.1.maxaccel
         3  float RW              0  stepgen.1.maxvel
         3  float RW              1  stepgen.1.position-scale
         3  s32   RO              0  stepgen.1.rawcounts
         3  u32   RW     0x00000001  stepgen.1.steplen
         3  u32   RW     0x00000001  stepgen.1.stepspace
         3  s32   RO              0  stepgen.capture-position.time
         3  s32   RW              0  stepgen.capture-position.tmax
         3  s32   RO              0  stepgen.make-pulses.time
         3  s32   RW              0  stepgen.make-pulses.tmax
```

```
        3  s32   RO              0  stepgen.update-freq.time
        3  s32   RW              0  stepgen.update-freq.tmax
```

14.3.5.2 Connexion des pins avec les signaux

Nous avons donc deux générateurs d'impulsions de pas 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 de pas. 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 HAL, les fils sont appelés signaux. Nous devons en créer deux. Nous pouvons les appeler comme on veut, dans cet exemple il y aura X-vel et Y-vel. Le signal X-vel partira de la sortie cosinus du générateur de signaux et arrivera sur l'entrée velocity du premier générateur d'impulsions de pas. 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

    Signals:
    Type          Value  Name     (linked to)
    float             0  X-vel <== siggen.0.cosine
```

Quand un signal est connecté à une ou plusieurs pins, la commande show liste les pins immédiatement suivies par le nom du signal. Les flèches donnent la direction du flux de données, dans ce cas, le flux va de la pin siggen.0.cosine vers le signal X-vel. Maintenant, connectons X-vel à l'entrée velocity du générateur d'impulsions de pas:

```
halcmd: net X-vel => stepgen.0.velocity-cmd
```

Nous pouvons aussi connecter l'axe Y au signal Y-vel. Il doit partir de la sortie sinus du générateur de signaux pour arriver sur l'entrée du second générateur d'impulsions de pas. 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 => stepgen.1.velocity-cmd
```

Pour voir l'effet de la commande net, regardons encore les signaux et les pins:

```
halcmd: show sig

    Signals:
    Type          Value  Name     (linked to)
    float             0  X-vel <== siggen.0.cosine
                                ==> stepgen.0.velocity-cmd
    float             0  Y-vel <== siggen.0.sine
                                ==> stepgen.1.velocity-cmd
```

La commande show sig montre clairement comment les flux de données circulent dans HAL. Par exemple, le signal X-vel provient de la pin siggen.0.cosine et va vers la pin stepgen.0.velocity-cmd.

14.3.5.3 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 délicates à appréhender. 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
     00004  f9992000  fc731278  YES      0   siggen.0.update
     00003  f998b20f  fc7310b8  YES      0   stepgen.capture-position
     00003  f998b000  fc7310b8  NO       0   stepgen.make-pulses
     00003  f998b307  fc7310b8  YES      0   stepgen.update-freq
```

En 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, dent de scie, triangle, carrée. Pour générer un signal régulier, il doit fonctionner à des intervalles très précis.

Les trois autres fonctions sont relatives au générateur d'impulsions de pas:

La première, stepgen.capture-position, est utilisée pour un retour de position. Elle capture la valeur d'un compteur interne comptant 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 stepgen.make-pulses. Chaque fois que make-pulses démarre, elle décide qu'il est temps de faire un pas, si oui elle fixe les sorties en conséquence. Pour des pas plus doux, elle doit fonctionner le plus souvent possible. Parce qu'elle a besoin de fonctionner de manière rapide, make-pulses est hautement optimisée et n'effectue que quelques calculs. Contrairement aux autres, elle n'a pas besoin de virgule flottante pour ses calculs.

La dernière fonction, stepgen.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 allons faire tourner siggen.0.update à une vitesse modérée pour le calcul des valeurs sinus et cosinus. Immédiatement après avoir lancé siggen.0.update, nous lançons stepgen.0.update_freq pour charger les nouvelles valeurs dans le générateur d'impulsions. Finalement nous lancerons stepgen.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 stepgen.capture_position.

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

```
halcmd: show thread

    Realtime Threads:
         Period  FP     Name               (     Time, Max-Time )
         996980  YES                  slow (        0,        0 )
          49849  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 stepgen.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 stepgen.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 stepgen.update-freq slow

halcmd: addf stepgen.make-pulses fast
```

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

```
halcmd: show thread

    Realtime Threads:
         Period  FP     Name               (     Time, Max-Time )
         996980  YES                  slow (        0,        0 )
                      1 siggen.0.update
                      2 stepgen.update-freq
          49849  NO                   fast (        0,        0 )
                      1 stepgen.make-pulses
```

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

14.3.5.4 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 10 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 stepgen.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 stepgen.0.position-scale 10000

halcmd: setp stepgen.1.position-scale 10000

halcmd: setp stepgen.0.enable 1

halcmd: setp stepgen.1.enable 1
```

Cela signifie que, avec la pin stepgen.0.velocity-cmd 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.

14.3.5.5 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.

14.3.6 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 offre un, appelé halscope. Il permet de capturer la valeur des pins, des signaux et des paramètres en fonction du temps.

14.3.6.1 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 Fonction temps réel non liée visible sur la figure ci-dessous:

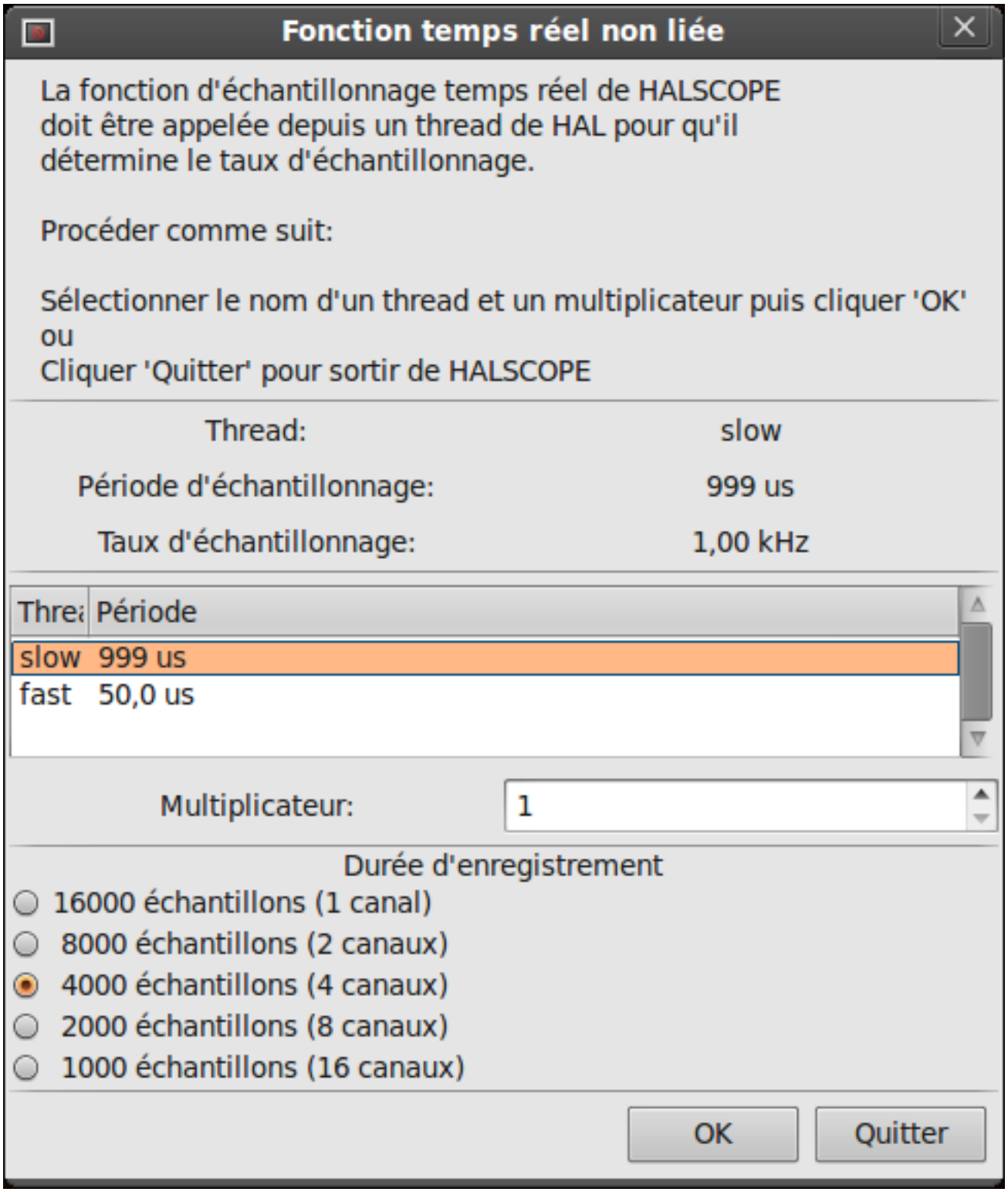


Figure 14.4: Dialogue Fonction temps réel non liée

C'est dans ce dialogue que vous définissez le taux d'échantillonnage de l'oscilloscope. Pour le moment nous voulons un échantillon par milliseconde, alors cliquez sur le thread slow et laissez le multiplicateur à 1. Nous allons aussi passer la longueur d'enregistrement à 4000 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 initiale du scope s'ouvre, comme ci-dessous.

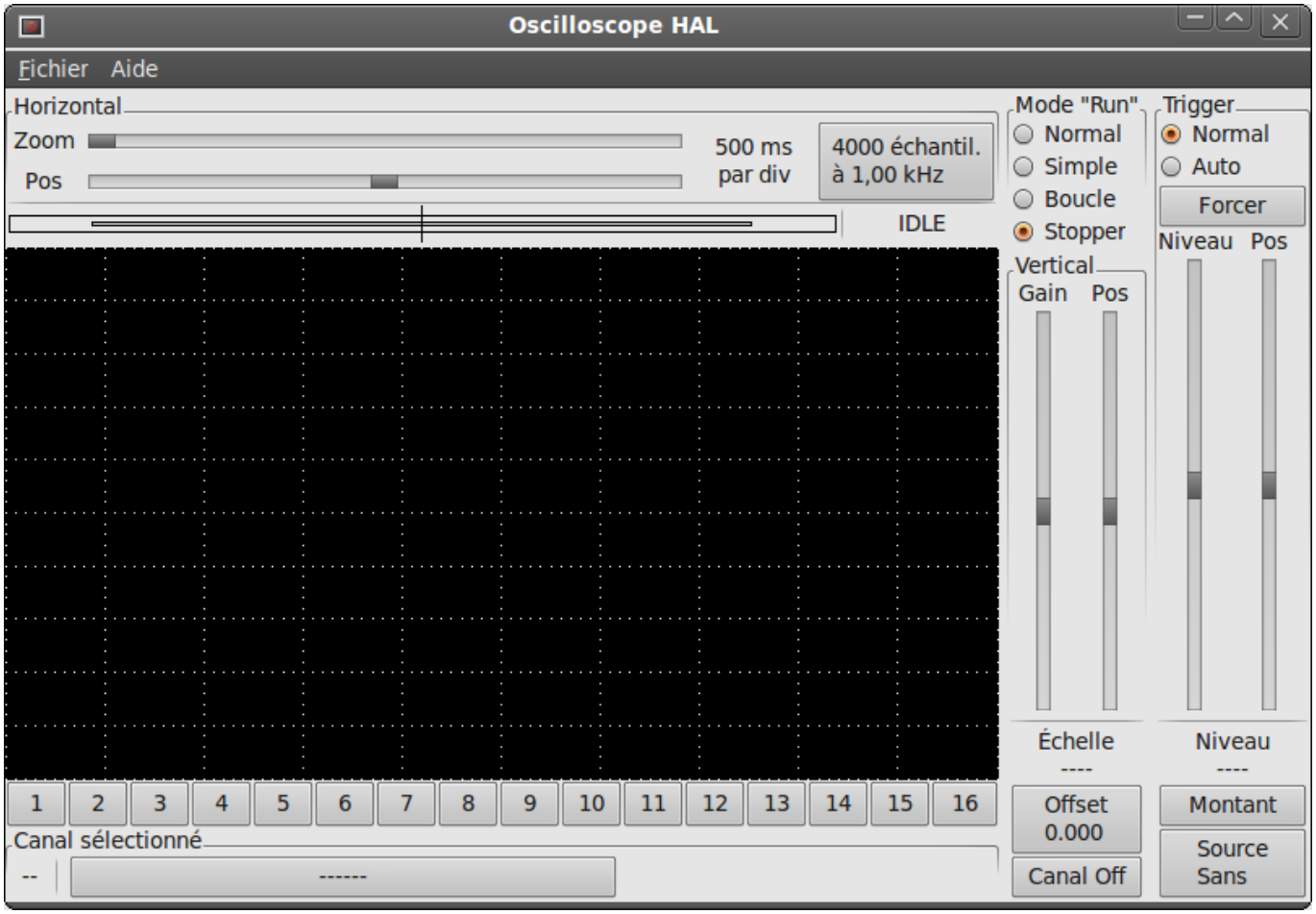


Figure 14.5: Fenêtre initiale du scope

14.3.6.2 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 de canaux utilisables à un moment donné dépend de la longueur d'enregistrement, plus il y a de canaux, plus les enregistrements seront 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 bouton 1 et vous verrez apparaître le dialogue de sélection des sources dans lequel vous devrez choisir _la source qui devra s'afficher sur le canal 1, comme sur la figure ci-dessous. Ce dialogue est très similaire à celui utilisé par halmeter.

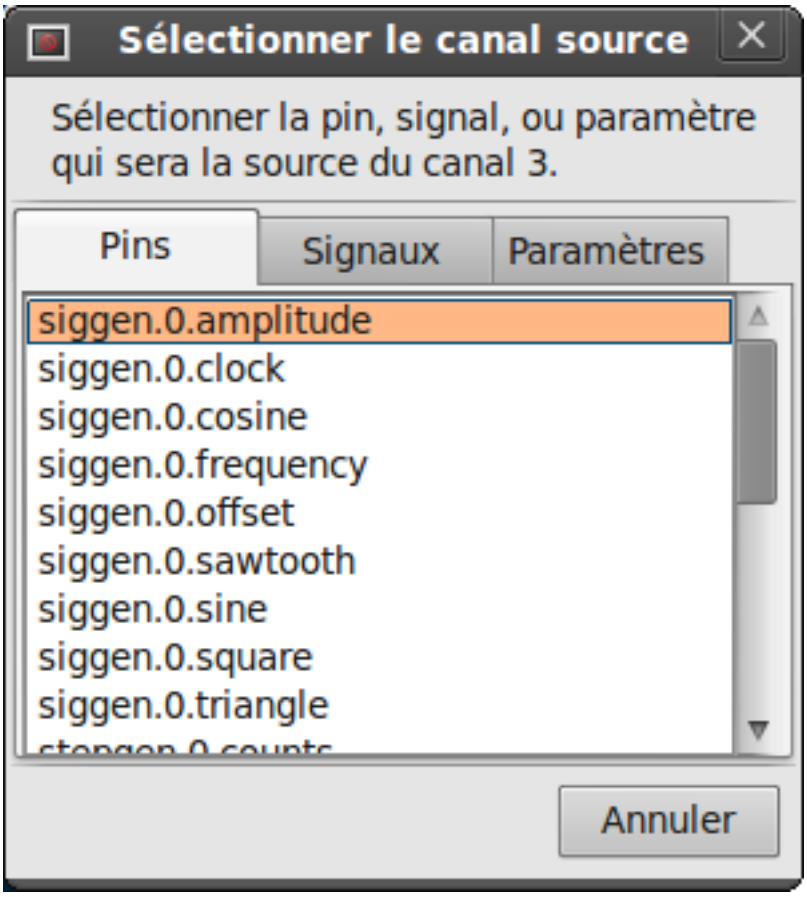


Figure 14.6: Dialogue de sélection des sources

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 HAL, dans notre exemple nous avons seulement les deux signaux X-vel et Y-vel, comme ci-dessous.

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.

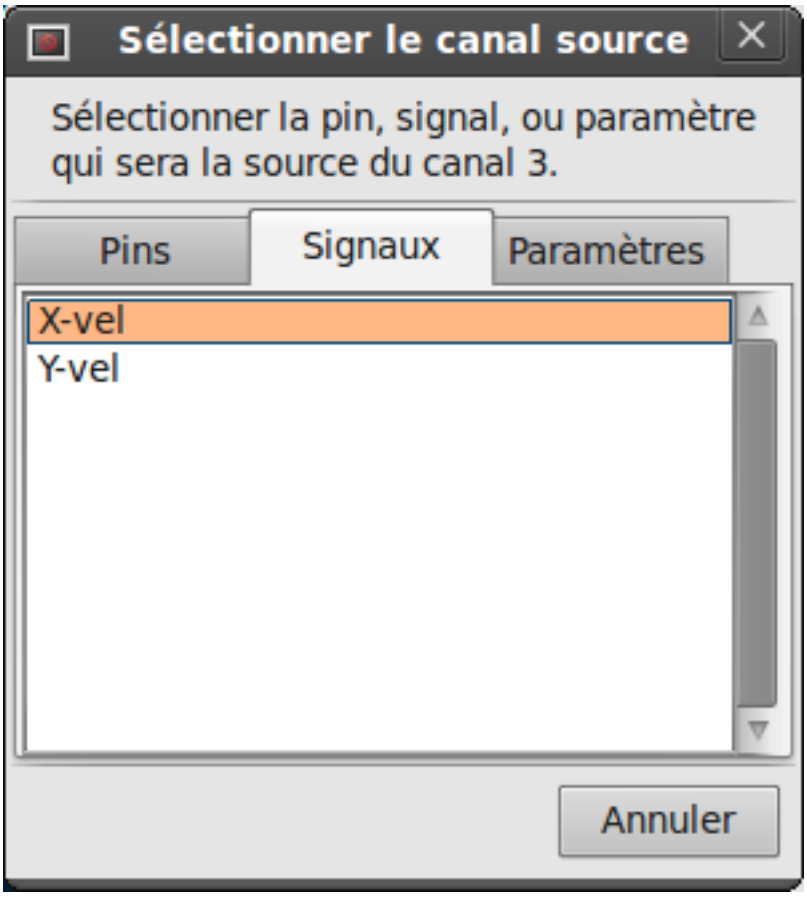


Figure 14.7: Sélection du signal

Le bouton du canal 1 est pressé, 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.

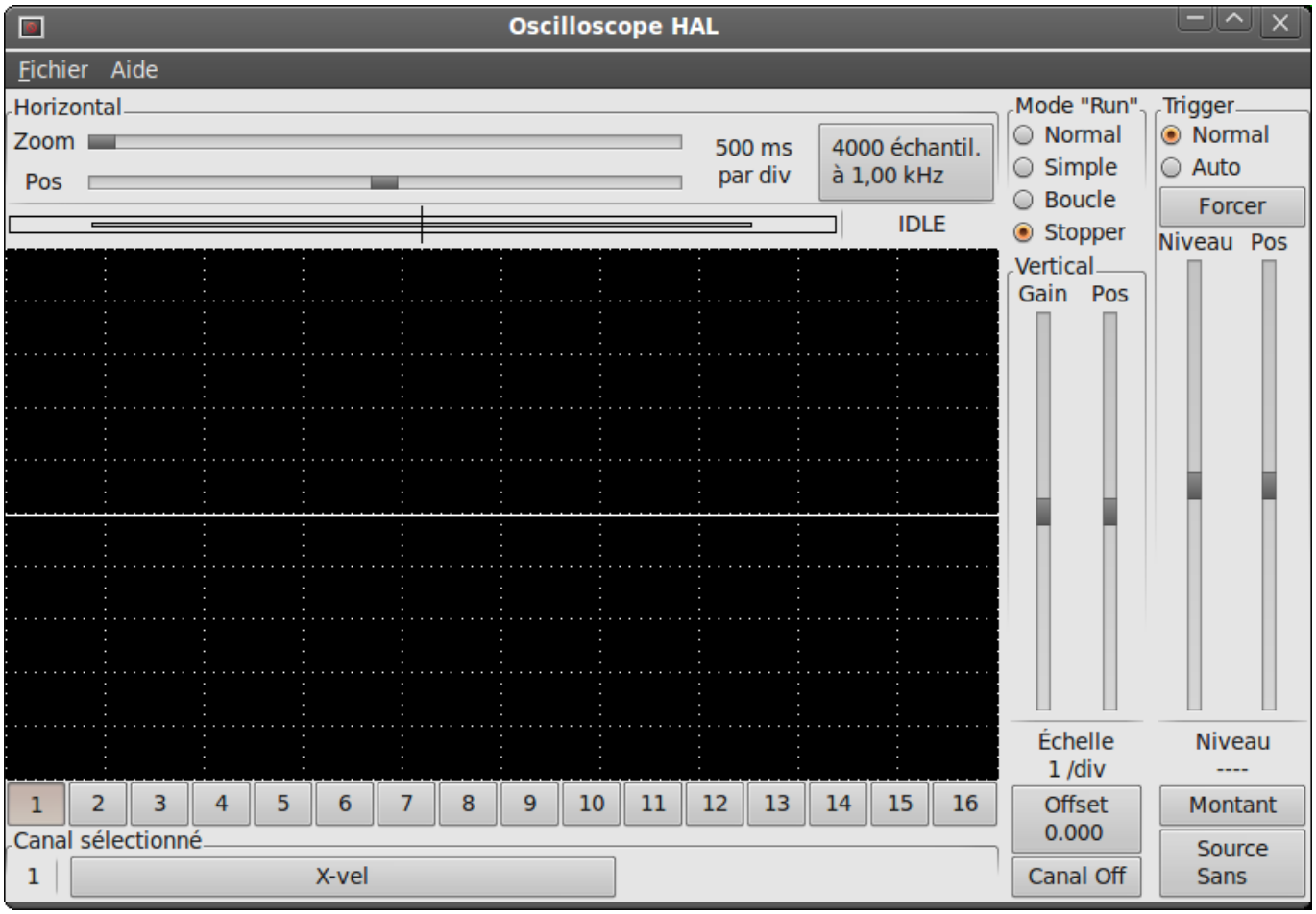


Figure 14.8: halscope

Les différents contrôles comme la position verticale et l'amplitude sont toujours relatifs au canal 1. Pour ajouter un signal sur le canal 2, cliquer sur le bouton 2. Dans la fenêtre de dialogue, cliquer sur l'onglet Signaux, puis cliquer 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.

14.3.6.3 Capturer notre première forme d'onde

Maintenant que nous avons plusieurs sondes branchées sur HAL, nous pouvons capturer quelques formes d'ondes. Pour démarrer le scope, cochez la case Normal du groupe Mode "Run" (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 (Trigger). Puisque nous n'en avons pas encore configuré, il attendra toujours. Pour déclencher manuellement, cliquez sur le bouton Forcer 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 ci-dessous.

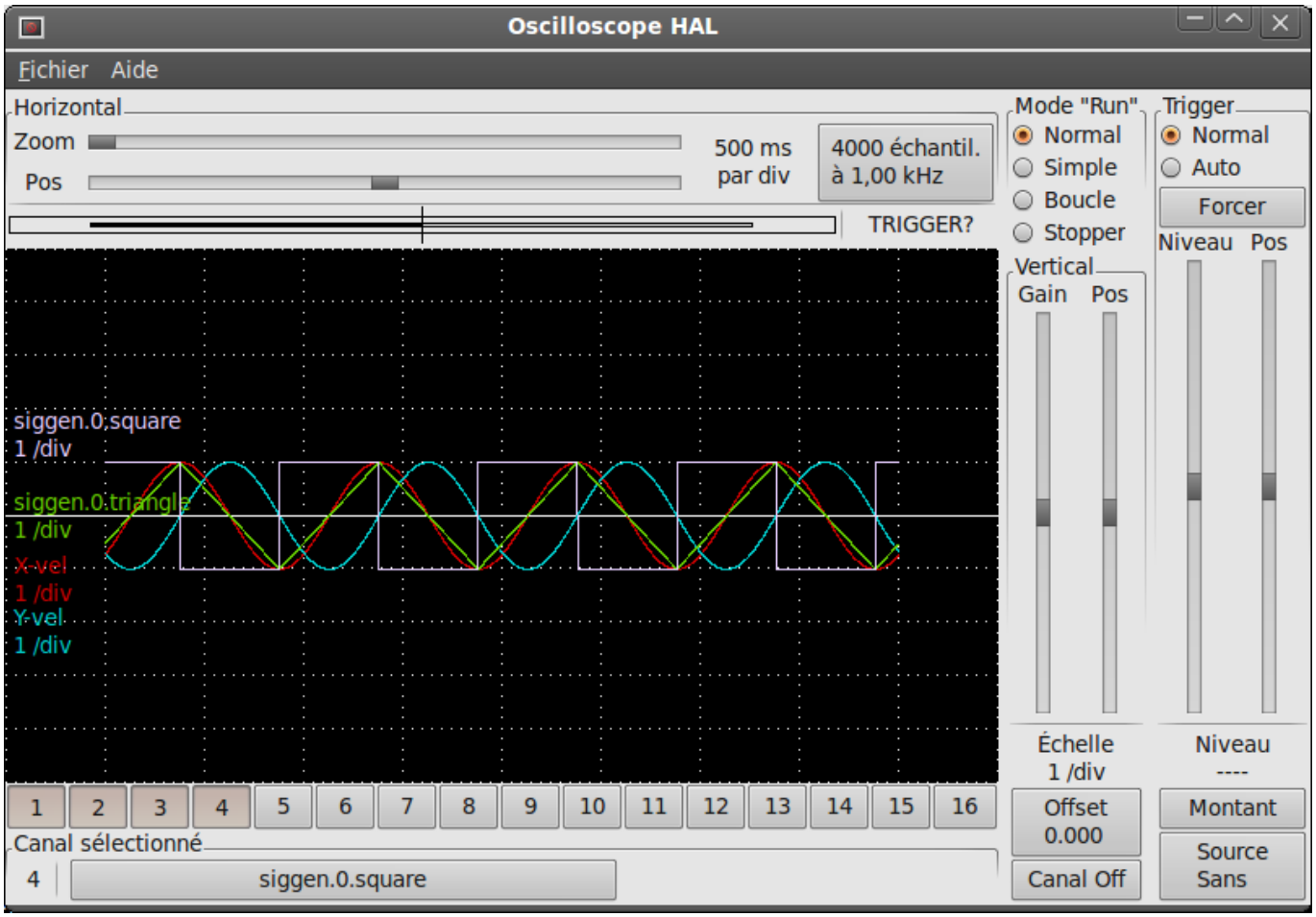


Figure 14.9: Capture d’ondes

14.3.6.4 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 les curseurs du groupe Vertical situé à droite de l’écran. Ces deux curseurs agissent sur le canal actuellement sélectionné. En ajustant le Gain, notez qu’il couvre une large échelle (contrairement aux oscilloscopes réels), celle-ci permet d’afficher des signaux très petits (pico unités) à très grands (Tera - unités). Le curseur Pos déplace la trace affichée de haut en bas sur toute la hauteur de l’écran. Pour de plus grands ajustements le bouton Offset peut être utilisé.

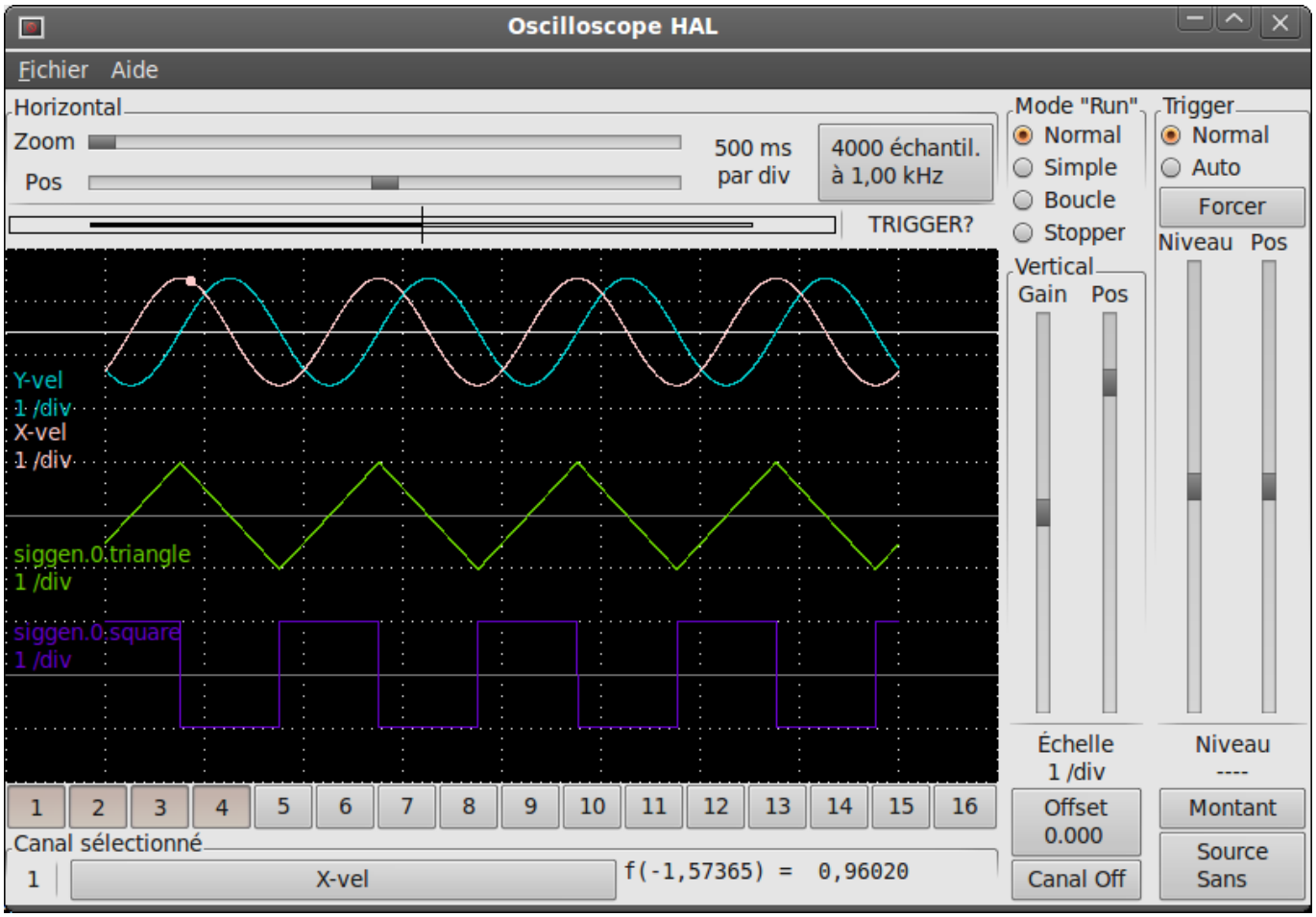


Figure 14.10: Ajustement vertical

Le grand bouton Canal sélectionné en bas, indique que le canal 1 est actuellement le canal sélectionné et qu'il correspond au signal X-vel. Essayez de cliquer sur les autres canaux pour mettre leurs traces en évidence et pouvoir les déplacer avec le curseur Pos.

14.3.6.5 Déclenchement (Triggering)

L'utilisation du bouton Forcer n'est parfois pas satisfaisante pour déclencher le scope. Pour régler un déclenchement réel, cliquer 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, voir la figure ci-dessous. Sélectionner la sonde à utiliser pour déclencher en cliquant dessus. Pour notre exemple nous utilisons 3 canaux, essayons l'onde triangle. Quand le dialogue ce referme, après le choix, le bouton affiche Source Canal n où n est le numéro du canal venant d'être choisi comme déclencheur.

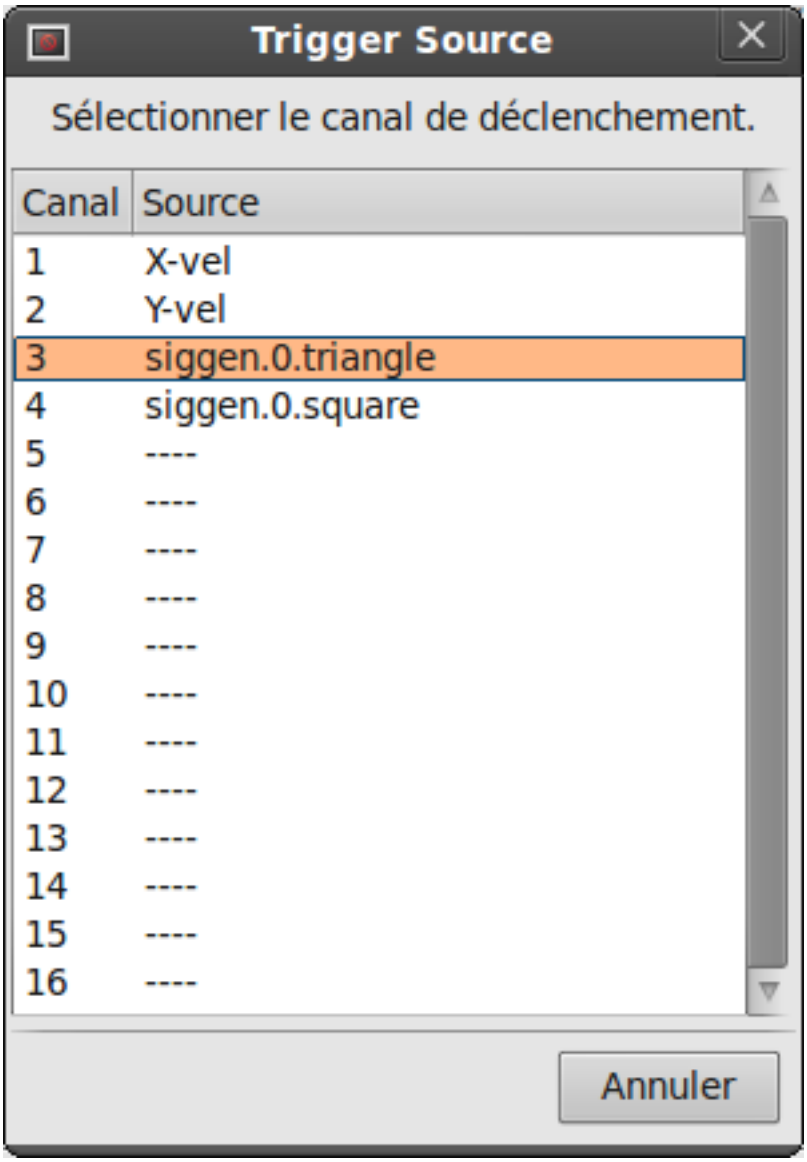


Figure 14.11: Dialogue des sources de déclenchement

Après avoir défini la source de déclenchement, il est possible d'ajuster le niveau de déclenchement avec les curseurs du groupe Trigger le long du bord droit. Le niveau peut être modifié à partir du haut vers le bas de l'écran, il 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 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 déclenchement. Le point de déclenchement est visible comme une petite 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 Montant situé juste sous l'affichage du niveau de déclenchement, il deviendra alors descendant. 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 Mode "Run".

Maintenant que nous avons réglé la position verticale et le déclenchement, l'écran doit ressembler à la figure ci-dessous.

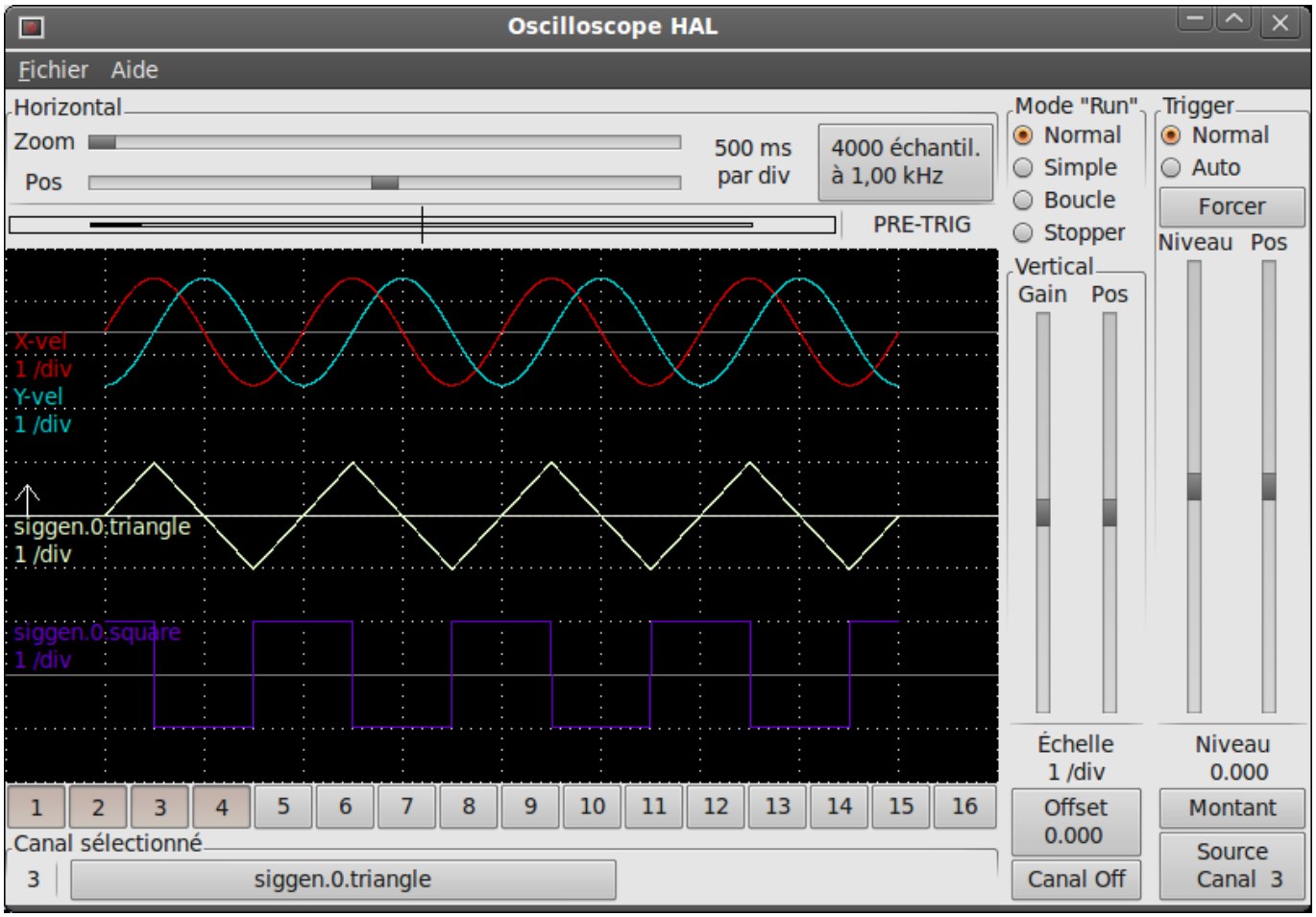


Figure 14.12: Formes d'ondes avec déclenchement

14.3.6.6 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 horizontale, Pos du groupe Horizontal, 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 50 us de long, l'échantillonnage à 1kHz n'est pas assez rapide. Pour changer le taux d'échantillonnage, cliquer sur le bouton qui affiche le nombre d'échantillons pour avoir le dialogue Sélectionner un taux d'échantillonnage, figure ci-dessous. Pour notre exemple, nous cliquerons sur le thread fast, qui fournira un échantillonnage à environ 20kHz. Maintenant au lieu d'afficher environ 4 secondes de données, un enregistrement sera de 4000 échantillons à 20kHz, soit environ 0.20 seconde.

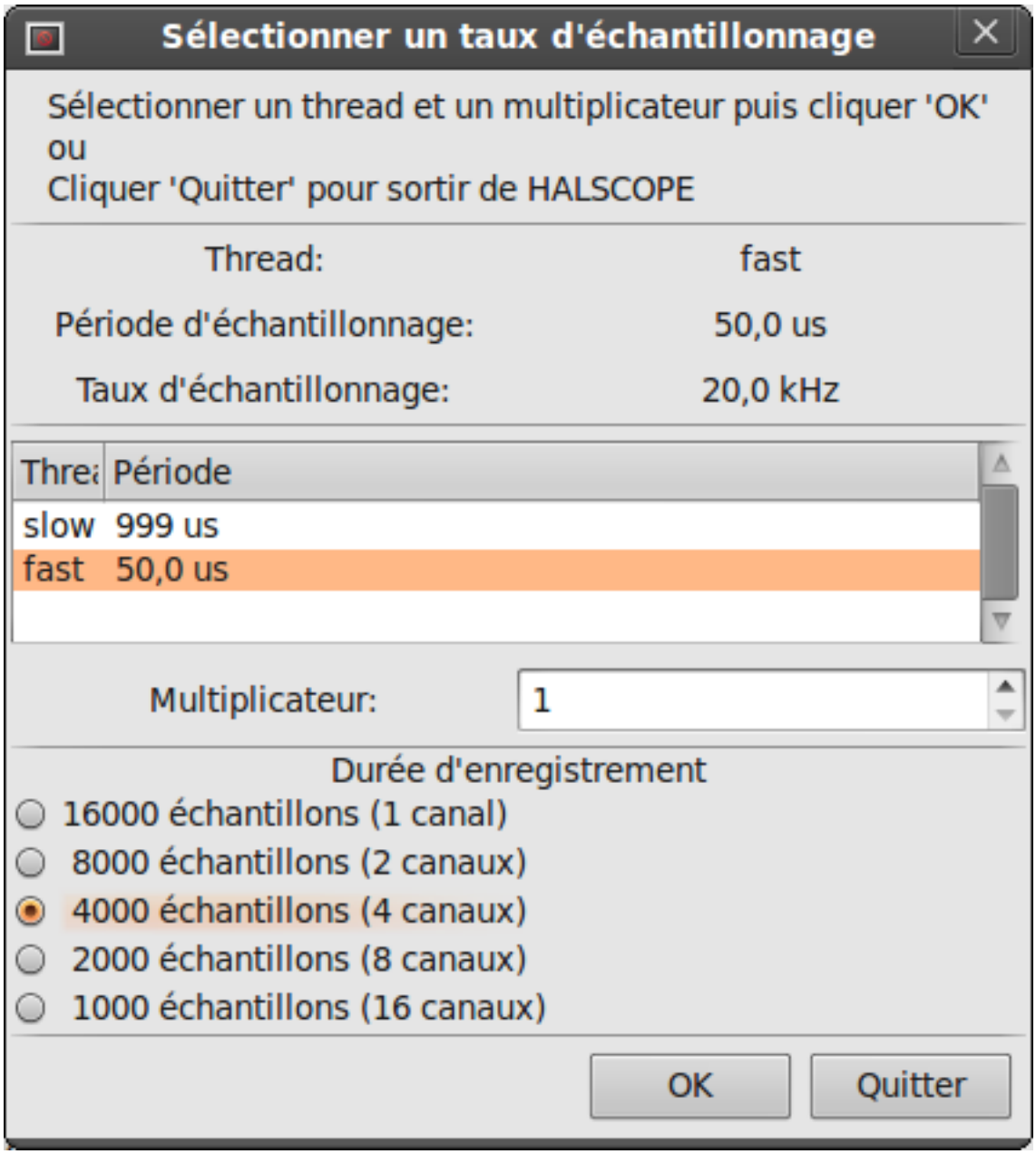


Figure 14.13: Dialogue de choix d'échantillonnage

14.3.6.7 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. Cliquer sur le canal 2, puis sur le bouton Canal Off sous le groupe vertical. Ensuite, cliquez sur le canal 3, le mettre Off et faire de même pour le canal 4. Même si les circuits sont éteints, ils sont encore en mémoire et restent connectés, en fait, nous continuerons a utiliser le canal 3 comme source de déclenchement. Pour ajouter de nouveaux canaux, sélectionner le canal 5, choisir la pin stepgen.0.dir, puis le canal 6 et sélectionner stepgen.0.step. Ensuite, cliquer sur mode Normal pour lancer le scope, ajustez le zoom horizontal à 10 ms par division. Vous devriez voir les impulsions de pas ralentir à la vitesse commandée approcher de zéro, puis la pin de direction changer d'état et les impulsions de pas se resserrer de nouveau en même temps que la vitesse augmente. Vous aurez peut

être besoin d'ajuster le gain sur le canal 1 afin de mieux voir l'action de la vitesse sur l'évolution des impulsions de pas. Le résultat devrait être proche de celui de la figure ci-dessous. Ce type de mesure est délicate car il y a un énorme écart d'échelle entre la fréquence des pas et l'action sur la vitesse, d'ou la courbe X-vel assez plate et les impulsions de pas très resserrées.

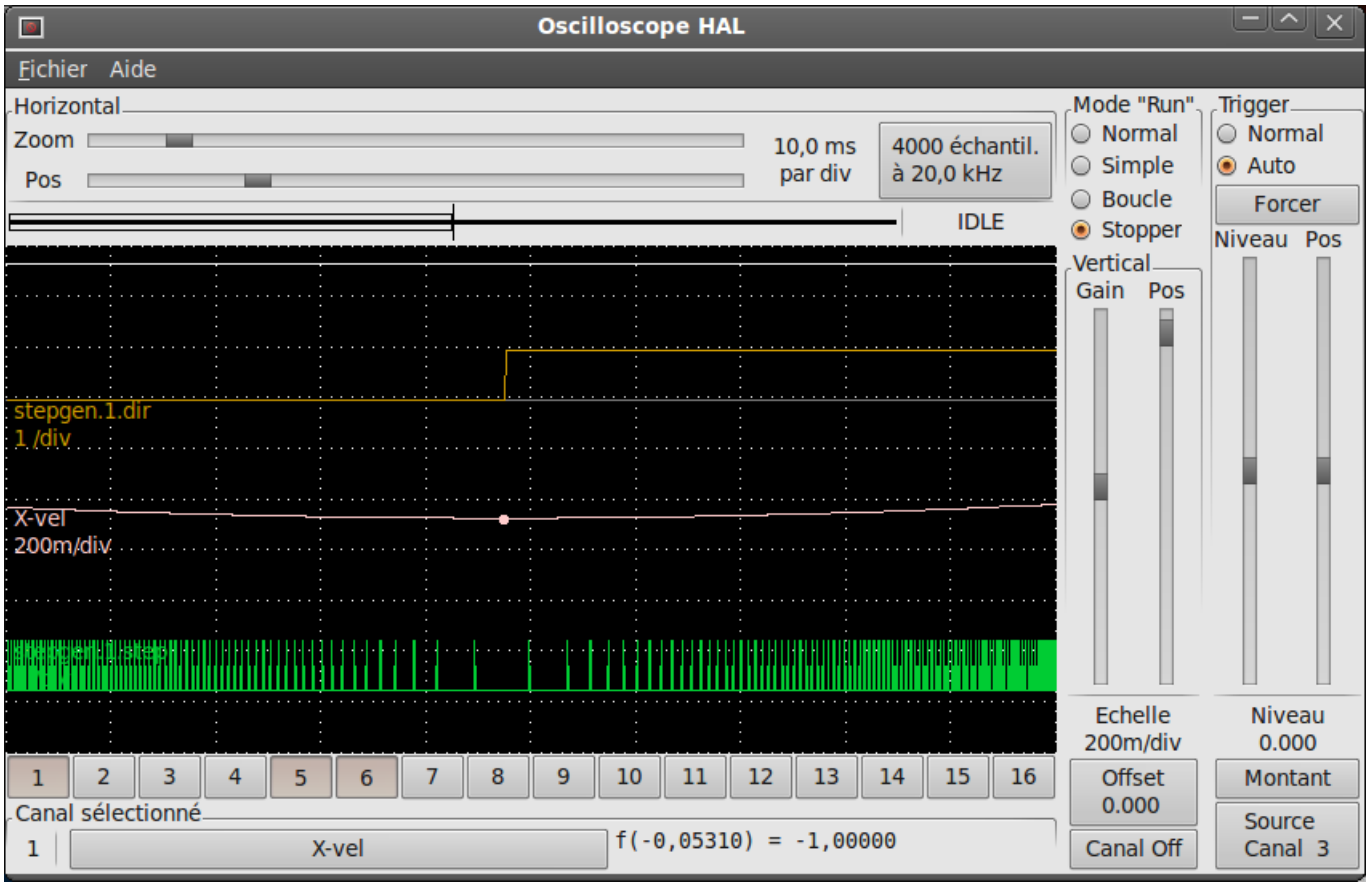


Figure 14.14: Observer les impulsions de pas

14.3.6.8 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)

14.4 Conventions générales

14.4.1 Les noms

Toutes les entités de HAL sont accessibles et manipulées par leurs noms, donc, documenter les noms des pins, signaux, paramètres, etc, est très important. Les noms dans HAL ont un maximum de

41 caractères de long (comme défini par HAL_NAME_LEN dans hal.h). De nombreux noms seront présentés dans la forme générale, avec un texte mis en forme <comme-cela> représentant les champs de valeurs diverses.

Quand les pins, signaux, ou paramètres sont décrits pour la première fois, leur nom sera précédé par leur type entre parenthèses (float) et suivi d'une brève description. Les définitions typiques de pins ressemblent à ces exemples:

(bit) parport.<portnum>.pin-<pinnum>-in
La HAL pin associée avec la broche physique d'entrée <pinnum> du connecteur db25.

(float) pid.<loopnum>.output
La sortie de la boucle PID.

De temps en temps, une version abrégée du nom peut être utilisée, par exemple la deuxième pin ci-dessus pourrait être appelée simplement avec .output quand cela peut être fait sans prêter à confusion.

14.4.2 Conventions générales de nommage

Le but des conventions de nommage est de rendre l'utilisation de HAL plus facile. Par exemple, si plusieurs interfaces de codeur fournissent le même jeu de pins et qu'elles sont nommées de la même façon, il serait facile de changer l'interface d'un codeur à un autre. Malheureusement, comme tout projet open-source, HAL est la combinaison de choses diversement conçues et comme les choses simples évoluent. Il en résulte de nombreuses incohérences. Cette section vise à remédier à ce problème en définissant certaines conventions, mais il faudra certainement un certain temps avant que tous les modules soient convertis pour les suivre.

Halcmd et d'autres utilitaires HAL de bas niveau, traitent les noms HAL comme de simples entités, sans structure. Toutefois, la plupart des modules ont une certaine structure implicite. Par exemple, une carte fournit plusieurs blocs fonctionnels, chaque bloc peut avoir plusieurs canaux et chaque canal, une ou plusieurs broches. La structure qui en résulte ressemble à une arborescence de répertoires. Même si halcmd ne reconnait pas la structure arborescente, la convention de nommage est un bon choix, elles lui permettra de regrouper ensemble, les items du même groupe, car il trie les noms. En outre, les outils de haut niveau peuvent être conçus pour reconnaitre de telles structures si les noms fournissent les informations nécessaires. Pour cela, tous les modules de HAL devraient suivrent les règles suivantes:

- Les points (.) séparent les niveaux hiérarchiques. C'est analogue à la barre de fraction (/) dans les noms de fichiers.
- Le tiret (-) sépare les mots ou les champs dans la même hiérarchie.
- Les modules HAL ne doivent pas utiliser le caractère souligné ou les casses mélangées. [8]
- Utiliser seulement des caractères minuscules, lettres et chiffres.

14.4.3 Conventions de nommage des pilotes de matériels

[9]

[8] Les caractères souslignés ont été enlevés, mais il reste quelques cas de mélange de casses, par exemple pid.0.Pgain au lieux de pid.0.p-gain.

[9] La plupart des pilotes ne suivent pas ces conventions dans la version 2.0. Ce chapitre est réellement un guide pour les développements futurs.

14.4.3.1 Noms de pin/paramètre

Les pilotes matériels devraient utiliser cinq champs (sur trois niveaux) pour obtenir un nom de pin ou de paramètre, comme le suivant:

```
<device-name>.<device-num>.<io-type>.<chan-num>.<specific-name>
```

Les champs individuels sont:

<device-name>
Le matériel avec lequel le pilote est sensé travailler. Il s'agit le plus souvent d'une carte d'interface d'un certain type, mais il existe d'autres possibilités.

<device-num>
Il est possible d'installer plusieurs cartes servo, ports parallèles ou autre périphérique matériel dans un ordinateur. Le numéro du périphérique identifie un périphérique spécifique. Les numéros de périphériques commencent à 0 et s'incrémentent.[10]

<io-type>
La plupart des périphériques fournissent plus d'un type d'I/O. Même le simple port parallèle a, à la fois plusieurs entrées et plusieurs sorties numériques. Les cartes commerciales plus complexes peuvent avoir des entrées et des sorties numériques, des compteurs de codeurs, des générateurs d'impulsions de pas ou de PWM, des convertisseurs numérique/analogique, analogique/numérique et d'autres possibilités plus spécifiques. Le I/O type est utilisé pour identifier le type d'I/O avec lequel la pin ou le paramètre est associé. Idéalement, les pilotes qui implémentent les mêmes type d'I/O, même sur des dispositifs très différents, devraient fournir un jeu de pins et de paramètres cohérents et de comportements identiques. Par exemple, toutes les entrées numériques doivent se comporter de la même manière quand elles sont vues de l'intérieur de HAL, indépendamment du périphérique.

<chan-num>
Quasiment tous les périphériques d'I/O ont plusieurs canaux, le numéro de canal chan-num identifie un de ceux ci. Comme les numéros de périphériques device-num, les numéros de canaux, chan-num, commencent à zéro et s'incrémentent.[11] Si plusieurs périphériques sont installés, les numéro de canaux des périphériques supplémentaires recommencent à zéro. Comme il est possible d'avoir un numéro de canal supérieur à 9, les numéros de canaux doivent avoir deux chiffres, avec un zéro en tête pour les nombres inférieur à 10 pour préserver l'ordre des tris. Certains modules ont des pins et/ou des paramètres qui affectent plusieurs canaux. Par exemple un générateur de PWM peut avoir quatre canaux avec quatre entrées duty-cycle indépendantes, mais un seul paramètre frequency qui contrôle les quatres canaux (à cause de limitations matérielles). Le paramètre frequency doit utiliser les numéros de canaux de 00-03.

<specific-name>
Un canal individuel d'I/O peut avoir une seule HAL pin associée avec lui, mais la plupart en ont plus. Par exemple, une entrée numérique a deux pins, une qui est l'état de la broche physique, l'autre qui est la même chose mais inversée. Cela permet au configurateur de choisir entre les deux états de l'entrée, active haute ou active basse. Pour la plupart des types d' entrée/sortie, il existe un jeu standard de broches et de paramètres, (appelé l'interface canonique) que le pilote doit implémenter. Les interfaces canoniques sont décrites dans ce chapitre qui leur est dédié.

[10] Certains matériels utilisent des cavaliers ou d'autres dispositifs pour définir une identification spécifique à chacun. Idéalement, le pilote fournit une manière à l'utilisateur de dire, le device-num 0 est spécifique au périphérique qui a l'ID XXX, ses sous-ensembles porterons tous un numéro commençant par 0. Mais à l'heure actuelle, certains pilotes utilisent l'ID directement comme numéro de périphérique. Ce qui signifie qu'il est possible d'avoir un périphérique Numéro 2, sans en avoir en Numéro 0. C'est un bug qui devrait disparaître en version 2.1.

[11] Une exception à la règle du numéro de canal commençant à zéro est le port parallèle. Ses HAL pins sont numérotées avec le numéro de la broche correspondante du connecteur DB-25. C'est plus pratique pour le câblage, mais non cohérent avec les autres pilotes. Il y a débat pour savoir si c'est un bogue ou une fonctionnalité.

motenc.0.encoder.2.position
La sortie position du troisième canal codeur sur la première carte Motenc.

stg.0.din.03.in
L'état de la quatrième entrée numérique sur la première carte Servo-to-Go.

ppmc.0.pwm.00-03.frequency
La fréquence porteuse utilisée sur les canaux PWM de 0 à 3 sur la première carte Pico Systems ppmc.

14.4.3.2 Noms des fonctions

Les pilotes matériels ont généralement seulement deux types de fonctions HAL, une qui lit l'état du matériel et met à jour les pins HAL, l'autre qui écrit sur le matériel en utilisant les données fournies sur les pins HAL. Ce qui devrait être nommé de la façon suivante:

```
<device-name>-<device-num>.<io-type>-<chan-num-range>.read|write
```

<device-name>
Le même que celui utilisé pour les pins et les paramètres.

<device-num>
Le périphérique spécifique auquel la fonction aura accès.

<io-type>
Optionnel. Une fonction peut accéder à toutes les d'entrées/sorties d'une carte ou, elle peut accéder seulement à un certain type. Par exemple, il peut y avoir des fonctions indépendantes pour lire les compteurs de codeurs et lire les entrées/sorties numériques. Si de telles fonctions indépendantes existent, le champ <io-type> identifie le type d'I/O auxquelles elles auront accès. Si une simple fonction lit toutes les entrés/sorties fournies par la carte, <io-type> n'est pas utilisé.[12]

<chan-num-range>
Optionnel. Utilisé seulement si l'entrée/sortie <io-type> est cassée dans des groupes et est accédée par différentes fonctions.

read|write
Indique si la fonction lit le matériel ou lui écrit.

motenc.0.encoder.read
Lit tous les codeurs sur la première carte motenc.

generic8255.0.din.09-15.read
Lit le deuxième port 8 bits sur la première carte d'entrées/sorties à base de 8255.

ppmc.0.write
Écrit toutes les sorties (générateur de pas, pwm, DAC et ADC) sur la première carte Pico Systems ppmc.

[12] Note aux programmeurs de pilotes: ne PAS implémenter des fonctions séparées pour différents types d'I/O à moins qu'elles ne soient interruptibles et puissent marcher dans des threads indépendants. Si l'interruption de la lecture d'un codeur pour lire des entrées numériques, puis reprendre la lecture du codeur peut poser problème, alors implémentez une fonction unique qui fera tout.

14.4.4 Périphériques d'interfaces canoniques

Les sections qui suivent expliquent les pins, paramètres et fonctions qui sont fournies par les périphériques canoniques. Tous les pilotes de périphériques HAL devraient fournir les mêmes pins et paramètres et implémenter les mêmes comportements.

Noter que seuls les champs <io-type> et <specific-name> sont définis pour un périphérique canonique. Les champs <device-name>, <device-num> et <chan-num> sont définis en fonction des caractéristiques du périphérique réel.

14.4.5 Entrée numérique (Digital Input)

L'entrée numérique canonique (I/O type: digin) est assez simple.

14.4.5.1 Pins

(bit) in
État de l'entrée matérielle.

(bit) in-not
État inversé de l'entrée matérielle.

14.4.5.2 Paramètres

Aucun

14.4.5.3 Fonctions

(funct) read
Lire le matériel et ajuster les HAL pins in et in-not.

14.4.6 Sortie numérique (Digital Output)

La sortie numérique canonique est également très simple (I/O type: digout).

14.4.6.1 Pins

(bit) out
Valeur à écrire (éventuellement inversée) sur une sortie matérielle.

14.4.6.2 Paramètres

(bit) invert
Si TRUE, out est inversée avant écriture sur la sortie matérielle.

14.4.6.3 Fonctions

(funct) write
Lit out et invert et ajuste la sortie en conséquence.

14.4.7 Entrée analogique (Analog Input)

L'entrée analogique canonique (I/O type: adcin). Devrait être utilisée pour les convertisseurs analogiques/nu
qui convertissent par exemple, les tensions en une échelle continue de valeurs.

14.4.7.1 Pins

(float) value
Lecture du matériel, avec mise à l'échelle ajustée par les paramètres scale et offset. Value = ((lecture entrée, en unités dépendantes du matériel) x scale) - offset

14.4.7.2 Paramètres

(float) scale
La tension d'entrée (ou l'intensité) sera multipliée par scale avant d'être placée dans value.

(float) offset
Sera soustrait à la tension d'entrée (ou l'intensité) après que la mise à l'échelle par scale ait été appliquée.

(float) bit_weight
Valeur du bit le moins significatif (LSB). C'est effectivement, la granularité de lecture en entrée.

(float) hw_offset
Valeur présente sur l'entrée quand aucune tension n'est appliquée sur la pin.

14.4.7.3 Fonctions

(funct) read
Lit les valeurs de ce canal d'entrée analogique. Peut être utilisé pour lire un canal individuellement, ou pour lire tous les canaux à la fois.

14.4.8 Sortie analogique (Analog Output)

La sortie analogique canonique (I/O Type: adcout). Elle est destinée à tout type de matériel capable de sortir une échelle plus ou moins étendue de valeurs. Comme par exemple les convertisseurs numérique/analogique ou les générateurs de PWM.

14.4.8.1 Pins

(float) value
La valeur à écrire. La valeur réelle sur la sortie matérielle dépends de la mise à l'échelle des paramètres d'offset.

(bit) enable
Si fausse, la sortie matérielle passera à 0, indépendamment de la pin value.

14.4.8.2 Paramètres

(float) offset
Sera ajouté à value avant l'actualisation du matériel.

(float) scale
Doit être défini de sorte qu'une entrée avec 1 dans value produira 1V

(float) high_limit
(optionnel) Quand la valeur en sortie matérielle est calculée, si value + offset est plus grande que high_limit, alors high_limit lui sera substitué.

(float) low_limit
(optionnel) Quand la valeur en sortie matérielle est calculée, si value + offset est plus petite que low_limit, alors low_limit lui sera substitué.

(float) bit_weight
(optionnel) La valeur du bit le moins significatif (LSB), en Volts (ou mA, pour les sorties courant)

(float) hw_offset
(optionnel) La tension actuelle (ou l'intensité) présente sur la sortie quand 0 est écrit sur le matériel.

14.4.8.3 Fonctions

(funct) write
Ecrit la valeur calculée sur la sortie matérielle. Si enable est FALSE, la sortie passera à 0, indépendamment des valeurs de value, scale et offset . La signification de 0 dépend du matériel. Par exemple, un convertisseur A/D 12 bits peut vouloir écrire 0x1FF (milieu d'échelle) alors que le convertisseur D/A reçoit 0 Volt de la broche matérielle. Si enable est TRUE, l'échelle, l'offset et la valeur sont traités et (scale _ value) + offset sont envoyés à la sortie du DAC . Si enable est FALSE, la sortie passe à 0.

14.5 Les fonctionnalités de Halshow

14.5.1 Le script Halshow

Le script halshow peut vous aider à retrouver votre chemin dans un HAL en fonctionnement. Il s'agit d'un système très spécialisé qui doit se connecter à un HAL en marche. Il ne peut pas fonctionner seul car il repose sur la capacité de HAL de rapporter ce qu'il connaît de lui même par la librairie d'interface de halcmd. Chaque fois que halshow fonctionne avec une configuration de LinuxCNC différente, il sera différent.

Comme nous le verrons bientôt, cette capacité de HAL de se documenter lui même est un des facteurs clés pour arriver à un système CNC optimum.

On peut accéder à Halshow depuis Axis, pour cela, aller dans le menu Machine puis choisir Afficher la configuration de HAL.

14.5.1.1 Zone de l'arborescence de Hal

La gauche de l'écran que montre la figure ci-dessous est une arborescence, un peu comme vous pouvez le voir avec certains navigateurs de fichiers. Sur la droite, une zone avec deux onglets: MONTRER et WATCH.

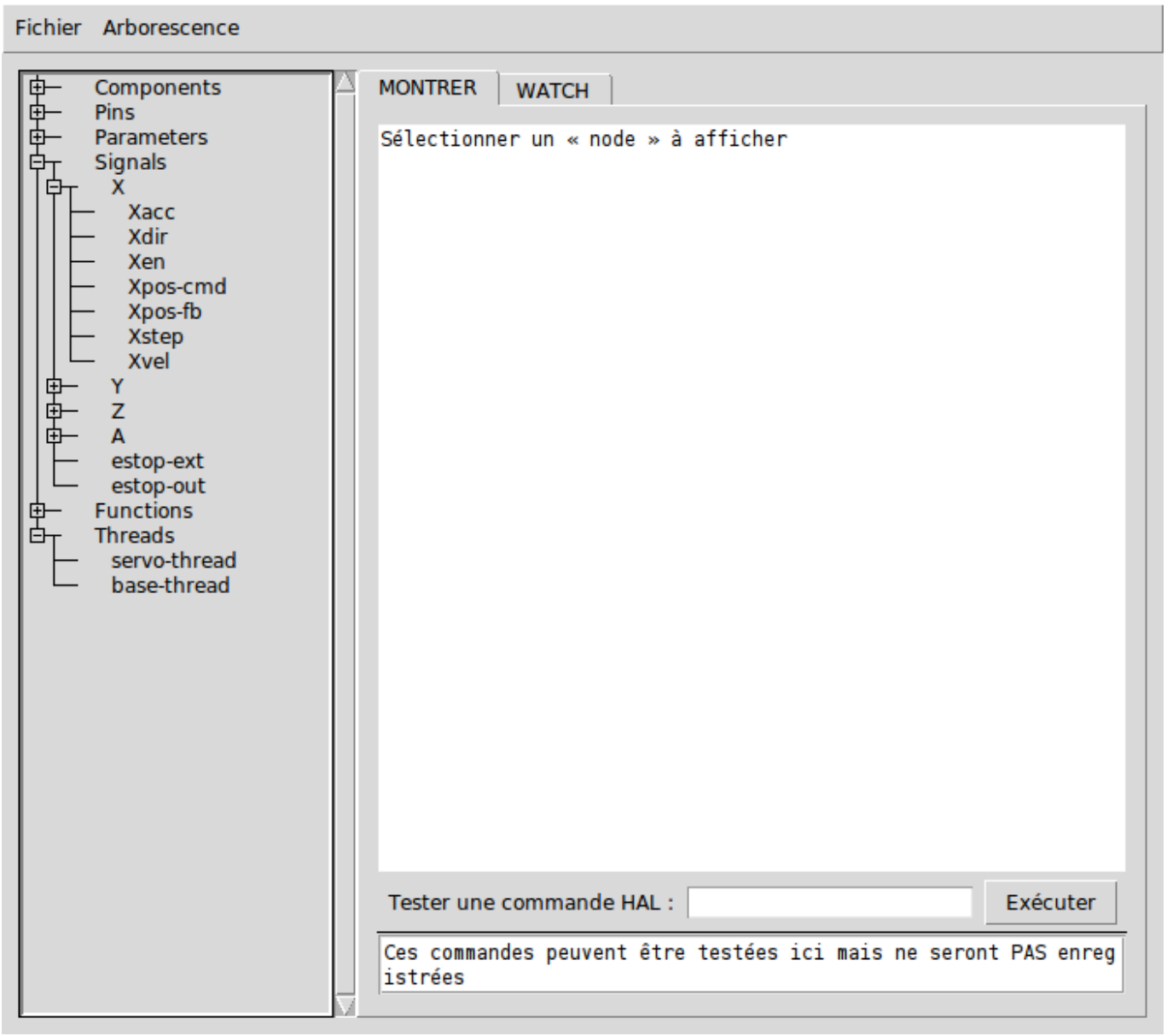


Figure 14.15: La fenêtre de Halshow

L'arborescence montre toutes les parties principales de HAL. En face de chacune d'entre elles, se trouve un petit signe + ou - dans une case. Cliquer sur le signe plus pour déployer cette partie de

l'arborescence et affichera son contenu. Si cette case affiche un signe moins, cliquer dessus repliera cette section de l'arborescence.

Il est également possible de déployer et de replier l'arborescence complète depuis le menu Arborescence.

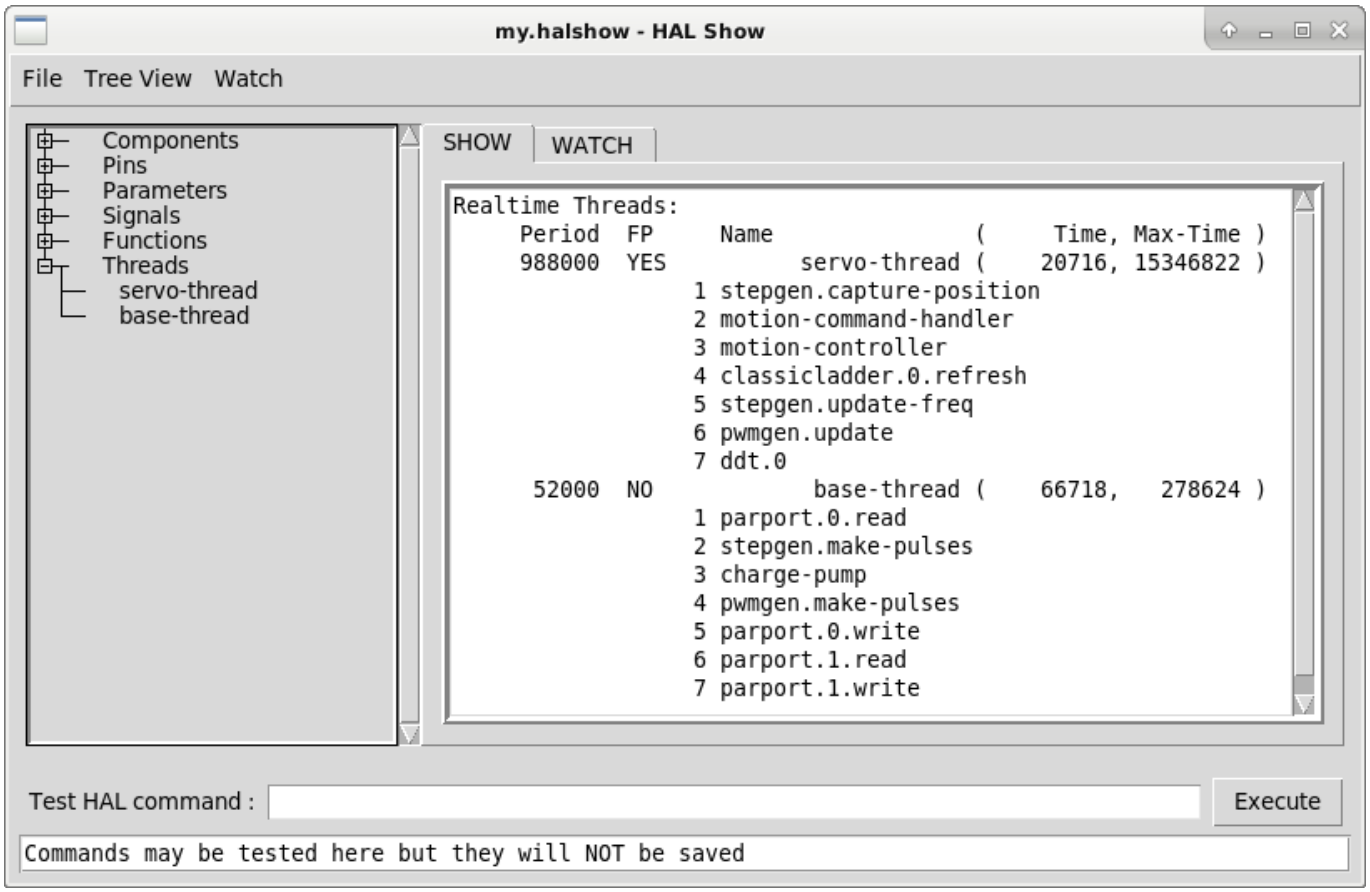


Figure 14.16: L'onglet Montrer

14.5.1.2 Zone de l'onglet MONTRER

En cliquant sur un nom dans l'arborescence plutôt que sur son signe plus ou moins, par exemple le mot Components, HAL affichera tout ce qu'il connait du contenu de celui-ci. La figure halshow montre une liste comme celle que vous verrez si vous cliquez sur Components avec une carte servo standard m5i20 en fonctionnement. L'affichage des informations est exactement le même que celui des traditionnels outils d'analyse de HAL en mode texte. L'avantage ici, c'est que nous y avons accès d'un clic de souris. Accès qui peut être aussi large ou aussi focalisé que vous le voulez.

Si nous examinons de plus près l'affichage de l'arborescence, nous pouvons voir que les six éléments principaux peuvent tous être déployés d'au moins un niveau. Quand ces niveaux sont à leur tour déployés vous obtenez une information de plus en plus focalisée en cliquant sur le nom des éléments dans l'arborescence. Vous trouverez que certaines hal pins et certains paramètres affichent plusieurs réponses. C'est dû à la nature des routines de recherche dans halcmd lui même. Si vous cherchez une pin, vous pouvez en trouver deux comme cela:

```
Component Pins:
Owner  Type  Dir  Value  Name
06     bit    -W   TRUE  parport.0.pin-10-in
06     bit    -W  FALSE  parport.0.pin-10-in-not
```

Le deuxième nom de pin contient le nom complémenté du premier.

Dans le bas de l'onglet Montrer, un champ de saisie permet de jouer sur le fonctionnement de HAL. Les commandes que vous entrez ici et leur effet sur HAL, ne sont pas enregistrés. Elles persisteront tant que LinuxCNC tournera, mais disparaîtront dès son arrêt.

Le champ de saisie marqué Tester une commande HAL: acceptera n'importe quelle commande valide pour halcmd. Elles incluent:

- Loadrt, unloadrt (chargement / déchargement en temps réel du module)
- Loadusr, unloadusr (chargement / déchargement de l'espace utilisateur des composants)
- addf, delf (ajout / suppression d'une fonction de / vers un fil en temps réel)
- net (créer une connexion entre deux ou plusieurs articles)
- setp (jeu de paramètres (ou broches) à une valeur)

Ce petit éditeur entrera une commande à chaque fois que vous presserez Entrée ou que vous cliquerez sur le bouton Exécuter. Si une commande y est mal formée, un dialogue d'erreur s'affichera. Si vous n'êtes pas sûr de savoir comment formuler une commande, vous trouverez la réponse dans la documentation de halcmd et des modules spécifiques avec lesquels vous travaillez.

Nous allons utiliser cet éditeur pour ajouter un module différentiel à HAL et le connecter à la position d'un axe pour voir le ratio de changement de position, par exemple, l'accélération. Il faut d'abord charger un module de HAL nommé blocks, l'ajouter au thread servo et le connecter à la pin position d'un axe. Une fois cela fait, nous pourrons retrouver la sortie du différenciateur dans halscope. Alors allons-y. (oui j'ai vérifié).

Ndt: le message qui s'affiche au chargement de blocks ne l'empêche pas de fonctionner.

```
loadrt blocks ddt=1
```

Maintenant, regardez dans components, vous devriez y voir blocks.

```
    Loaded HAL Components:
    ID Type        Name
    10 User halcmd29800
    09 User halcmd29374
    08   RT      blocks
    06   RT hal_parport
    05   RT    scope_rt
    04   RT     stepgen
    03   RT      motmod
    02 User   iocontrol
```

Effectivement, il est là. Dans notre cas l'id est 08. Ensuite nous devons savoir quelles fonctions sont disponibles avec lui, nous regardons dans Functions.

```
    Exported Functions:
    Owner  CodeAddr      Arg  FP Users Name
      08   E0B97630 E0DC7674 YES     0 ddt.0
      03   E0DEF83C 00000000 YES     1 motion-command-handler
      03   E0DF0BF3 00000000 YES     1 motion-controller
      06   E0B541FE E0DC75B8  NO     1 parport.0.read
      06   E0B54270 E0DC75B8  NO     1 parport.0.write
      06   E0B54309 E0DC75B8  NO     0 parport.read-all
      06   E0B5433A E0DC75B8  NO     0 parport.write-all
      05   E0AD712D 00000000  NO     0 scope.sample
      04   E0B618C1 E0DC7448 YES     1 stepgen.capture-position
      04   E0B612F5 E0DC7448  NO     1 stepgen.make-pulses
      04   E0B614AD E0DC7448 YES     1 stepgen.update-freq
```

Ici, nous cherchons owner #08 et voyons que blocks a exporté une fonction nommée ddt.0. Nous devrions être en mesure d'ajouter ddt.0 au thread servo et il fera ses calculs chaque fois que le thread sera mis à jour. Encore une fois recherchons la commande addf et on voit qu'elle utilise trois arguments comme cela:

```
addf <functname> <threadname> [<position>]
```

Nous connaissons déjà functname=ddt.0, pour trouver le nom du thread, déployons l'arborescence des Threads. Nous y trouvons deux threads, servo-thread et base-thread. La position de ddt.0 dans le thread n'est pas critique. Passons la commande:

```
addf ddt.0 servo-thread
```

Comme c'est juste pour visualiser, nous laissons la position en blanc pour obtenir la dernière position dans le thread. La figure sur la commande addf affiche l'état de halshow après que cette commande a été exécutée.

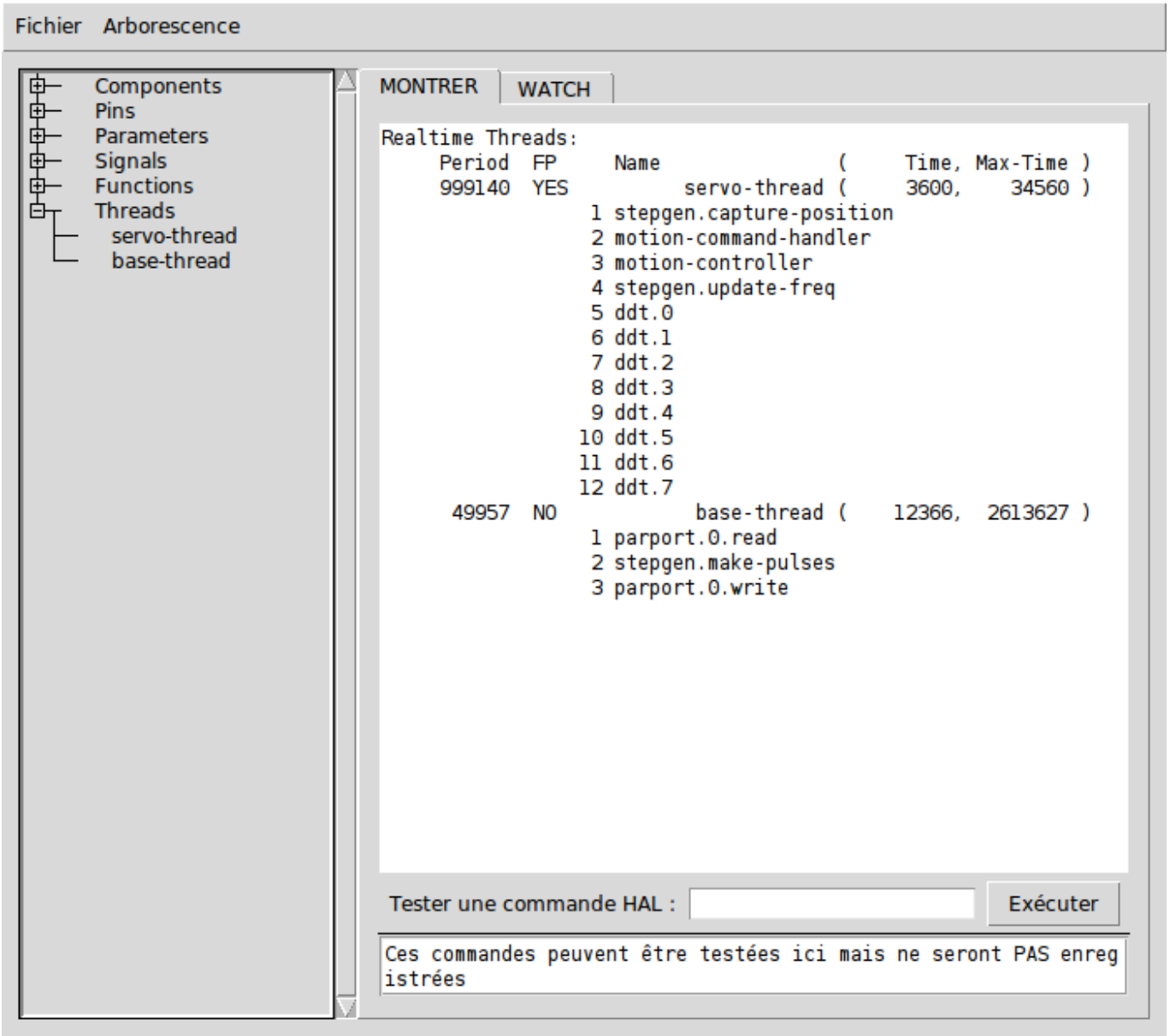


Figure 14.17: Commande Addf

Ensuite, nous devons connecter ce bloc à quelque chose. Mais comment savoir quelles pins sont disponibles? La réponse se trouve dans l'arbre, en regardant sous Pins. On y trouve ddt et on voit:

```
    Component Pins:
    Owner Type  Dir Value       Name
    08    float R-  0.00000e+00 ddt.0.in
    08    float -W  0.00000e+00 ddt.0.out
```

Cela semble assez facile à comprendre, mais à quel signal ou pin voulons-nous nous connecter, ça pourrait être une pin d'axe, une pin de stepgen, ou un signal. On vois cela en regardant dans axis.0.

```
    Component Pins:
    Owner Type  Dir Value       Name
    03    float -W  0.00000e+00 axis.0.motor-pos-cmd ==> Xpos-cmd
```

Donc, il semble que Xpos-cmd devrait être un bon signal à utiliser. Retour à l'éditeur et entrons la commande suivante:

```
linksp Xpos-cmd ddt.0.in
```

Maintenant si on regarde le signal Xpos-cmd dans l'arbre, on voit ce qu'on a fait.

```
    Signals:
    Type Value Name
    float 0.00000e+00 Xpos-cmd
    <== axis.0.motor-pos-cmd
    ==> ddt.0.in
    ==> stepgen.0.position-cmd
```

Nous voyons que ce signal provient de axis.0.motor-pos-cmd et va, à la fois, sur ddt.0.in et sur stepgen.0.position-cmd. En connectant notre bloc au signal nous avons évité les complications avec le flux normal de cette commande de mouvement.

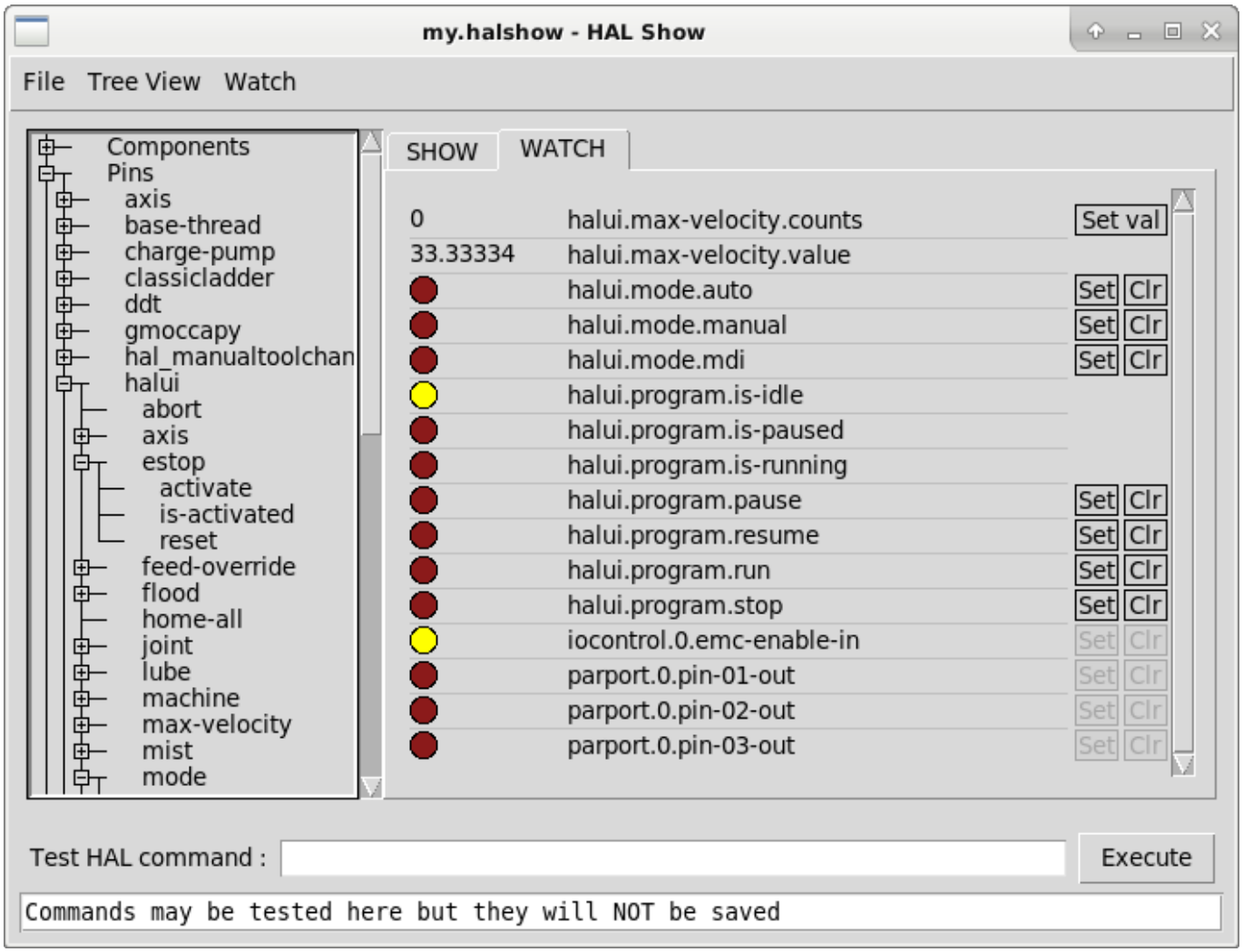
La zone de l'onglet Montrer utilise halcmd pour découvrir ce qui se passe à l'intérieur de HAL pendant son fonctionnement. Il vous donne une information complète de ce qu'il découvre. Il met aussi à jour dès qu'une commande est envoyée depuis le petit éditeur pour modifier ce HAL. Il arrive un temps ou vous voulez autre chose d'affiché, sans la totalité des informations disponibles dans cette zone. C'est la grande valeur de l'onglet WATCH d'offrir cela graphiquement.

14.5.1.3 Zone de l'onglet WATCH

En cliquant sur l'onglet Watch, une zone vide s'affichera. [13] Vous pouvez ajouter des pins ou des signaux quand l'onglet Watch est ouvert, en cliquant sur leurs noms. La figure 4 montre cette zone avec plusieurs signaux de type bit. Parmis ces signaux, les enable-out pour les trois premiers axes et deux de la branche iocontrol, les signaux estop. Notez que les axes ne sont pas activés même si les signaux estop disent que LinuxCNC n'est pas en estop. Un bref regard sur TkLinuxCNC en arrière plan, montre que l'état de LinuxCNC est ESTOP RESET. L'activation des amplis ne deviendra pas vraie tant que la machine ne sera pas mise en marche.

L'onglet WATCH

[13] Le taux de rafraîchissement de la zone Watch est plus lent que celui de Halmeter ou de Halscope. Si vous avez besoin d'une bonne résolution dans le timming des signaux, ces outils sont plus efficaces.



Les cercles de deux couleurs, simili Leds, sont toujours bruns foncé quand un signal est faux. Elle sont jaunes quand le signal est vrai. Quand une pin ou un signal est sélectionné mais n'est pas de type bit, sa valeur numérique s'affiche.

Watch permet de visualiser rapidement le résultat de tests sur des contacts ou de voir l'effet d'un changement que vous faites dans LinuxCNC en utilisant l'interface graphique. Le taux de rafraîchissement de Watch est un peu trop lent pour visualiser les impulsions de pas d'un moteur mais vous pouvez l'utiliser si vous déplacez un axe très lentement ou par très petits incréments de distance. Si vous avez déjà utilisé IO_Show dans LinuxCNC, la page de Watch de halshow peut être réglée pour afficher ce que fait le port parallèle.

14.6 Les composants de HAL

14.6.1 Composants de commandes et composants de l'espace utilisateur

Certaines de ces descriptions sont plus approfondies dans leurs pages man. Certaines y auront une description exhaustive, d'autres, juste une description limitée. Chaque composant a sa man page. La liste ci-dessous, montre les composants existants, avec le nom et le N° de section de leur page man. Par exemple dans une console, tapez man axis pour accéder aux informations de la man page d'Axis. Ou peut être man 1 axis, si le système exige le N° de section des man pages.

- axis - L'interface graphique AXIS pour LinuxCNC (The Enhanced Machine Controller).

- axis-remote - Interface de télécommande d'AXIS.
- comp - Crée, compile et installe des composants de HAL.
- linuxcnc - LINUXCNC (The Enhanced Machine Controller).
- gladevcp - Panneau de contrôle virtuel pour LinuxCNC, repose sur Glade, Gtk et les widgets HAL.
- gs2 - composant de l'espace utilisateur de HAL, pour le variateur de fréquence GS2 de la société Automation Direct.
- halcmd - Manipulation de HAL, depuis la ligne de commandes.
- hal_input - Contrôler des pins d'entrée de HAL avec n'importe quelle matériel supporté par Linux, y compris les matériels USB HID.
- halmeter - Observer les pins de HAL, ses signaux et ses paramètres.
- halrun - Manipulation de HAL, depuis la ligne de commandes.
- halsampler - Échantillonner des données temps réel depuis HAL.
- halstreamer - Créer un flux de données temps réel dans HAL depuis un fichier.
- halui - Observer des pins de HAL et commander LinuxCNC au travers d'NML.
- io - Accepte les commandes NML I/O, interagi avec HAL dans l'espace utilisateur.
- iocontrol - Accepte les commandes NML I/O, interagi avec HAL dans l'espace utilisateur.
- pyvcp - Panneau de Contrôle Virtuel pour LinuxCNC (Python Virtual Control Panel).
- shuttle - Contrôle des pins de HAL avec la manette ShuttleXpress et ShuttlePRO, de la société Contour Design.

14.6.2 Composants temps réel et modules du noyau

Certaines de ces descriptions sont plus approfondies dans leur man page. Certaines auront juste une description limitée. Chaque composant a sa man page. A partir de cette liste vous connaîtrez quels composants existent avec le nom et le N° de leur man page permettant d'avoir plus de détails.

Note
Si le composant requière un thread avec flottant, c'est normalement le plus lent, soit servo-thread.

14.6.2.1 Composants du coeur de LinuxCNC

- motion - Accepte les commandes de mouvement NML, interagi en temps réel avec HAL.
- axis - Commandes de mouvement NML acceptées, intéragi en temps réel avec HAL
- classicladder - Automate temps réel programmable en logique Ladder.
- gladevcp - Affiche un panneaux de contrôle virtuel construit avec GladeVCP.
- threads - Crée des threads de HAL temps réel.

14.6.2.2 Composants binaires et logiques

- and2 - Porte AND (ET) à deux entrées.
- not - Inverseur.
- or2 - Porte OR (OU) à deux entrées.
- xor2 - Porte XOR (OU exclusif) à deux entrées.
- debounce - Filtre une entrée digitale bruitée (typiquement antirebond).
- edge _ Détecteur de front.
- flipflop - Bascule D.
- oneshot - Générateur d'impulsion monostable. Crée sur sa sortie une impulsion de longueur variable quand son entrée change d'état.
- logic - Composant expérimental de logique générale.
- lut5 - Fonction logique arbitraire à cinq entrées, basée sur une table de correspondance.
- match8 - Détecteur de coïncidence binaire sur 8 bits.
- select8 - Détecteur de coïncidence binaire sur 8 bits.

14.6.2.3 Composants arithmétiques et flottants

- abs - Calcule la valeur absolue et le signe d'un signal d'entrée.
- blend - Provoque une interpolation linéaire entre deux valeurs
- comp - Comparateur à deux entrées avec hystérésis.
- constant - Utilise un paramètre pour positionner une pin.
- sum2 - Somme de deux entrées (chacune avec son gain) et d'un offset.
- counter - Comptage d'impulsions d'entrée (obsolète).

Utiliser le composant encoder avec … counter-mode = TRUE. Voir la section codeur.

- updown - Compteur/décompteur avec limites optionnelles et bouclage en cas de dépassement.
- ddt - Calcule la dérivée de la fonction d'entrée.
- deadzone - Retourne le centre si il est dans le seuil.
- hypot - Calculateur d'hypoténuse à trois entrées (distance Euclidienne).
- mult2 - Le produit de deux entrées.
- mux16 - Sélection d'une valeur d'entrée sur seize.
- mux2 - Sélection d'une valeur d'entrée sur deux.
- mux4 - Sélection d'une valeur d'entrée sur quatre.
- mux8 - Sélection d'une valeur d'entrée sur huit.
- near - Détermine si deux valeurs sont à peu près égales.
- offset - Ajoute un décalage à une entrée et la soustrait à la valeur de retour.

- integ - Intégrateur.
- invert - Calcule l'inverse du signal d'entrée.
- wcomp - Comparateur à fenêtre.
- weighted_sum - Converti un groupe de bits en un entier.
- biquad - Filtre biquad IIR
- lowpass - Filtre passe-bas.
- limit1 - Limite le signal de sortie pour qu'il soit entre min et max. [14]
- limit2 - Limite le signal de sortie pour qu'il soit entre min et max. Limite sa vitesse de montée à moins de MaxV par seconde. [15]
- limit3 - Limite le signal de sortie pour qu'il soit entre min et max. Limite sa vitesse de montée à moins de MaxV par seconde. Limite sa dérivée seconde à moins de MaxA par seconde carré. [16]
- maj3 - Calcule l'entrée majoritaire parmi 3.
- scale - Applique une échelle et un décalage à son entrée.

14.6.2.4 Conversions de type

- conv_bit_s32 - Converti une valeur de bit vers s32 (entier 32 bits signé).
- conv_bit_u32 - Converti une valeur de bit vers u32 (entier 32 bit non signé).
- conv_float_s32 - Converti la valeur d'un flottant vers s32.
- conv_float_u32 - Converti la valeur d'un flottant vers u32.
- conv_s32_bit - Converti une valeur de s32 en bit.
- conv_s32_float - Converti une valeur de s32 en flottant.
- conv_s32_u32 - Converti une valeur de s32 en u32.
- conv_u32_bit - Converti une valeur de u32 en bit.
- conv_u32_float - Converti une valeur de u32 en flottant.
- conv_u32_s32 - Converti une valeur de u32 en s32.

14.6.2.5 Pilotes de matériel

- hm2_7i43 - Pilote HAL pour les cartes Mesa Electronics 7i43 EPP, toutes les cartes avec HostMot2.
- hm2_pci - Pilote HAL pour les cartes Mesa Electronics 5i20, 5i22, 5i23, 4i65 et 4i68, toutes les cartes avec micro logiciel HostMot2.
- hostmot2 - Pilote HAL pour micro logiciel Mesa Electronics HostMot2.
- mesa_7i65 - Support pour la carte huit axes Mesa 7i65 pour servomoteurs.
- pluto_servo - Pilote matériel et micro programme pour la carte Pluto-P parallel-port FPGA, utilisation avec servomoteurs.

[14] Lorsque l'entrée est une position, cela signifie que la position est limitée.
[15] Lorsque l'entrée est une position, cela signifie que la position et la vitesse sont limitées.
[16] Lorsque l'entrée est une position, cela signifie que la position, la vitesse et l'accélération sont limitées.

- pluto_step - Pilote matériel et micro programme pour la carte Pluto-P parallel-port FPGA, utilisation avec moteurs pas à pas.
- thc - Contrôle de la hauteur de torche, en utilisant une carte Mesa THC.
- serport - Pilote matériel pour les entrées/sorties numériques de port série avec circuits 8250 et 16550.

14.6.2.6 Composants cinématiques

- kins - Définition des cinématiques pour linuxcnc.
- gantrykins - Module de cinématique pour un seul axe à articulations multiples.
- genhexkins - Donne six degrés de liberté en position et en orientation (XYZABC). L'emplacement des moteurs est défini au moment de la compilation.
- genserkins - Cinématique capable de modéliser une bras manipulateur avec un maximum de 6 articulations angulaires.
- maxkins - Cinématique d'une fraiseuse 5 axes nommée max, avec tête inclinable (axe B) et un axe rotatif horizontal monté sur la table (axe C). Fourni les mouvements UVW dans le système de coordonnées système basculé. Le fichier source, maxkins.c, peut être un point de départ utile pour d'autres systèmes 5 axes.
- tripodkins - Les articulations représentent la distance du point contrôlé à partir de trois emplacements prédéfinis (les moteurs), ce qui donne trois degrés de liberté en position (XYZ).
- trivkins - Il y a une correspondance 1:1 entre les articulations et les axes. La plupart des fraiseuses standard et des tours utilisent ce module de cinématique triviale.
- pumakins - Cinématique pour robot style PUMA.
- rotatekins - Les axes X et Y sont pivotés de 45 degrés par rapport aux articulations 0 et 1.
- scarakins - Cinématique des robots de type SCARA.

14.6.2.7 Composants de contrôle moteur

- at_pid - Contrôleur Proportionnelle/Intégrale/dérivée avec réglage automatique.
- pid - Contrôleur Proportionnelle/Intégrale/dérivée.
- pwmgen - Générateur logiciel de PWM/PDM, voir la section PWMgen
- encoder - Comptage logiciel de signaux de codeur en quadrature, voir la section codeur
- stepgen - Générateur d'impulsions de pas logiciel, voir la section stepgen

14.6.2.8 BLDC and 3-phase motor control

- bldc_hall3 - Commutateur bipolaire trapézoïdal à 3 directions pour moteur sans balais (BLDC) avec capteurs de Hall.
- clarke2 - Transformation de Clarke, version à deux entrées.
- clarke3 - Transformation de Clarke, à 3 entrées vers cartésien.
- clarkeinv - Transformation de Clarke inverse.

14.6.2.9 Autres composants

- charge_pump - Crée un signal carré destiné à l'entrée pompe de charge de certaines cartes de contrôle. Le composant charg_pump doit être ajouté à base thread. Quand il est activé, sa sortie est haute pour une période puis basse pour une autre période. Pour calculer la fréquence de sortie faire 1/(durée de la période en secondes * 2) = fréquence en Hz. Par exemple, si vous avez une période de base de 100000ns soit 0.0001 seconde, la formule devient: 1/(0.0001 * 2) = 5000 Hz ou 5kHz.
- encoder_ratio - Un engrenage électronique pour synchroniser deux axes.
- estop_latch - Verrou d'Arrêt d'Urgence.
- feedcomp - Multiplie l'entrée par le ratio vitesse courante / vitesse d'avance travail.
- gearchange - Sélectionne une grandeur de vitesse parmi deux.
- ilowpass - Filtre passe-bas avec entrées et sorties au format entier.

  Sur une machine ayant une grande accélération, un petit jog peut s'apparenter à une avance par pas. En intercalant un filtre ilowpass entre la sortie de comptage du codeur de la manivelle et l'entrée jog-counts de l'axe, le mouvement se trouve lissé.

  Choisir prudemment l'échelle, de sorte que durant une simple session, elle ne dépasse pas environ 2e9/scale impulsions visibles sur le MPG. Choisir le gain selon le niveau de douceur désiré. Diviser les valeurs de axis.N.jog-scale par l'échelle.
- joyhandle - Définit les mouvements d'un joypad non linéaire, zones mortes et échelles.
- knob2float - Convertisseur de comptage (probablement d'un codeur) vers une valeur en virgule flottante.
- minmax - Suiveur de valeurs minimum et maximum de l'entrée vers les sorties.
- sample_hold - Échantillonneur bloqueur.
- sampler - Échantillonneur de données de HAL en temps réel.
- siggen - Générateur de signal, voir la section siggen
- sim_encoder - Codeur en quadrature simulé, voir la section codeur simulé
- sphereprobe - Sonde hémisphérique.
- steptest - Utilisé par Stepconf pour permettre de tester les valeurs d'accélération et de vitesse d'un axe.
- streamer - Flux temps réel depuis un fichier vers HAL.
- supply - Set output pins with values from parameters (obsolète).
- threadtest - Composant de HAL pour tester le comportement des threads.
- time - Compteur de temps écoulé HH:MM:SS avec entrée actif.
- timedelay - L'équivalent d'un relais temporisé.
- timedelta - Composant pour mesurer le comportement temporel des threads.
- toggle2nist - Bouton à bascule pour logique NIST.
- toggle - Bouton à bascule NO/NF à partir d'un bouton poussoir momentané.
- tristate_bit - Place un signal sur une pin d'I/O seulement quand elle est validée, similaire à un tampon trois états en électronique.
- tristate_float - Place un signal sur une pin d'I/O seulement quand elle est validée, similaire à un tampon trois états en électronique.
- watchdog - Moniteur de fréquence (chien de garde) sur 1 à 32 entrées.

14.6.3 Appels à l’API de HAL (liste de la section 3 des man pages)

hal_add_funct_to_thread.3hal
hal_bit_t.3hal
hal_create_thread.3hal
hal_del_funct_from_thread.3hal
hal_exit.3hal
hal_export_funct.3hal
hal_float_t.3hal
hal_get_lock.3hal
hal_init.3hal
hal_link.3hal
hal_malloc.3hal
hal_param_bit_new.3hal
hal_param_bit_newf.3hal
hal_param_float_new.3hal
hal_param_float_newf.3hal
hal_param_new.3hal
hal_param_s32_new.3hal
hal_param_s32_newf.3hal
hal_param_u32_new.3hal
hal_param_u32_newf.3hal
hal_parport.3hal
hal_pin_bit_new.3hal
hal_pin_bit_newf.3hal
hal_pin_float_new.3hal
hal_pin_float_newf.3hal
hal_pin_new.3hal
hal_pin_s32_new.3hal
hal_pin_s32_newf.3hal
hal_pin_u32_new.3hal
hal_pin_u32_newf.3hal
hal_ready.3hal
hal_s32_t.3hal
hal_set_constructor.3hal
hal_set_lock.3hal
hal_signal_delete.3hal
hal_signal_new.3hal
hal_start_threads.3hal
hal_type_t.3hal
hal_u32_t.3hal
hal_unlink.3hal
intro.3hal
undocumented.3hal

14.6.4 Appels à RTAPI

EXPORT_FUNCTION.3rtapi
MODULE_AUTHOR.3rtapi
MODULE_DESCRIPTION.3rtapi
MODULE_LICENSE.3rtapi
RTAPI_MP_ARRAY_INT.3rtapi
RTAPI_MP_ARRAY_LONG.3rtapi
RTAPI_MP_ARRAY_STRING.3rtapi
RTAPI_MP_INT.3rtapi

```
RTAPI_MP_LONG.3rtapi
RTAPI_MP_STRING.3rtapi
intro.3rtapi
rtapi_app_exit.3rtapi
rtapi_app_main.3rtapi
rtapi_clock_set_period.3rtapi
rtapi_delay.3rtapi
rtapi_delay_max.3rtapi
rtapi_exit.3rtapi
rtapi_get_clocks.3rtapi
rtapi_get_msg_level.3rtapi
rtapi_get_time.3rtapi
rtapi_inb.3rtapi
rtapi_init.3rtapi
rtapi_module_param.3rtapi
RTAPI_MP_ARRAY_INT.3rtapi
RTAPI_MP_ARRAY_LONG.3rtapi
RTAPI_MP_ARRAY_STRING.3rtapi
RTAPI_MP_INT.3rtapi
RTAPI_MP_LONG.3rtapi
RTAPI_MP_STRING.3rtapi
rtapi_mutex.3rtapi
rtapi_outb.3rtapi
rtapi_print.3rtap
rtapi_prio.3rtapi
rtapi_prio_highest.3rtapi
rtapi_prio_lowest.3rtapi
rtapi_prio_next_higher.3rtapi
rtapi_prio_next_lower.3rtapi
rtapi_region.3rtapi
rtapi_release_region.3rtapi
rtapi_request_region.3rtapi
rtapi_set_msg_level.3rtapi
rtapi_shmem.3rtapi
rtapi_shmem_delete.3rtapi
rtapi_shmem_getptr.3rtapi
rtapi_shmem_new.3rtapi
rtapi_snprintf.3rtapi
rtapi_task_delete.3rtpi
rtapi_task_new.3rtapi
rtapi_task_pause.3rtapi
rtapi_task_resume.3rtapi
rtapi_task_start.3rtapi
rtapi_task_wait.3rtapi
```

14.7 Les composants temps réel

14.7.1 Stepgen

Ce composant fournit un générateur logiciel d'impulsions de pas répondant aux commandes de position ou de vitesse. En mode position, il contient une boucle de position pré-réglée, de sorte que les réglages de PID ne sont pas nécessaires. En mode vitesse, il pilote un moteur à la vitesse commandée, tout en obéissant aux limites de vitesse et d'accélération. C'est un composant uniquement temps réel, dépendant de plusieurs facteurs comme la vitesse du CPU, etc, il est capable de fournir

des fréquences de pas maximum comprise entre 10kHz et 50kHz. La figure ci-dessous montre trois schémas fonctionnels, chacun est un simple générateur d'impulsions de pas. Le premier diagramme est pour le type 0, (pas et direction). Le second est pour le type 1 (up/down, ou pseudo-PWM) et le troisième est pour les types 2 jusqu'à 14 (les différentes séquences de pas). Les deux premiers diagrammes montrent le mode de commande position et le troisième montre le mode vitesse. Le mode de commande et le type de pas, se règlent indépendamment et n'importe quelle combinaison peut être choisie.

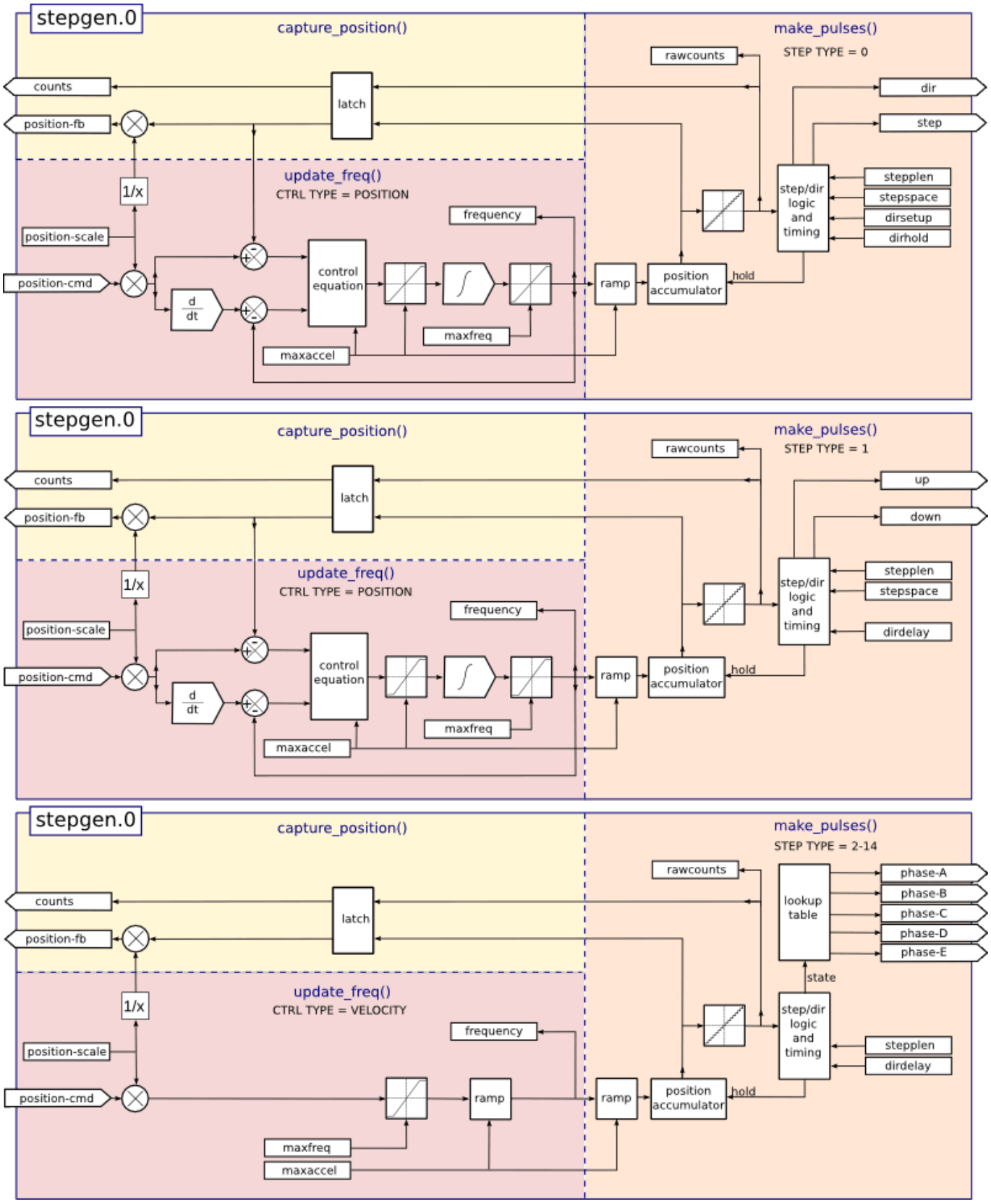


Figure 14.18: Diagramme bloc du générateur de pas stepgen

14.7.1.1 L'installer

```
halcmd: loadrt stepgen step_type=<type-array> [ctrl_type=<ctrl_array>]
```

<type-array> est une série d'entiers décimaux séparés par des virgules. Chaque chiffre provoquera le chargement d'un simple générateur d'impulsions de pas, la valeur de ce chiffre déterminera le type de pas. <ctrl_array> est une série de lettres p ou v séparées par des virgules, qui spécifient le mode pas ou le mode vitesse. ctrl_type est optionnel, si il est omis, tous les générateurs de pas seront en mode position. Par exemple, la commande:

```
halcmd: loadrt stepgen step_type=0,0,2 ctrl_type=p,p,v
```

va installer trois générateurs de pas. Les deux premiers utilisent le type de pas 0 (pas et direction) et fonctionnent en mode position. Le dernier utilise le type de pas 2 (quadrature) et fonctionne en mode vitesse. La valeur par défaut de <config-array> est 0,0,0 qui va installer trois générateurs de type 0 (step/dir). Le nombre maximum de générateurs de pas est de 8 (comme définit par MAX_CHAN dans stepgen.c). Chaque générateur est indépendant, mais tous sont actualisés par la même fonction(s), au même instant. Dans les descriptions qui suivent, <chan> est le nombre de générateurs spécifiques. La numérotation des générateurs commence à 0.

14.7.1.2 Le désinstaller

```
halcmd: unloadrt stepgen
```

14.7.1.3 Pins

Chaque générateur d'impulsions de pas n'aura que certaines de ces pins, selon le type de pas et le mode de contrôle sélectionné.

- (float) stepgen.<chan>.position-cmd — Position désirée du moteur, en unités de longueur (mode position seulement).
- (float) stepgen.<chan>.velocity-cmd — Vitesse désirée du moteur, en unités de longueur par seconde (mode vitesse seulement).
- (s32) stepgen.<chan>.counts — Rétroaction de la position en unités de comptage, actualisée par la fonction capture_position().
- (float) stepgen.<chan>.position-fb — Rétroaction de la position en unités de longueur, actualisée par la fonction capture_position().
- (bit) stepgen.<chan>.step — Sortie des impulsions de pas (type de pas 0 seulement).
- (bit) stepgen.<chan>.dir — Sortie direction (type de pas 0 seulement).
- (bit) stepgen.<chan>.up — Sortie UP en pseudo-PWM (type de pas 1 seulement).
- (bit) stepgen.<chan>.down — Sortie DOWN en pseudo-PWM (type de pas 1 seulement).
- (bit) stepgen.<chan>.phase-A — Sortie phase A (séquences de pas 2 à 14 seulement).
- (bit) stepgen.<chan>.phase-B — Sortie phase B (séquences de pas 2 à 14 seulement).
- (bit) stepgen.<chan>.phase-C — Sortie phase C (séquences de pas 3 à 14 seulement).
- (bit) stepgen.<chan>.phase-D — Sortie phase D (séquences de pas 5 à 14 seulement).
- (bit) stepgen.<chan>.phase-E — Sortie phase E (séquences de pas 11 à 14 seulement).

14.7.1.4 Paramètres

- (float) stepgen.<chan>.position-scale — Pas par unité de longueur. Ce paramètre est utilisé pour les sorties et les rétroactions.
- (float) stepgen.<chan>.maxvel — Vitesse maximale, en unités de longueur par seconde. Si égal à 0.0, n'a aucun effet.
- (float) stepgen.<chan>.maxaccel — Valeur maximale d'accélération, en unités de longueur par seconde au carré. Si égal à 0.0, n'a aucun effet.
- (float) stepgen.<chan>.frequency — Fréquence des pas, en pas par seconde.
- (float) stepgen.<chan>.steplen — Durée de l'impulsion de pas (types de pas 0 et 1) ou durée minimum dans un état donné (séquences de pas 2 à 14), en nanosecondes.
- (float) stepgen.<chan>.stepspace — Espace minimum entre deux impulsions de pas (types de pas 0 et 1 seulement), en nanosecondes.
- (float) stepgen.<chan>.dirsetup — Durée minimale entre un changement de direction et le début de la prochaine impulsion de pas (type de pas 0 seulement), en nanosecondes.
- (float) stepgen.<chan>.dirhold — Durée minimale entre la fin d'une impulsion de pas et un changement de direction (type de pas 0 seulement), en nanosecondes.
- (float) stepgen.<chan>.dirdelay — Durée minimale entre un pas dans une direction et un pas dans la direction opposée (séquences de pas 1 à 14 seulement), en nanosecondes.
- (s32) stepgen.<chan>.rawcounts — Valeur de comptage brute (count) de la rétroaction, réactualisée par la fonction make_pulses().

En mode position, les valeurs de maxvel et de maxaccel sont utilisées par la boucle de position interne pour éviter de générer des trains d'impulsions de pas que le moteur ne peut pas suivre. Lorsqu'elles sont réglées sur des valeurs appropriées pour le moteur, même un grand changement instantané dans la position commandée produira un mouvement trapézoïdal en douceur vers la nouvelle position. L'algorithme fonctionne en mesurant à la fois, l'erreur de position et l'erreur de vitesse, puis en calculant une accélération qui tende à réduire vers zéro, les deux en même temps. Pour plus de détails, y compris les contenus de la boîte d'équation de contrôle, consulter le code source.

En mode vitesse, maxvel est une simple limite qui est appliquée à la vitesse commandée, maxaccel est utilisé pour créer une rampe avec la fréquence actuelle, si la vitesse commandée change brutalement. Comme dans le mode position, des valeurs appropriées de ces paramètres assurent que le moteur pourra suivre le train d'impulsions généré.

14.7.1.5 Séquences de pas

Le générateur de pas supporte 15 différentes séquences de pas. Le type de pas 0 est le plus familier, c'est le standard pas et direction (step/dir). Quand stepgen est configuré pour le type 0, il y a quatre paramètres supplémentaires qui déterminent le timing exact des signaux de pas et de direction. Voir la figure ci-dessous pour la signification de ces paramètres. Les paramètres sont en nanosecondes, mais ils doivent être arrondis à un entier, multiple de la période du thread qui appelle make_pulses(). Par exemple, si make_pulses() est appelée toutes les 16µs et que steplen est à 20000, alors l'impulsion de pas aura une durée de 2 x 16 = 32µs. La valeur par défaut de ces quatre paramètres est de 1ns, mais l'arrondi automatique prendra effet au premier lancement du code. Puisqu'un pas exige d'être haut pendant steplen ns et bas pendant stepspace ns, la fréquence maximale est 1.000.000.000 divisé par (steplen+stepspace). Si maxfreq est réglé plus haut que cette limite, il sera abaissé automatiquement. Si maxfreq est à zéro, il restera à zéro, mais la fréquence de sortie sera toujours limitée.

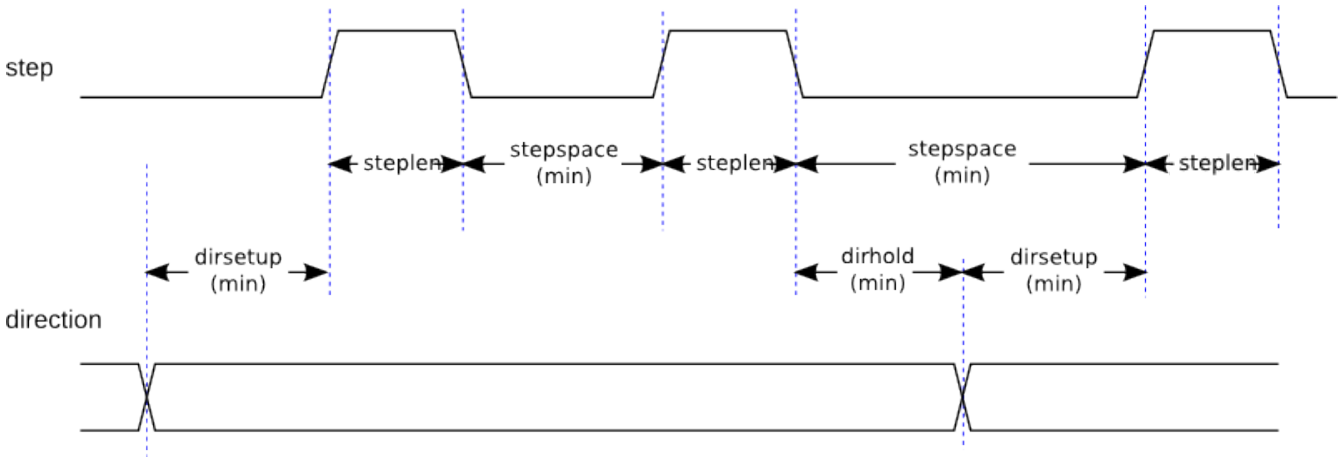


Figure 14.19: Timing pas et direction

Le type de pas 1 a deux sorties, up et down. Les impulsions apparaissent sur l'une ou l'autre, selon la direction du déplacement. Chaque impulsion a une durée de steplen ns et les impulsions sont séparées de stepspace ns. La fréquence maximale est la même que pour le type 0. Si maxfreq est réglé plus haut que cette limite il sera abaissé automatiquement. Si maxfreq est à zéro, il restera à zéro, mais la fréquence de sortie sera toujours limitée.

Les séquences 2 jusqu'à 14 sont basées sur les états et ont entre deux et cinq sorties. Pour chaque pas, un compteur d'état est incrémenté ou décrémenté. Les figures suivantes:

- Trois phases en quadrature,
- Quatre phases,
- Cinq phases

montrent les différentes séquences des sorties en fonction de l'état du compteur. La fréquence maximale est 1.000.000.000 ($1*10^9$) divisé par steplen et comme dans les autres séquences, maxfreq sera abaissé si il est au dessus de cette limite.

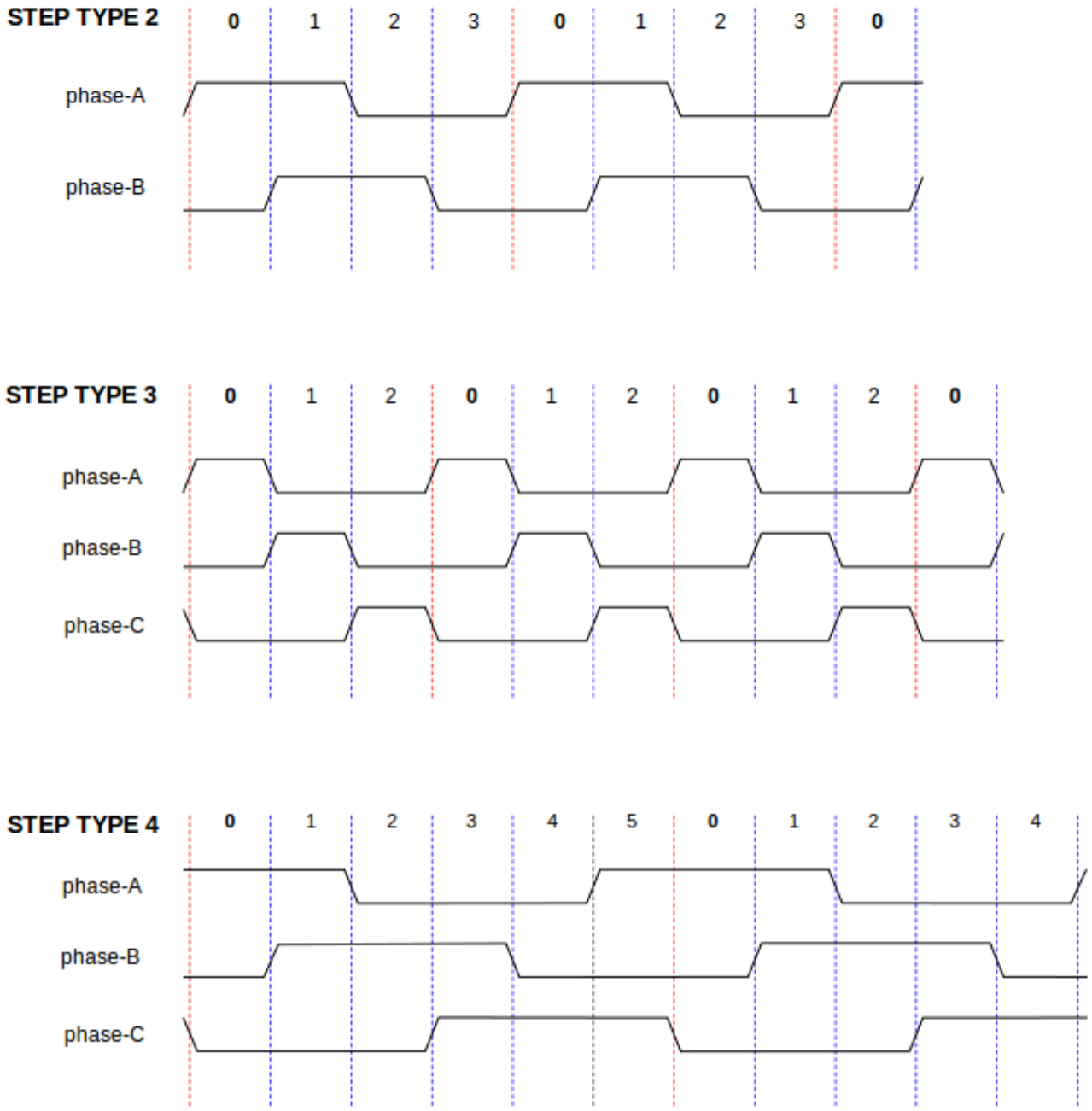


Figure 14.20: Séquences de pas à trois phases

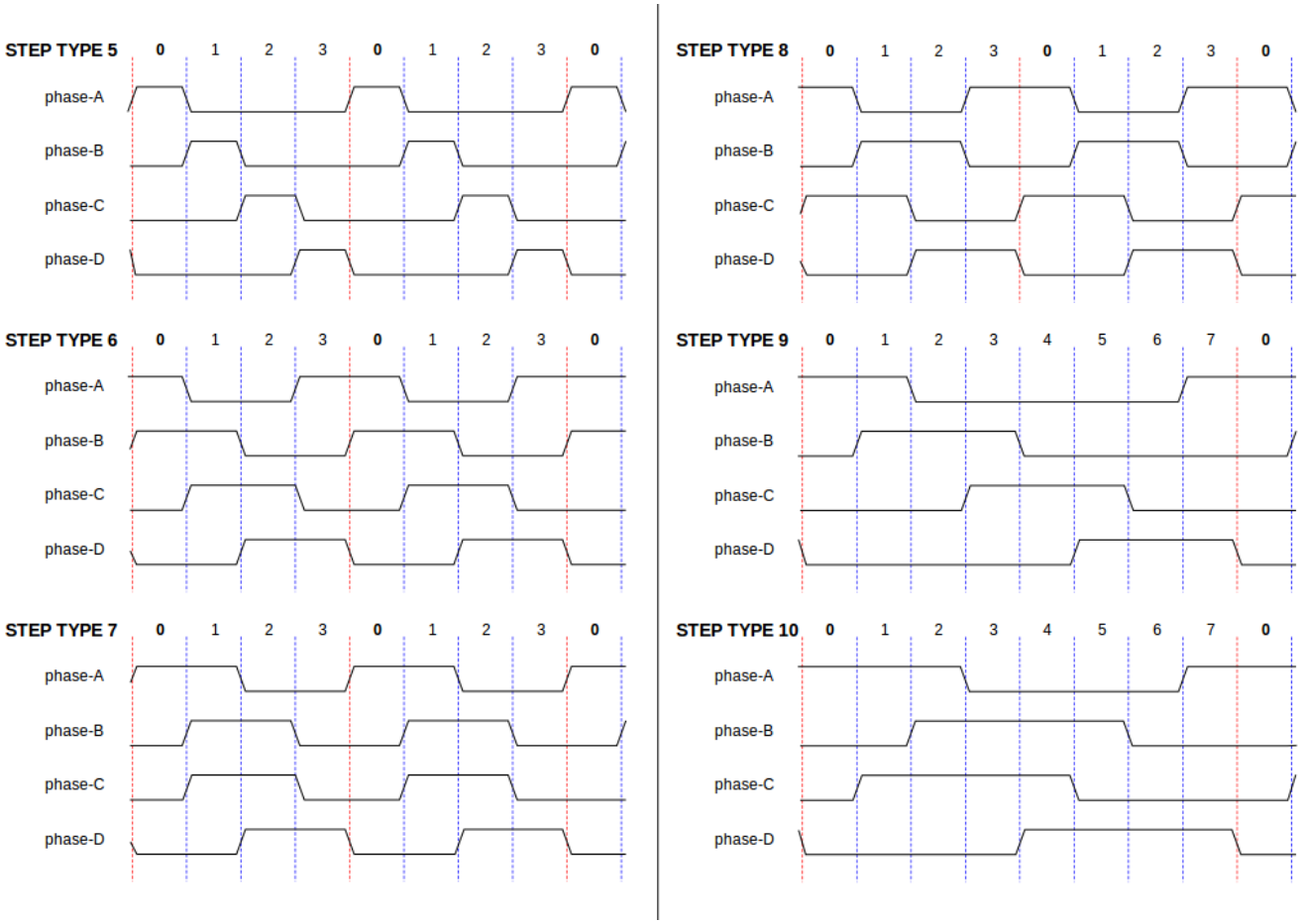


Figure 14.21: Séquences de pas à quatre phases

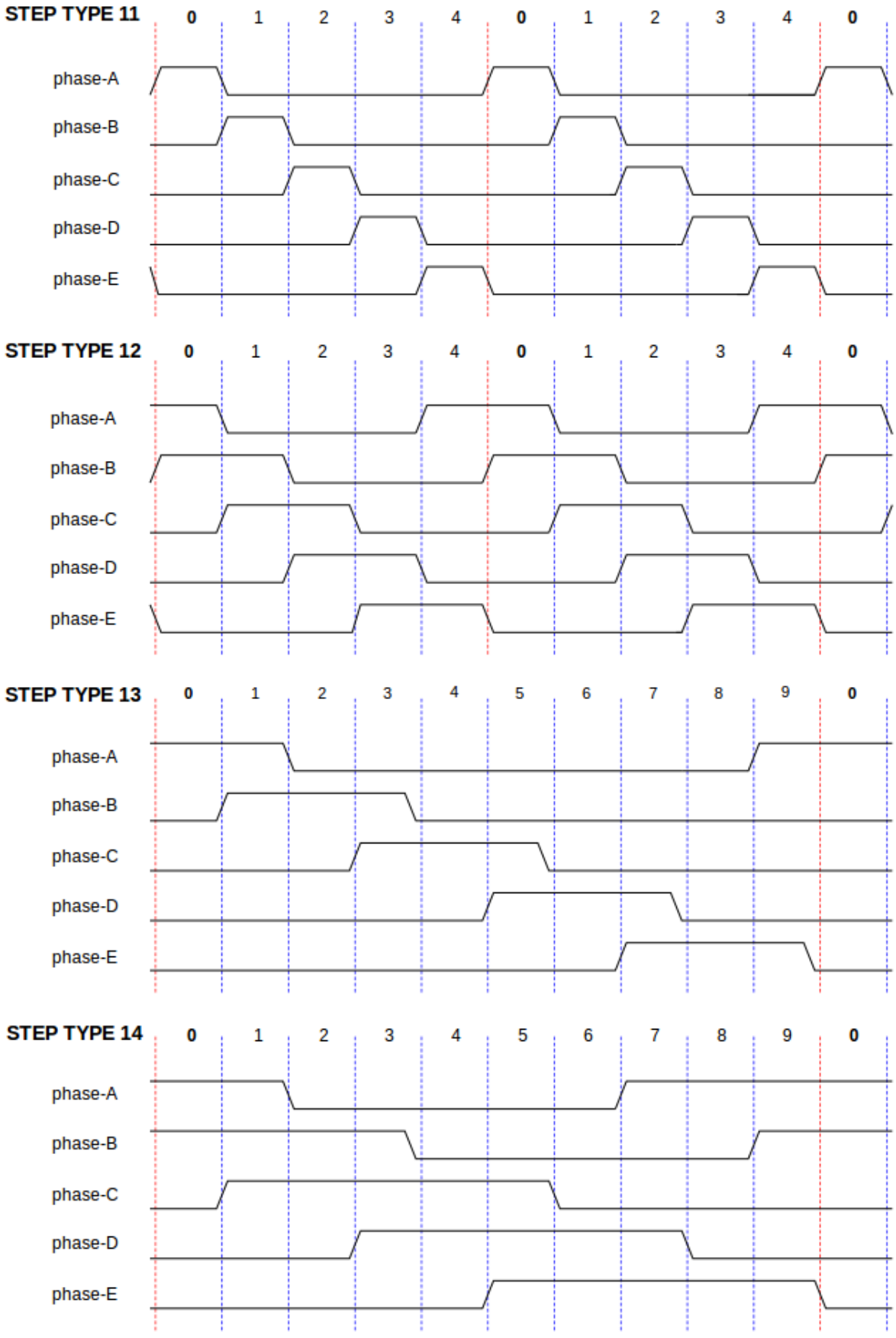


Figure 14.22: Séquence de pas à cinq phases

14.7.1.6 Fonctions

Le composant exporte trois fonctions. Chaque fonction agit sur tous les générateurs d'impulsions de pas. Lancer différents générateurs dans différents threads n'est pas supporté.

- (funct) stepgen.make-pulses — Fonction haute vitesse de génération et de comptage des impulsions (non flottant).
- (funct) stepgen.update-freq — Fonction basse vitesse de conversion de position en vitesse, mise à l'échelle et traitement des limitations.
- (funct) stepgen.capture-position — Fonction basse vitesse pour la rétroaction, met à jour les latches et les mesures de position.

La fonction à grande vitesse stepgen.make-pulses devrait être lancée dans un thread très rapide, entre 10 et 50us selon les capacités de l'ordinateur. C'est la période de ce thread qui détermine la fréquence maximale des pas, de steplen, stepspace, dirsetup, dirhold et dirdelay, tous sont arrondis au multiple entier de la période du thread en nanosecondes. Les deux autres fonctions peuvent être appelées beaucoup plus lentement.

14.7.2 PWMgen

Ce composant fournit un générateur logiciel de PWM (modulation de largeur d'impulsions) et PDM (modulation de densité d'impulsions). C'est un composant temps réel uniquement, dépendant de plusieurs facteurs comme la vitesse du CPU, etc, Il est capable de générer des fréquences PWM de quelques centaines de Hertz en assez bonne résolution, à peut-être 10kHz avec une résolution limitée.

14.7.2.1 L'installer

```
halcmd: loadrt pwmgen output_type=<config-array>
```

<config-array> est une série d'entiers décimaux séparés par des virgules. Chaque chiffre provoquera le chargement d'un simple générateur de PWM, la valeur de ce chiffre determinera le type de sortie.

Exemple avec pwmgen

```
halcmd: loadrt pwmgen output_type=0,1,2
```

va installer trois générateurs de PWM. Le premier utilisera une sortie de type 0 (PWM seule), le suivant utilisera une sortie de type 1 (PWM et direction) et le troisième utilisera une sortie de type 2 (UP et DOWN). Il n'y a pas de valeur par défaut, si <config-array> n'est pas spécifié, aucun générateur de PWM ne sera installé. Le nombre maximum de générateurs de fréquences est de 8 (comme définit par MAX_CHAN dans pwmgen.c). Chaque générateur est indépendant, mais tous sont mis à jour par la même fonction(s), au même instant. Dans les descriptions qui suivent, <chan> est le nombre de générateurs spécifiques. La numérotation des générateurs de PWM commence à 0.

14.7.2.2 Le désinstaller

```
halcmd: unloadrt pwmgen
```

14.7.2.3 Pins

Chaque générateur de PWM aura les pins suivantes:

- (float) pwmgen.<chan>.value — Valeur commandée, en unités arbitraires. Sera mise à l'échelle par le paramètre d'échelle (voir ci-dessous).
- (bit) pwmgen.<chan>.enable — Active ou désactive les sorties du générateur de PWM.

Chaque générateur de PWM aura également certaines de ces pins, selon le type de sortie choisi:

- (bit) pwmgen.<chan>.pwm — Sortie PWM (ou PDM), (types de sortie 0 et 1 seulement).
- (bit) pwmgen.<chan>.dir — Sortie direction (type de sortie 1 seulement).
- (bit) pwmgen.<chan>.up — Sortie PWM/PDM pour une valeur positive en entrée (type de sortie 2 seulement).
- (bit) pwmgen.<chan>.down — Sortie PWM/PDM pour une valeur négative en entrée (type de sortie 2 seulement).

14.7.2.4 Paramètres

- (float) pwmgen.<chan>.scale — Facteur d'échelle pour convertir les valeurs en unités arbitraires, en coefficients de facteur cyclique.
- (float) pwmgen.<chan>.pwm-freq — Fréquence de PWM désirée, en Hz. Si égale à 0.0, la modulation sera PDM au lieu de PWM. Si elle est réglée plus haute que les limites internes, au prochain appel de la fonction update_freq() elle sera ramenée aux limites internes. Si elle est différente de zéro et si le lissage est faux, au prochain appel de la fonction update_freq() elle sera réglée au plus proche entier multiple de la période de la fonction make_pulses().
- (bit) pwmgen.<chan>.dither-pwm — Si vrai, active le lissage pour affiner la fréquence PWM ou le rapport cyclique qui ne pourraient pas être obtenus avec une pure PWM. Si faux, la fréquence PWM et le rapport cyclique seront tous les deux arrondis aux valeurs pouvant être atteintes exactement.
- (float) pwmgen.<chan>.min-dc — Rapport cyclique minimum compris entre 0.0 et 1.0 (Le rapport cyclique sera à zéro quand il est désactivé, indépendamment de ce paramètre).
- (float) pwmgen.<chan>.max-dc — Rapport cyclique maximum compris entre 0.0 et 1.0.
- (float) pwmgen.<chan>.curr-dc — Rapport cyclique courant, après toutes les limitations et les arrondis (lecture seule).

14.7.2.5 Types de sortie

Le générateur de PWM supporte trois types de sortie.

- Le type 0 - A une seule pin de sortie. Seules, les commandes positives sont acceptées, les valeurs négatives sont traitées comme zéro (elle seront affectées par le paramètre min-dc si il est différent de zéro).
- Le type 1 - A deux pins de sortie, une pour le signal PWM/PDM et une pour la direction. Le rapport cyclique d'une pin PWM est basé sur la valeur absolue de la commande, de sorte que les valeurs négatives sont acceptables. La pin de direction est fausse pour les commandes positives et vraie pour les commandes négatives.
- Le type 2 - A également deux sorties, appelées up et down. Pour les commandes positives, le signal PWM apparaît sur la sortie up et la sortie down reste fausse. Pour les commandes négatives, le signal PWM apparaît sur la sortie down et la sortie up reste fausse. Les sorties de type 2 sont appropriées pour piloter la plupart des ponts en H.

14.7.2.6 Fonctions

Le composant exporte deux fonctions. Chaque fonction agit sur tous les générateurs de PWM, lancer différents générateurs dans différents threads n'est pas supporté.

- (funct) pwmgen.make-pulses — Fonction haute vitesse de génération de fréquences PWM (non flottant).
- (funct) pwmgen.update — Fonction basse vitesse de mise à l'échelle, limitation des valeurs et traitement d'autres paramètres.

La fonction haute vitesse pwmgen.make-pulses devrait être lancée dans un thread très rapide, entre 10 et 50 us selon les capacités de l'ordinateur. C'est la période de ce thread qui détermine la fréquence maximale de la porteuse PWM, ainsi que la résolution des signaux PWM ou PDM. L'autre fonction peut être appelée beaucoup plus lentement.

14.7.3 Codeur

Ce composant fournit, en logiciel, le comptage des signaux provenant d'encodeurs en quadrature. Il s'agit d'un composant temps réel uniquement, il est dépendant de divers facteurs comme la vitesse du CPU, etc, il est capable de compter des signaux de fréquences comprises entre 10kHz à peut être 50kHz. La figure ci-dessous représente le diagramme bloc d'une voie de comptage de codeur.

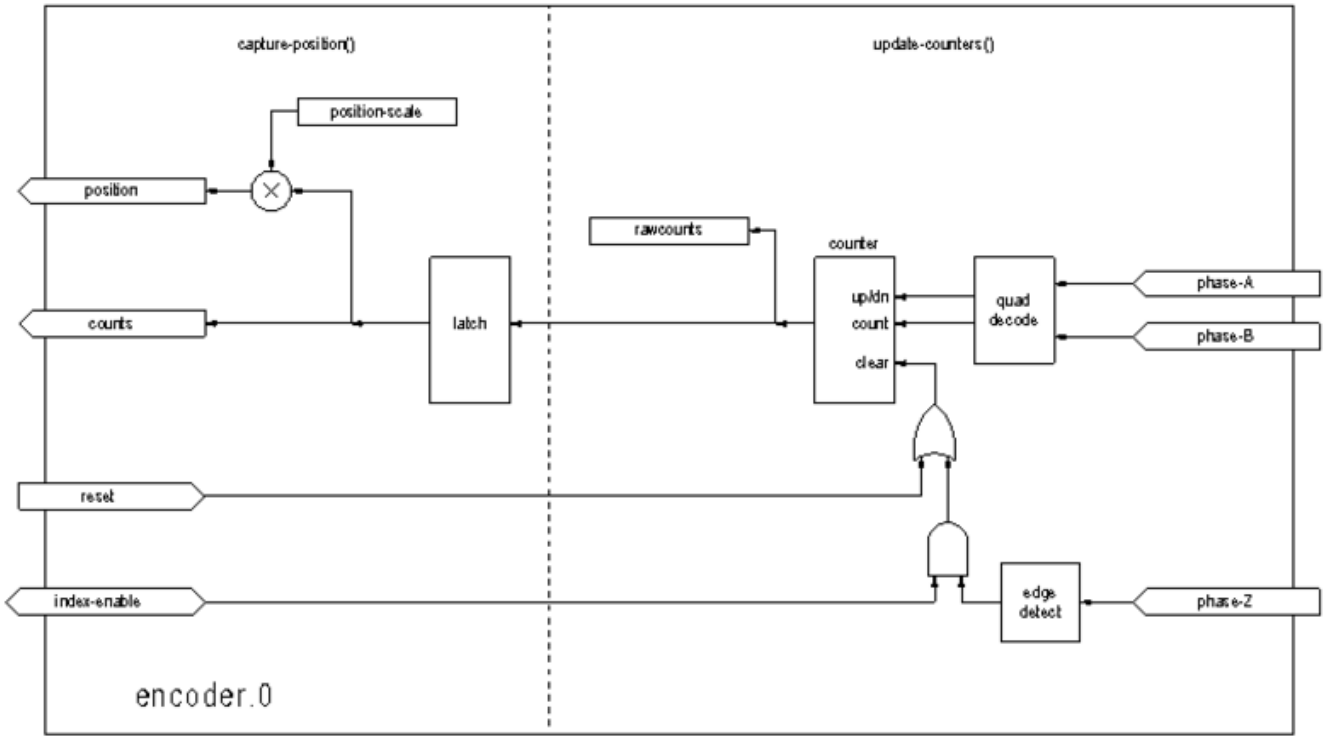


Figure 14.23: Diagramme bloc du codeur

14.7.3.1 L'installer

```
halcmd: loadrt encoder [num_chan=<counters>]
```

<counters> est le nombre de compteurs de codeur à installer. Si numchan n'est pas spécifié, trois compteurs seront installés. Le nombre maximum de compteurs est de 8 (comme définit par MAX_CHAN dans encoder.c). Chaque compteur est indépendant, mais tous sont mis à jour par la même fonction(s) au même instant. Dans les descriptions qui suivent, <chan> est le nombre de compteurs spécifiques. La numérotation des compteurs commence à 0.

14.7.3.2 Le désinstaller

```
halcmd: unloadrt encoder
```

14.7.3.3 Pins

- Encodeur <chan> counter-mode (bit, I/O) (par défaut: FALSE) — Permet le mode compteur. Lorsque TRUE, le compteur compte chaque front montant de l'entrée phase-A, ignorant la valeur de la phase-B. Ceci est utile pour compter la sortie d'un capteur simple canal (pas de quadrature). Si FALSE, il compte en mode quadrature.
- encoder.<chan>.counts (s32, Out) — Position en comptage du codeur.
- encoder.<chan>.counts-latched (s32, Out) — Non utilisé à ce moment.
- encoder.<chan> index-enable (bit, I/O) — Si TRUE, counts et position sont remis à zéro au prochain front montant de la phase Z. En même temps, index-enable est remis à zéro pour indiquer que le front montant est survenu. La broche index-enable est bi-directionnelle. Si index-enable est FALSE, la phase Z du codeur sera ignorée et le compteur comptera normalement. Le pilote du codeur ne doit jamais mettre index-enable TRUE. Cependant, d'autres composants peuvent le faire.
- encoder.<chan>.latch-falling (bit, In) (par défaut: TRUE) — Non utilisé à ce moment.
- encoder.<chan>.latch-input (bit, In) (par défaut: TRUE) — Non utilisé à ce moment.
- encoder.<chan>.latch-rising (bit, In) — Non utilisé à ce moment.
- encoder.<chan>.min-speed-estimate (Float, In) — Effectue une estimation de la vitesse minimale réelle, à partir de laquelle, la vitesse sera estimée comme non nulle et la position interpolées, comme étant interpolée. Les unités de vitesse min-speed-estimate sont les mêmes que les unités de velocity. Le facteur d'échelle, en compte par unité de longueur. Régler ce paramètre trop bas, fera prendre beaucoup de temps pour que la vitesse arrive à 0 après que les impulsions du codeur aient cessé d'arriver.
- encoder.<chan>.phase-A (bit, In) — Signal de la phase A du codeur en quadrature.
- encoder.<chan>.phase-B (bit, In) — Signal de la phase B du codeur en quadrature.
- encoder.<chan>.phase-Z (bit, In) — Signal de la phase Z (impulsion d'index) du codeur en quadrature.
- encoder.<chan>.position (float, Out) - Position en unités mises à l'échelle (voir position échelle).
- encoder.<chan>.position-interpolated (float, Out) - Position en unités mises à l'échelle, interpolées entre les comptes du codeur. position-interpolated tente d'interpoler entre les comptes du codeur, basée sur la mesure de vitesse la plus récente. Valable uniquement lorsque la vitesse est approximativement constante et supérieure à min-speed-estimate. Ne pas utiliser pour le contrôle de position, puisque sa valeur est incorrecte en basse vitesse, lors des inversions de direction et pendant les changements de vitesse. Toutefois, il permet à un codeur à PPR faible (y compris les codeur à une impulsion par tour) d'être utilisé pour du filetage sur tour et peut aussi avoir d'autres usages.
- encoder.<chan>.position-latched (float, Out) — Non utilisé à ce moment.
- encoder.<chan>.position-scale (float, I/O) — Le facteur d'échelle, en comptes par unité de longueur. Par exemple, si position-scale est à 500, alors à 1000 comptes codeur, la position sera donnée à 2,0 unités.
- encoder.<chan>.rawcounts (s32, In) — Le compte brut, tel que déterminé par _update-counters. Cette valeur est mise à jour plus fréquemment que compte et position. Il n'est également pas affecté par le reset ou l'impulsion d'index.
- encoder.<chan>.reset (bit, In) — Si TRUE, force counts et position immédiatement à zéro.
- encoder.<chan>.velocity (float, Out) — Vitesse en unités mises à l'échelle par secondes. encoder utilise un algorithme qui réduit considérablement la quantification du bruit comparé à simplement différencier la sortie position. Lorsque la magnitude de la vitesse réelle est inférieure à min-speed-estimate, la sortie velocity est à 0.

- encoder.<chan>.x4-mode (bit, I/O) (par défaut: TRUE) — Permet le mode x4. Lorsqu'il est TRUE, le compteur compte chaque front de l'onde en quadrature (quatre compte par cycle complet). Si FALSE, il ne compte qu'une seule fois par cycle complet. En mode compteur, ce paramètre est ignoré. Le mode 1x est utile pour certaines manivelles électroniques.

14.7.3.4 Paramètres

- encoder.<chan>.capture-position.time (s32, RO)
- encoder.<chan>.capture-position.tmax (s32, RW)
- encoder.<chan>.update-counters.time (s32, RO)
- encoder.<chan>.update-counter.tmax (s32, RW)

14.7.3.5 Fonctions

Le composant exporte deux fonctions. Chaque fonction agit sur tous les compteurs de codeur, lancer différents compteurs de codeur dans différents threads n'est pas supporté.

- (funct) encoder.update-counters — Fonction haute vitesse de comptage d'impulsions (non flottant).
- (funct) encoder.capture-position — Fonction basse vitesse d'actualisation des latches et mise à l'échelle de la position.

14.7.4 PID

Ce composant fournit une boucle de contrôle Proportionnelle/Intégrale/Dérivée. C'est un composant temps réel uniquement. Par souci de simplicité, cette discussion suppose que nous parlons de boucles de position, mais ce composant peut aussi être utilisé pour implémenter d'autres boucles de rétroaction telles que vitesse, hauteur de torche, température, etc. La figure ci-dessous est le schéma fonctionnel d'une simple boucle PID.

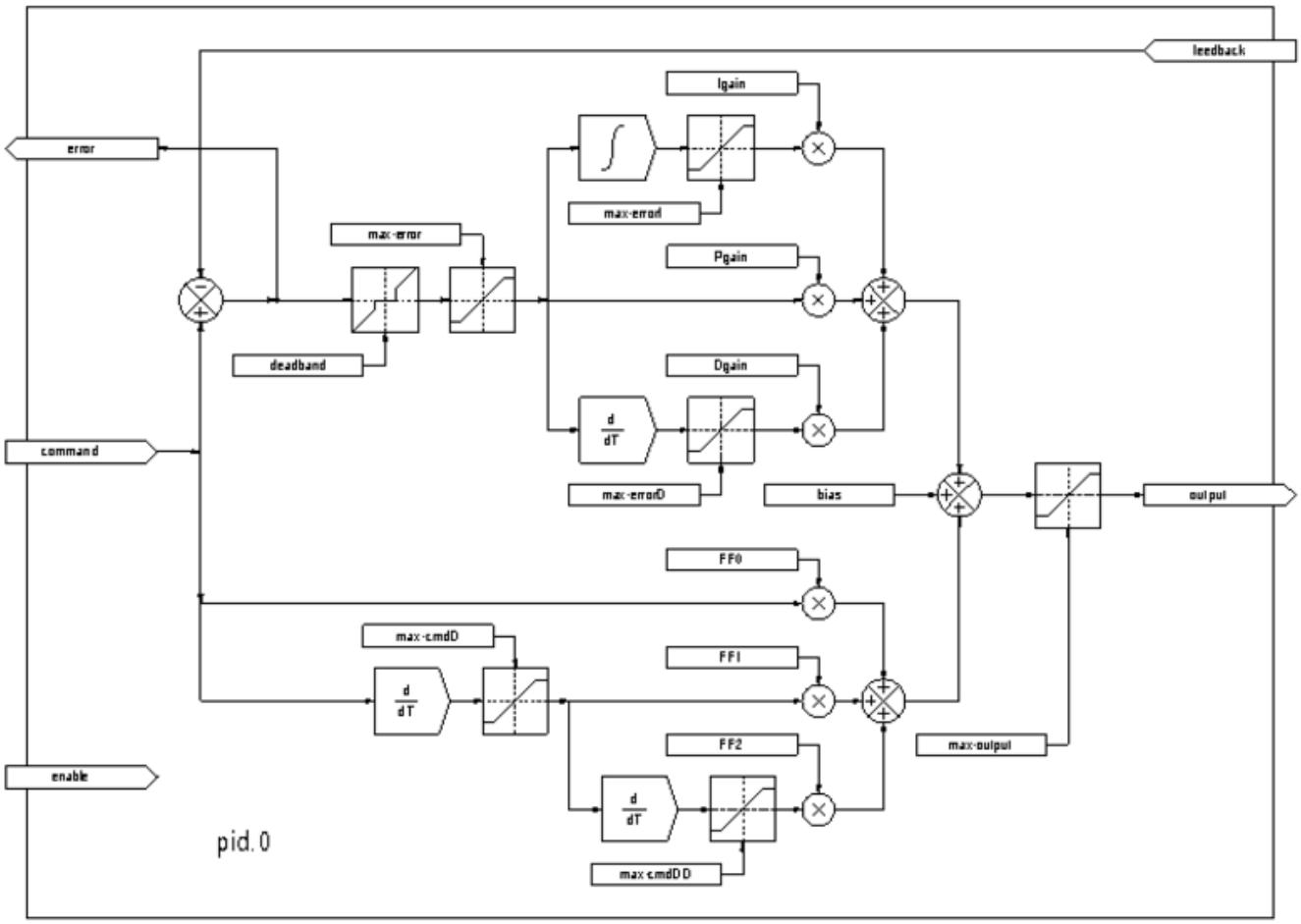


Figure 14.24: Diagramme bloc d’une boucle PID

14.7.4.1 L’installer

```
halcmd: loadrt pid [num_chan=<loops>] [debug=1]
```

<loops> est le nombre de boucles PID à installer. Si numchan n’est pas spécifié, une seule boucle sera installée. Le nombre maximum de boucles est de 16 (comme définit par MAX_CHAN dans pid.c). Chaque boucle est complétement indépendante. Dans les descriptions qui suivent, <loopnum> est le nombre de boucles spécifiques. La numérotation des boucle PID commence à 0.

Si debug=1 est spécifié, le composant exporte quelques paramètres destinés au débogage et aux réglages. Par défaut, ces paramètres ne sont pas exportés, pour économiser la mémoire partagée et éviter d’encombrer la liste des paramètres.

14.7.4.2 Le désinstaller

```
halcmd: unloadrt pid
```

14.7.4.3 Pins

Les trois principales pins sont:

- (float) pid.<loopnum>.command — La position désirée (consigne), telle que commandée par un autre composant système.
- (float) pid.<loopnum>.feedback — La position actuelle (mesure), telle que mesurée par un organe de rétroaction comme un codeur de position.
- (float) pid.<loopnum>.output — Une commande de vitesse qui tend à aller de la position actuelle à la position désirée.

Pour une boucle de position, command et feedback sont en unités de longueur. Pour un axe linéaire, cela pourrait être des pouces, mm, mètres, ou tout autre unité pertinente. De même pour un axe angulaire, ils pourraient être des degrés, radians, etc. Les unités sur la pin output représentent l'écart nécessaire pour que la rétroaction coïncide avec la commande. Pour une boucle de position, output est une vitesse exprimée en pouces/seconde, mm/seconde, degrés/seconde, etc. Les unités de temps sont toujours des secondes et les unités de vitesses restent cohérentes avec les unités de longueur. Si la commande et la rétroaction sont en mètres, la sortie sera en mètres par seconde.

Chaque boucle PID a deux autres pins qui sont utilisées pour surveiller ou contrôler le fonctionnement général du composant.

- (float) pid.<loopnum>.error — Egal à .command moins .feedback. (consigne - mesure)
- (bit) pid.<loopnum>.enable — Un bit qui active la boucle. Si .enable est faux, tous les intégrateurs sont remis à zéro et les sorties sont forcées à zéro. Si .enable est vrai, la boucle opère normalement.

Pins utilisé pour signaler la saturation. La saturation se produit lorsque la sortie de le bloc PID est à son maximum ou limiter au minimum.

- (bit) pid.<loopnum>.saturated — True lorsque la sortie est saturée.
- (float) pid.<loopnum>.saturated_s — Le temps de la sortie a été saturé.
- (s32) pid.<loopnum>.saturated_count — Le temps de la sortie a été saturé.

14.7.4.4 Paramètres

Le gain PID, les limites et autres caractéristiques accordables de la boucle sont implémentés comme des paramètres.

- (float) pid.<loopnum>.Pgain — Gain de la composante proportionnelle.
- (float) pid.<loopnum>.Igain — Gain de la composante intégrale.
- (float) pid.<loopnum>.Dgain — Gain de la composante dérivée.
- (float) pid.<loopnum>.bias — Constante du décalage de sortie.
- (float) pid.<loopnum>.FF0 — Correcteur prédictif d'ordre zéro (retour vitesse) sortie proportionnelle à la commande (position).
- (float) pid.<loopnum>.FF1 — Correcteur prédictif de premier ordre (retour vitesse) sortie proportionnelle à la dérivée de la commande (vitesse).
- (float) pid.<loopnum>.FF2 — Correcteur prédictif de second ordre (retour vitesse) sortie proportionnelle à la dérivée seconde de la commande (accélération). [17]

[17] FF2 n'est actuellement pas implémenté, mais il pourrait l'être. Considérez cette note comme un "FIXME" dans le code.

- (float) pid.<loopnum>.deadband — Définit la bande morte tolérable.
- (float) pid.<loopnum>.maxerror — Limite d'erreur.
- (float) pid.<loopnum>.maxerrorI — Limite d'erreur intégrale.
- (float) pid.<loopnum>.maxerrorD — Limite d'erreur dérivée.
- (float) pid.<loopnum>.maxcmdD — Limite de la commande dérivée.
- (float) pid.<loopnum>.maxcmdDD — Limite de la commande dérivée seconde.
- (float) pid.<loopnum>.maxoutput — Limite de la valeur de sortie.

Toutes les limites max???, sont implémentées de sorte que si la valeur de ce paramètre est à zéro, il n'y a pas de limite.

Si debug=1 est spécifié quand le composant est installé, quatre paramètres supplémentaires seront exportés:

- (float) pid.<loopnum>.errorI — Intégrale de l'erreur.
- (float) pid.<loopnum>.errorD — Dérivée de l'erreur.
- (float) pid.<loopnum>.commandD — Dérivée de la commande.
- (float) pid.<loopnum>.commandDD — Dérivée seconde de la commande.

14.7.4.5 Fonctions

Le composant exporte une fonction pour chaque boucle PID. Cette fonction exécute tous les calculs nécessaires à la boucle. Puisque chaque boucle a sa propre fonction, les différentes boucles peuvent être incluses dans les différents threads et exécutées à différents rythmes.

- (funct) pid.<loopnum>.do_pid_calcs — Exécute tous les calculs d'une seule boucle PID.

Si vous voulez comprendre exactement l'algorithme utilisé pour calculer la sortie d'une boucle PID, référez vous à la figure PID, les commentaires au début du source linuxcnc/src/hal/components/pid.c et bien sûr, au code lui même. Les calculs de boucle sont dans la fonction C calc_pid().

14.7.5 Codeur simulé

Le codeur simulé est exactement la même chose qu'un codeur. Il produit des impulsions en quadrature avec une impulsion d'index, à une vitesse contrôlée par une pin de HAL. Surtout utile pour les essais.

14.7.5.1 L'installer

```
halcmd: loadrt sim-encoder num_chan=<number>
```

<number> est le nombre de canaux à simuler. Si rien n'est spécifié, un seul canal sera installé. Le nombre maximum de canaux est de 8 (comme défini par MAX_CHAN dans sim_encoder.c).

14.7.5.2 Le désinstaller

```
halcmd: unloadrt sim-encoder
```

14.7.5.3 Pins

- (float) sim-encoder.<chan-num>.speed — La vitesse commandée pour l'arbre simulé.
- (bit) sim-encoder.<chan-num>.phase-A — Sortie en quadrature.
- (bit) sim-encoder.<chan-num>.phase-B — Sortie en quadrature.
- (bit) sim-encoder.<chan-num>.phase-Z — Sortie de l'impulsion d'index.

Quand .speed est positive, .phase-A mène .phase-B.

14.7.5.4 Paramètres

- (u32) sim-encoder.<chan-num>.ppr — Impulsions par tour d'arbre.
- (float) sim-encoder.<chan-num>.scale — Facteur d'échelle pour speed. Par défaut est de 1.0, ce qui signifie que speed est en tours par seconde. Passer l'échelle à 60 pour des tours par minute, la passer à 360 pour des degrés par seconde, à 6.283185 pour des radians par seconde, etc.

Noter que les impulsions par tour ne sont pas identiques aux valeurs de comptage par tour (counts). Une impulsion est un cycle complet de quadrature. La plupart des codeurs comptent quatre fois pendant un cycle complet.

14.7.5.5 Fonctions

Le composant exporte deux fonctions. Chaque fonction affecte tous les codeurs simulés.

- (funct) sim-encoder.make-pulses — Fonction haute vitesse de génération d'impulsions en quadrature (non flottant).
- (funct) sim-encoder.update-speed — Fonction basse vitesse de lecture de speed, de mise à l'échelle et d'activation de make-pulses.

14.7.6 Anti-rebond

L'anti-rebond est un composant temps réel capable de filtrer les rebonds créés par les contacts mécaniques. Il est également très utile dans d'autres applications, où des impulsions très courtes doivent être supprimées.

14.7.6.1 L'installer

```
halcmd: loadrt debounce cfg=<config-string>
```

<config-string> est une série d'entiers décimaux séparés par des espaces. Chaque chiffre installe un groupe de filtres anti-rebond identiques, le chiffre détermine le nombre de filtres dans le groupe. Par exemple:

```
halcmd: loadrt debounce cfg=1,4,2
```

va installer trois groupes de filtres. Le groupe 0 contient un filtre, le groupe 1 en contient quatre et le groupe 2 en contient deux. La valeur par défaut de <config-string> est 1 qui installe un seul groupe contenant un seul filtre. Le nombre maximum de groupes est de 8 (comme définit par MAX_GROUPS dans debounce.c). Le nombre maximum de filtres dans un groupe est limité seulement par l'espace de la mémoire partagée. Chaque groupe est complétement indépendant. Tous les filtres dans un même groupe sont identiques et ils sont tous mis à jour par la même fonction, au même instant. Dans les descriptions qui suivent, <G> est le numéro du groupe et <F> est le numéro du filtre dans le groupe. Le premier filtre est le filtre 0 dans le groupe 0.

14.7.6.2 Le désinstaller

```
halcmd: unloadrt debounce
```

14.7.6.3 Pins

Chaque filtre individuel a deux pins.

- (bit) debounce.<G>.<F>.in — Entrée du filtre <F> du groupe <G>.
- (bit) debounce.<G>.<F>.out — Sortie du filtre <F> du groupe <G>.

14.7.6.4 Paramètres

Chaque groupe de filtre a un paramètre. [18]

- (s32) debounce.<G>.delay — Délai de filtrage pour tous les filtres du groupe <G>.

Le délai du filtre est dans l'unité de la période du thread. Le délai minimum est de zéro. La sortie d'un filtre avec un délai de zéro, suit exactement son entrée, il ne filtre rien. Plus le délai augmente, plus larges seront les impulsions rejetées. Si le délai est de 4, toutes les impulsions égales ou inférieures à quatre périodes du thread, seront rejetées.

14.7.6.5 Fonctions

Chaque groupe de filtres exporte une fonction qui met à jour tous les filtres de ce groupe simultanément. Différents groupes de filtres peuvent être mis à jour dans différents threads et à différentes périodes.

- (funct) debounce.<G> — Met à jour tous les filtres du groupe <G>.

14.7.7 Siggen

Siggen est un composant temps réel qui génère des signaux carrés, triangulaires et sinusoïdaux. Il est principalement utilisé pour les essais.

14.7.7.1 L'installer

```
halcmd: loadrt siggen [num_chan=<chans>]
```

<chans> est le nombre de générateurs de signaux à installer. Si numchan n'est pas spécifié, un seul générateur de signaux sera installé. Le nombre maximum de générateurs est de 16 (comme définit par MAX_CHAN dans siggen.c). Chaque générateur est complétement indépendant. Dans les descriptions qui suivent, <chan> est le numéro d'un générateur spécifique. Les numéros de générateur commencent à 0.

[18] Chaque filtre individuel a également une variable d'état interne. C'est un switch du compilateur qui peut exporter cette variable comme un paramètre. Ceci est prévu pour des essais et devrait juste être un gaspillage de mémoire partagée dans des circonstances normales.

14.7.7.2 Le désinstaller

```
halcmd: unloadrt siggen
```

14.7.7.3 Pins

Chaque générateur a cinq pins de sortie.

- (float) siggen.<chan>.sine — Sortie de l'onde sinusoïdale.
- (float) siggen.<chan>.cosine — Sortie de l'onde cosinusoïdale.
- (float) siggen.<chan>.sawtooth — Sortie de l'onde en dents de scie.
- (float) siggen.<chan>.triangle — Sortie de l'onde triangulaire.
- (float) siggen.<chan>.square — Sortie de l'onde carrée.

Les cinq sorties ont les mêmes fréquence, amplitude et offset.

Trois pins de contrôle s'ajoutent aux pins de sortie:

- (float) siggen.<chan>.frequency — Réglage de la fréquence en Hertz, par défaut la valeur est de 1 Hz.
- (float) siggen.<chan>.amplitude — Réglage de l'amplitude de pic des signaux de sortie, par défaut, est à 1.
- (float) siggen.<chan>.offset — Réglage de la composante continue des signaux de sortie, par défaut, est à 0.

Par exemple, si siggen.0.amplitude est à 1.0 et siggen.0.offset est à 0.0, les sorties oscilleront entre -1.0 et +1.0. Si siggen.0.amplitude est à 2.5 et siggen.0.offset est à 10.0, les sorties oscilleront entre 7.5 et 12.5.

14.7.7.4 Paramètres

Aucun. [19]

14.7.7.5 Fonctions

- (funct) siggen.<chan>.update — Calcule les nouvelles valeurs pour les cinq sorties.

14.7.8 lut5

Le composant lut5 est un composant de logique à 5 entrées basé sur une table de vérité.

- lut5 ne requiert pas un thread à virgule flottante.

Installation

[19] Dans les versions antérieures à la 2.1, fréquence, amplitude et offset étaient des paramètres. Ils ont été modifiés en pins pour permettre le contrôle par d'autres composants.

```
loadrt lut5 [count=N|names=name1[,name2...]]
addf lut5.N servo-thread | base-thread
setp lut5.N.function 0xN
```

Calcul de la valeur de la fonction Pour calculer la valeur hexadécimale de la fonction, démarrer par le haut et entrer un 1 où un 0 pour indiquer si cette colonne devra être vraie où fausse. Ensuite écrire les valeurs en dessous, d'abord dans la colonne de sortie en commençant par le haut puis en écrivant les valeurs correspondantes de la droite vers la gauche. Le nombre binaire sera celui contenu dans la colonne de sortie. Utiliser une calculette comme celle fournie sous Ubuntu, entrer ce nombre binaire et le convertir en hexadécimal pour obtenir la valeur pour la fonction.

Table 14.1: Table de vérité

Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Output
0	0	0	0	0	
0	0	0	0	1	
0	0	0	1	0	
0	0	0	1	1	
0	0	1	0	0	
0	0	1	0	1	
0	0	1	1	0	
0	0	1	1	1	
0	1	0	0	0	
0	1	0	0	1	
0	1	0	1	0	
0	1	0	1	1	
0	1	1	0	0	
0	1	1	0	1	
0	1	1	1	0	
0	1	1	1	1	
1	0	0	0	0	
1	0	0	0	1	
1	0	0	1	0	
1	0	0	1	1	
1	0	1	0	0	
1	0	1	0	1	
1	0	1	1	0	
1	0	1	1	1	
1	1	0	0	0	
1	1	0	0	1	
1	1	0	1	0	
1	1	0	1	1	
1	1	1	0	0	
1	1	1	0	1	
1	1	1	1	0	
1	1	1	1	1	

Un exemple de fonction. Dans la table suivante nous avons sélectionné l'état de sortie pour chaque ligne que nous souhaitons vraie.

Table 14.2: Table de vérité

Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Output
0	0	0	0	0	0
0	0	0	0	1	1
0	0	0	1	0	0
0	0	0	1	1	1

En regardant la colonne de sortie de notre exemple, nous voulons que la sortie soit active quand le bit 0 OU le bit 0 ET le bit 1 soient actifs et rien d'autre. Le nombre binaire est b1010 (rotation de la sortie de 90° en sens horaire). Entrer ce nombre dans une calculette, le convertir en hexadécimal et le nombre demandé pour cette fonction est 0xa. Le préfixe 0x étant celui des nombres hexadécimaux.

14.8 Exemples pour HAL

Tous ces exemples s'appuient sur une configuration créée par Stepconf, elle a deux threads, base-thread et servo-thread.

L'assistant de configuration Stepconf aura créé le fichier vide custom.hal et le fichier custom_postgui.hal.

Le fichier custom.hal sera chargé après le fichier de configuration de HAL, le fichier custom_postgui.hal sera chargé après que l'interface graphique ne le soit.

14.8.1 Changement d'outil manuel

Dans cet exemple, il est supposé que la configuration a été réalisée et qu'il faut lui ajouter la fenêtre de changement d'outil de HAL. Le composant de changement d'outil manuel de HAL est surtout intéressant si les outils sont mesurables avec précision en longueur et que les offsets sont stockés dans la table d'outils. Si il est nécessaire de faire un Toucher pour chaque outil, il sera préférable de scinder le programme G-code en plusieurs tronçons. Pour utiliser la fenêtre de changement manuel d'outil de HAL, il faut d'abord charger le composant hal_manualtoolchange puis envoyer l'ordre iocontrol tool change vers le change de hal_manualtoolchange ainsi qu'envoyer le changed de hal_manualtoolchange en retour sur le iocontrol tool changed.

Voici un exemple avec utilisation du composant de HAL, pour clarifier tout cela:

```
loadusr -W hal_manualtoolchange
net tool-change iocontrol.0.tool-change => hal_manualtoolchange.change
net tool-changed iocontrol.0.tool-changed <= hal_manualtoolchange.changed
net tool-number iocontrol.0.tool-prep-number => hal_manualtoolchange.number
net tool-prepare-loopback iocontrol.0.tool-prepare => iocontrol.0.tool-prepared
```

Et voici un exemple sans le composant de changement manuel:

```
net tool-number <= iocontrol.0.tool-prep-number
net tool-change-loopback iocontrol.0.tool.-change => iocontrol.0.tool-changed
net tool-prepare-loopback iocontrol.0.tool-prepare => iocontrol.0.tool-prepared
```

14.8.2 Calcul de vitesse

Cet exemple utilise ddt, mult2 et abs pour calculer la vitesse de déplacement sur un axe. Pour plus de détails, voir la section sur les composants de HAL.

En premier il convient de vérifier si la configuration contient déjà des composants temps réel. Il est possible de le vérifier en ouvrant la fenêtre de Halshow et en cherchant les composants dans la section des pins. Si il y en a déjà en activité, il faudra augmenter leur nombre et ajuster l'instance de ces composants à la valeur correcte. Ajouter le code suivant dans le fichier custom.hal.

Charger les composants temps réel.

```
loadrt ddt count=1
loadrt mult2 count=1
loadrt abs count=1
```

Ajouter les fonctions au thread pour qu'elles soient rafraîchies.

```
addf ddt.0 servo-thread
addf mult2.0 servo-thread
addf abs.0 servo-thread
```

Faire les connections.

```
setp mult2.in1 60
net xpos-cmd ddt.0.in
net X-IPS mult2.0.in0 <= ddt.0.out
net X-ABS abs.0.in <= mult2.0.out
net X-IPM abs.0.out
```

Dans la dernière section, nous avons fixé le mult2.0.in1 à 60 pour convertir les unités par seconde en unités par minute (dans cet exemple, des pouces/mn), nous l'obtenons sur la sortie ddt.0.out.

La commande xpos-cmd envoie la position commandée à l'entrée ddt.0.in. Le ddt calcule la dérivée de la variation du signal sur son entrée.

La sortie ddt2.0.out est multipliée par 60 pour obtenir des unités par minute.

La sortie mult2.0.out est envoyée au composant abs pour obtenir la valeur absolue.

La figure suivante montre le résultat quand l'axe X se déplace à 15 unités/mn dans la direction négative. Noter que la valeur absolue peut être prise sur la pin abs.0.out ou le signal X-IPM.

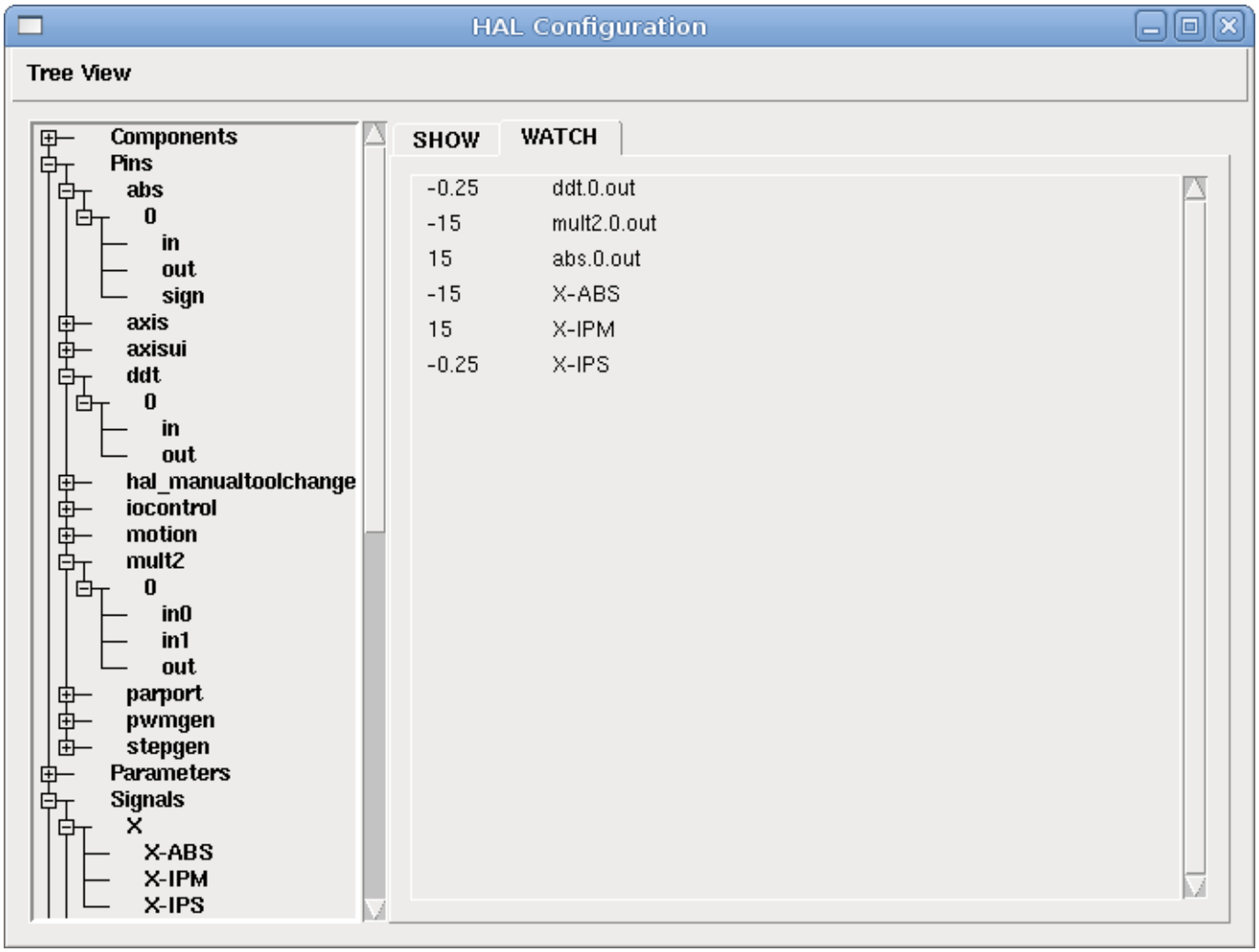


Figure 14.25: Exemple avec la vitesse

14.8.3 Amortissement d'un signal

Cette exemple montre comment les composants de HAL lowpass, limit2 ou limit3 peuvent être utilisés pour amortir de brusques changements d'un signal.

Nous sommes sur un tour dont la broche est pilotée par un servomoteur. Si nous envoyions directement la consigne de vitesse de broche sur le servo, celui-ci chercherait, à partir de la vitesse courante, à atteindre la vitesse commandée le plus vite possible. Cette situation est problématique et peut détériorer le matériel. Pour amortir ce changement de vitesse, nous pouvons faire passer la sortie spindle.N.speed-out à travers un limiteur avant d'aller au PID, de sorte que la valeur de commande du PID soit amortie.

Les trois composants intégrés pour amortir le signal seront:

limit2

- Limite le signal de sortie pour qu'il soit entre min et max.
- Limite sa vitesse de montée à moins de MaxV par seconde. (dérivée première)

limit3

- Limite le signal de sortie pour qu'il soit entre min et max.
- Limite sa vitesse de montée à moins de MaxV par seconde. (dérivée première)
- Limite sa vitesse de montée à moins de MaxV par seconde^2. (dérivée seconde)

lowpass

- Filtre passe-bas.

Pour plus de détails voir les composants de HAL ou les man pages des composants concernés.

Placer le code suivant dans un fichier appelé softstart.hal.

```
loadrt threads period1=1000000 name1=thread
loadrt siggen
loadrt lowpass
loadrt limit2
loadrt limit3
net square siggen.0.square => lowpass.0.in limit2.0.in limit3.0.in
net lowpass <= lowpass.0.out
net limit2 <= limit2.0.out
net limit3 <= limit3.0.out
setp siggen.0.frequency .1
setp lowpass.0.gain .01
setp limit2.0.maxv 2
setp limit3.0.maxv 2
setp limit3.0.maxa 10
addf siggen.0.update thread
addf lowpass.0 thread
addf limit2.0 thread
addf limit3.0 thread
start
loadusr halscope
```

Ouvrir un terminal et et lancer le fichier avec la commande suivante:

```
halrun -I softstart.hal
```

Pour démarrer l'oscilloscope de HAL pour la première fois, cliquer OK pour accepter le thread par défaut.

Ensuite, il faut ajouter les signaux à suivre aux canaux du scope. Cliquer sur le canal 1 puis sélectionner square depuis l'onglet Signaux. Répéter pour les canaux suivants en ajoutant lowpass, limit2 et limit3.

Ensuite, pour régler le signal du déclencheur cliquer sur le bouton Source est sélectionner square. Le bouton devrait changer pour Source Canal 1.

Puis, cliquer sur Simple dans le groupe Mode Run. L'oscillo devrait faire un balayage puis, afficher les traces.

Pour séparer les signaux et mieux les visualiser, cliquer sur un canal et utiliser le curseur de position verticale pour positionner les traces.

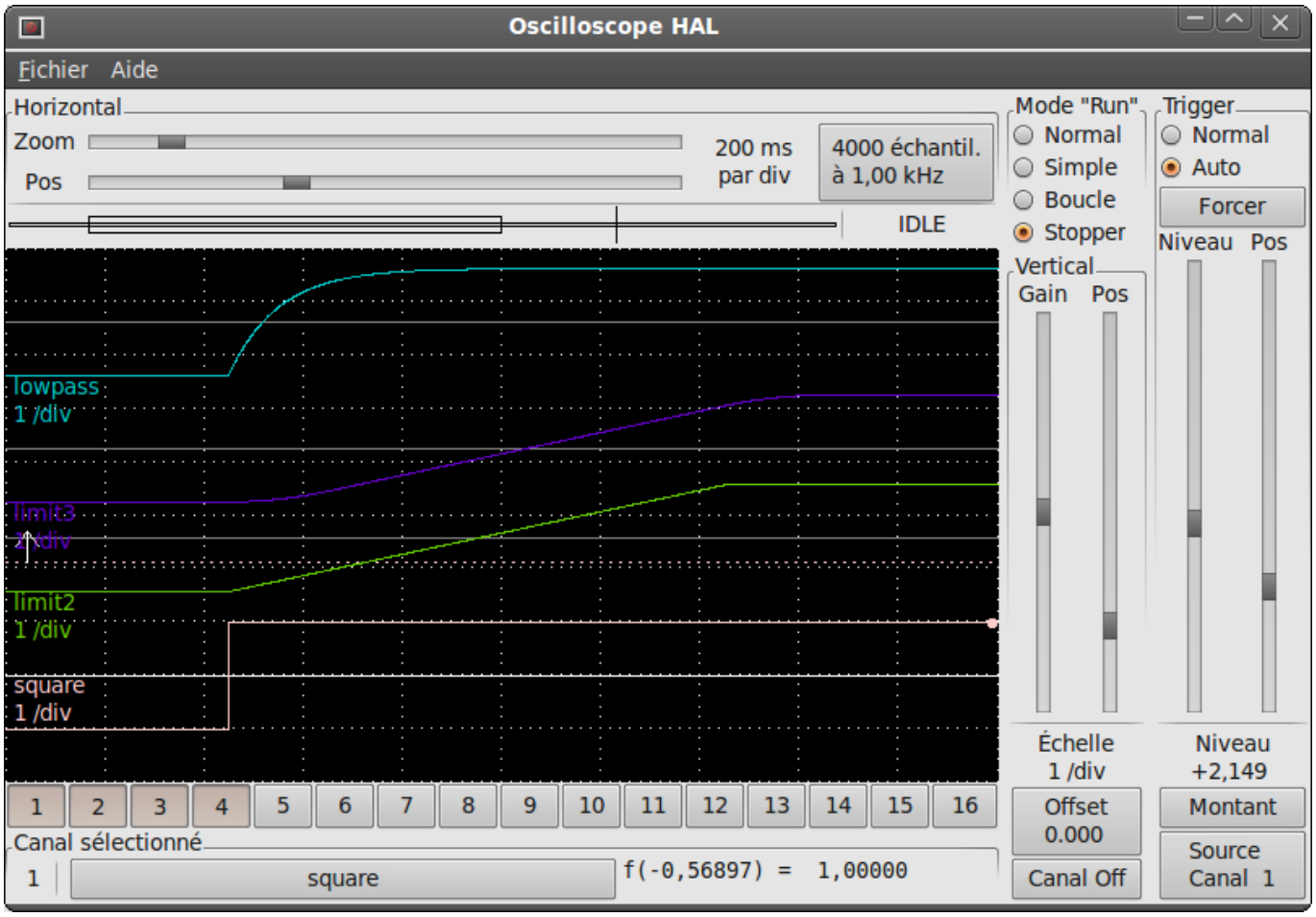


Figure 14.26: Amortissement d'un signal carré

Pour voir l'effet d'un changement du point de réglage des valeurs des composants, il est possible de passer des commandes depuis le terminal. Par exemple,pour voir différentes valeurs de gain pour le passe-bas, taper la commande suivante, puis essayer différentes valeurs:

```
setp lowpass.0.gain .01
```

Après un changement de réglage, relancer Halscope pour visualiser l'effet.

Pour terminer, taper exit dans le terminal pour fermer halrun et halscope. Ne pas refermer le terminal avec halrun en marche, la mémoire ne serait pas vidée proprement, ce qui pourrait empêcher LinuxCNC de se charger.

Pour tout savoir sur Halscope et Halrun voir le tutoriel de HAL.

14.8.4 HAL en autonome

Dans certains cas il peut être utile de lancer un écran GladeVCP avec juste HAL. Par exemple, nous avons un moteur pas à pas a piloter et tout ce qu'il nous faut pour notre application est une simple interface avec Marche/Arrêt plutôt que charger une application de CNC complète.

Dans l'exemple qui suit, nous allons créer ce simple panneau GladeVCP.

Syntaxe de base

```
# charge l’interface graphique winder.glade et la nome winder
loadusr -Wn winder gladevcp -c winder -u handler.py winder.glade

# charge les composants temps réel
loadrt threads name1=fast period1=50000 fp1=0 name2=slow period2=1000000
loadrt stepgen step_type=0 ctrl_type=v
loadrt hal_parport cfg=”0x378 out”

# ajoute les fonctions aux threads
addf stepgen.make-pulses fast
addf stepgen.update-freq slow
addf stepgen.capture-position slow
addf parport.0.read fast
addf parport.0.write fast

# effectue les connections de hal
net winder-step parport.0.pin-02-out <= stepgen.0.step
net winder-dir parport.0.pin-03-out <= stepgen.0.dir
net run-stepgen stepgen.0.enable <= winder.start_button



# démarre les threads
start

# commenter la ligne suivante pendant les essais et utiliser le mode interactif
pour voir les pins etc.
# option halrun -I -f start.hal

# attends que la GUI gladevcp nommée winder soit terminée
waitusr winder

# arrête tous les threads
stop

# décharge tous les composants de HAL avant de quitter
unloadrt all
```

14.9 comp: outil pour créer les modules HAL

14.9.1 Introduction

Écrire un composant de HAL peut se révéler être une tâche ennuyeuse, la plupart de cette tâche consiste à appeler des fonctions rtapi et hal et à contrôler les erreurs associées à ces fonctions. comp va écrire tout ce code pour vous, automatiquement.

Compiler un composant de HAL est également beaucoup plus simple en utilisant comp , que le composant fasse partie de l’arborescence de LinuxCNC, ou qu’il en soit extérieur.

Par exemple, cette portion des blocs ddt, codée en C, fait environ 80 lignes de code, alors que le composant équivalent est vraiment très court quand il est créé en utilisant le préprocesseur comp.

Exemple pour comp

```
component ddt ”Calcule la dérivée de la fonction d’entrée”;
pin in float in;
pin out float out;
variable float old;
```

```
function _;
license ”GPLv2 or later”;
;;
float tmp = in;
out = (tmp - old) / fperiod;
old = tmp;
```

14.9.2 Installation

Si une version pré-installée de LinuxCNC est utilisée, il sera nécessaire d’installer les paquets de développement en passant par Synaptic depuis le menu Système → Administration → Gestionnaire de paquets Synaptic ou en utilisant la commande suivante dans un terminal:

Installation des paquets de développement

```
sudo apt-get install linuxcnc-dev
```

14.9.3 Définitions

- component - Un composant est un simple module temps réel, qui se charge avec halcmd loadrt. Un fichier .comp spécifie un seul composant.
- instance - Un composant peut avoir plusieurs instances ou aucune. Chaque instance d’un composant est créée égale (elles ont toutes les mêmes pins, les mêmes paramètres, les mêmes fonctions et les mêmes données) mais elle se comporte de manière différente quand leurs pins, leurs paramètres et leur données ont des valeurs différentes.
- singleton - Il est possible pour un composant d’être un singleton (composant dont il n’existe qu’une seule instance), dans ce cas, exactement une seule instance est créée. Il est rarement logique d’écrire un composant singleton , à moins qu’il n’y ait qu’un seul objet de ce type dans le système (par exemple, un composant ayant pour but de fournir une pin avec le temps Unix courant, ou un pilote matériel pour le haut parleur interne du PC)

14.9.4 Création d’instance

Pour un singleton, une seule instance est créée quand le composant est chargé.

Pour un non-singleton, le paramètre count du module détermine combien d’instances seront créées. Si count n’est pas spécifié, le paramètre names du module détermine combien d’instances nommées seront créées. Si ni count, ni names n’est spécifié, une simple instance numérotée sera créée.

14.9.5 Paramètres implicites

Le paramètres period est implicitement passé aux fonctions, c’est la durée, en nanosecondes, de la dernière période d’exécution du comp. Les fonctions qui utilisent des flottants peuvent aussi se référer à fperiod, qui est la durée en secondes, soit (period*1e-9). Cela peut être utile pour les comps ayant besoin de l’information de timming.

14.9.6 Syntaxe

Un fichier .comp commence par un certain nombre de déclarations, puis par un délimiteur constitué de deux points virgule ;; seuls sur leur propre ligne et enfin du code C implémentant les fonctions du module.

Déclarations d'include:

- component HALNAME (DOC);
- pin PINDIRECTION TYPE HALNAME ([SIZE]|[MAXSIZE : CONDSIZE]) (if CONDITION) (= STARTVALUE) (DOC);
- param PARAMDIRECTION TYPE HALNAME ([SIZE]|[MAXSIZE : CONDSIZE]) (if CONDITION) (= STARTVALUE) (DOC);
- function HALNAME (fp | nofp) (DOC);
- option OPT (VALUE);
- variable CTYPE NAME ([SIZE]);
- description DOC;
- see'also DOC;
- license LICENSE;
- author AUTHOR;

Les parenthèses indiquent un item optionnel. Une barre verticale indique une alternative. Les mots en MAJUSCULES indiquent une variable texte, comme ci-dessous:

- NAME - Un identifiant C standard.
- STARREDNAME - Un identifiant C, précédé ou non d'une *. Cette syntaxe est utilisée pour déclarer les variables qui sont des pointeurs. Noter qu'à cause de la grammaire, il ne doit pas y avoir d'espace entre * et le nom de la variable.
- HALNAME - Un identifiant étendu. Lorsqu'ils sont utilisés pour créer un identifiant de HAL, tous les caractères soulignés sont remplacés par des tirets, tous les points et les virgules de fin, sont supprimés, ainsi **ce_nom_** est remplacé par **ce-nom**, si le nom est "", alors le point final est enlevé aussi, ainsi "function" donne un nom de fonction HAL tel que "component.<num>" au lieu de "component.<num>."

S'il est présent, le préfixe hal_ est enlevé du début d'un nom de composant lors de la création des pins, des paramètres et des fonctions.

Dans l'identifiant de HAL pour une pin ou un paramètre, # indique un membre de tableau, il doit être utilisé conjointement avec une déclaration [SIZE]. Les hash marks sont remplacées par des nombres de 0-barrés équivalents aux nombres de caractères #.

Quand ils sont utilisés pour créer des identifiants C, les changements de caractères suivants sont appliqués au HALNAME:

+ . Tous les caractères "#" sont enlevés ainsi que tous les caractères ".", "" ou "-" immédiatement devant eux. . Dans un nom, tous les caractères "." et "-" sont remplacés par "". . Les caractères "_" répétitifs sont remplacés par un seul caractère "_".

Un "_" final est maintenu, de sorte que les identifiants de HAL, qui autrement seraient en conflit avec les noms ou mots clé réservés (par exemple: min), puissent être utilisés.

HALNAME	Identifiant C	Identifiant HAL
x_y_z	x_y_z	x-y-z
x-y.z	x_y_z	x-y.z
x_y_z_	x_y_z_	x-y-z
x.##.y	x_y(MM)	x.MM.z
x.##	x(MM)	x.MM

- if CONDITION - Une expression impliquant la personnalité d'une variable non nulle quand la variable ou le paramètre doit être créé.
- SIZE - Un nombre donnant la taille d'un tableau. Les items des tableaux sont numérotés de 0 à SIZE-1.
- MAXSIZE : CONDSIZE - Un nombre donnant la taille maximum d'un tableau, suivi d'une expression impliquant la personnalité d'une variable et qui aura toujours une valeur inférieure à MAXSIZE. Quand le tableau est créé sa taille est égale à CONDSIZE.
- DOC - Une chaine qui documente l'item. La chaine doit être au format C, entre guillemets, comme:

```
"Sélectionnez le front désiré: TRUE pour descendant, FALSE pour montant"
```

  ou au format Python triples guillemets, pouvant inclure des caractères newlines et des guillemets, comme:

```
"""The effect of this parameter, also known as "the orb of zot",
will require at least two paragraphs to explain.

Hopefully these paragraphs have allowed you to understand "zot"
better."""
```

  La chaine de documentation est en format "groff -man". Pour plus d'informations sur ce format de markup, voyez groff_man(7) . Souvenez vous que comp interprète backslash comme Echap dans les chaines, ainsi par exemple pour passer le mot example en font italique, écrivez:

```
\\fIexample\\fB
```

- TYPE - Un des types de HAL: bit, signed (signé), unsigned (non signé) ou float (flottant). Les anciens noms s32 et u32 peuvent encore être utilisés, mais signed et unsigned sont préférables.
- PINDIRECTION - Une des ces directions: in, out, ou io . Le composant pourra positionner la valeur d'une pin de sortie, il pourra lire la valeur sur une pin d'entrée et il pourra lire ou positionner la valeur d'une pin io.
- PARAMDIRECTION - Une des valeurs suivantes: r ou rw. Le composant pourra positionner la valeur d'un paramètre r et il pourra positionner ou lire la valeur d'un paramètre rw.
- STARTVALUE - Spécifie la valeur initiale d'une pin ou d'un paramètre. Si il n'est pas spécifié, alors la valeur par défaut est 0 ou FALSE, selon le type de l'item.

14.9.6.1 Fonctions HAL

- fp - Indique que la fonction effectuera ses calculs en virgule flottante.
- nofp - Indique que la fonction effectuera ses calculs sur des entiers. Si il n'est pas spécifié, fp est utilisé. Ni comp ni gcc ne peuvent détecter l'utilisation de calculs en virgule flottante dans les fonctions marquées nofp.

14.9.6.2 Options

Selon le nom de l'option OPT, les valeurs VALUE varient. Les options actuellement définies sont les suivantes:

- option singleton yes - (défaut: no) Ne crée pas le paramètre count de module et crée toujours une seule instance. Avec singleton, les items sont nommés composant-name.item-name et sans singleton, les items des différentes instances sont nommés composant-name.<num>.item-name.
- option default_count number - (défaut: 1) Normalement, le paramètre count par défaut est 0. Si spécifié, count remplace la valeur par défaut.
- option count_function yes - (défaut: no) Normalement, le numéro des instances à créer est specifié dans le paramètre count du module, si count_function est spécifié, la valeur retournée par la fonction int get_count(void) est utilisée à la place de la valeur par défaut et le paramètre count du module n'est pas défini.
- option rtapi_app no - (défaut: yes) Normalement, les fonctions rtapi_app_main et rtapi_app_exit sont définies automatiquement. Avec option rtapi_app no, elles ne le seront pas et doivent être fournies dans le code C. Quand vous implémentez votre propre rtapi_app_main, appellez la fonction int export(char *prefix, long extra_arg) pour enregistrer les pins, paramètres et fonctions pour préfixer.
- option data TYPE - (défaut: none) **obsolète** If specified, each instance of the component will have an associated data block of TYPE (which can be a simple type like float or the name of a type created with typedef). Dans les nouveaux components, variable doit être utilisé en remplacement.
- option extra_setup yes - (défaut: no) Si spécifié, appelle la fonction définie par EXTRA_SETUP pour chaque instance. Dans le cas de la rtapi_app_main automatiquement définie, extra_arg est le numéro de cette instance.
- option extra_cleanup yes - (défaut: no) Si spécifié, appelle la fonction définie par EXTRA_CLEANUP depuis la fonction définie automatiquement rtapi_app_exit, ou une erreur est détectée dans la fonction automatiquement définie rtapi_app_main.
- option userspace yes - (défaut: no) Si spécifié, ce fichier décrit un composant d'espace utilisateur, plutôt que le réel. Un composant d'espace utilisateur peut ne pas avoir de fonction définie par la directive de fonction. Au lieu de cela, après que toutes les instances soient construites, la fonction C user_mainloop() est appelée. Dès la fin de cette fonction, le composant se termine. En règle générale, user_mainloop() va utiliser FOR_ALL_INSTS() pour effectuer la mise à jour pour chaque action, puis attendre un court instant. Une autre action commune dans user_mainloop() peut être d'appeler le gestionnaire de boucles d'événements d'une interface graphique.
- option userinit yes - (défaut: no) Si spécifiée, la fonction userinit(argc,argv) est appelée avant rtapi_app_main() (et cela avant l'appel de hal_init()). Cette fonction peut traiter les arguments de la ligne de commande ou exécuter d'autres actions. Son type de retour est void; elle peut appeler exit() et si elle le veut, se terminer sans créer de composant HAL (par exemple, parce que les arguments de la ligne de commande sont invalides).

```
Si aucune option VALUE n'est spécifiée, alors c'est équivalent à
spécifier la valeur 'b''…b'' yes' . Le résultat consécutif à l'assignation
d'une valeur inappropriée à
une option est indéterminé. Le résultat consécutif à n'utiliser aucune
autre option est indéfini.
```

14.9.6.3 Licence et auteur

- LICENSE - Spécifie la license du module, pour la documentation et pour le module déclaré dans MODULE_LICENSE(). Par exemple, pour spécifier que la licence des modules est la GPL v2 ou suivantes, license "GPL"; // indique GPL v2 ou suivantes

```
Pour d'autres informations sur la signification du MODULE_LICENSE() et les
identificateurs de license additionnels, voir '<linux/module.h>'. ou la page
'rtapi_module_param(3)' du manuel.
```

- AUTHOR - Spécifie l'auteur du module, pour la documentation

14.9.6.4 Stockage des données par instance

- variable CTYPE STARREDNAME;
- variable CTYPE STARREDNAME[SIZE];
- variable CTYPE STARREDNAME = DEFAULT;
- variable CTYPE STARREDNAME[SIZE] = DEFAULT; Déclare la variable par-instance STARREDNAME de type CTYPE, optionnellement comme un tableau de SIZE items et optionnellement avec une valeur DEFAULT. Les items sans DEFAULT sont initialisés all-bits-zero. CTYPE est un simple mot de type C, comme float, u32, s32, etc. Les variables d'un tableau sont mises entre crochets.

  Si une variable doit être de type pointeur, il ne doit y avoir aucun espace entre l'étoile "*" et le nom de la variable. Néanmoins, la forme suivante est acceptable:

```
variable int *bonexemple;
```

  Mais les formes suivantes ne sont pas acceptables:

```
variable int* mauvaisexemple;
variable int * mauvaisexemple;
```

14.9.6.5 Commentaires

Les commentaires de style C++ une ligne (// …) et de style C multi-lignes (/* … */) sont supportés tous les deux dans la section déclaration.

14.9.7 Restrictions sur les fichiers comp

Bien que HAL permette à une pin, un paramètre et une fonction d'avoir le même nom, comp ne le permet pas.

Les noms de variable et de fonction qui ne doivent pas être utilisés ou qui posent problème sont les suivants:

- Tous noms commençant par _comp.
- comp_id
- fperiod
- rtapi_app_main
- rtapi_app_exit
- extra_setup
- extra_cleanup

14.9.8 Conventions des macros

En se basant sur les déclarations des items de section, comp crée une structure C appelée struct comp_state. Cependant, au lieu de faire référence aux membres de cette structure (par exemple: *(inst->name)), il leur sera généralement fait référence en utilisant les macros ci-dessous. Certains détails de struct comp_state et ces macros peuvent différer d'une version de comp à une autre.

- FUNCTION(name) - Cette macro s'utilise au début de la définition d'une fonction temps réel qui aura été précédemment déclarée avec function NAME. function inclus un paramètre period qui est le nombre entier de nanosecondes entre les appels à la fonction.
- EXTRA_SETUP() - Cette macro s'utilise au début de la définition de la fonction appelée pour exécuter les réglages complémentaires à cette instance. Une valeur de retour négative Unix errno indique un défaut (par exemple: elle retourne -EBUSY comme défaut à la réservation d'un port d'entrées/sorties), une valeur égale à 0 indique le succès.
- EXTRA_CLEANUP() - Cette macro s'utilise au début de la définition de la fonction appelée pour exécuter un nettoyage (cleanup) du composant. Noter que cette fonction doit nettoyer toutes les instances du composant, pas juste un. Les macros pin_name, parameter_name et data ne doivent pas être utilisées ici.
- pin_name ou parameter_name - Pour chaque pin, pin_name ou pour chaque paramètre, parameter_name il y a une macro qui permet d'utiliser le nom seul pour faire référence à la pin ou au paramètre. Quand pin_name ou parameter_name sont des tableaux, la macro est de la forme pin_name(idx) ou param_name(idx) dans laquelle idx est l'index dans le tableau de pins. Quand le tableau est de taille variable, il est seulement légal de faire référence aux items par leurs condsize.

  ```
  Quand un item est conditionnel, il est seulement légal de faire
  référence à cet item quand ses conditions sont évaluées à des
  valeurs différentes de zéro.
  ```

- variable_name - Pour chaque variable, il y a une macro variable_name qui permet au nom seul d'être utilisé pour faire référence à la variable. Quand variable_name est un tableau, le style normal de C est utilisé: variable_name[idx]
- data- Si l'option data est spécifiée, cette macro permet l'accès à l'instance de la donnée.
- fperiod - Le nombre de secondes en virgule flottante entre les appels à cette fonction temps réel.
- FOR_ALL_INSTS() {**...**} - Pour les composants de l'espace utilisateur. Cette macro utilise la variable *struct state inst pour itérer au dessus de toutes les instances définies. Dans le corps de la boucle, les macros pin_name, parameter_name et data travaillent comme elles le font dans les fonctions temps réel.

14.9.9 Composants avec une seule fonction

Si un composant a seulement une fonction et que la chaine FUNCTION n'apparaît nulle part après ;;, alors la portion après ;; est considérée comme étant le corps d'un composant simple fonction.

14.9.10 Personnalité du composant

Si un composant a n'importe combien de pins ou de paramètres avec un if condition ou [maxsize : condsize], il est appelé un composant avec personnalité. La personnalité de chaque instance est spécifiée quand le module est chargé. La personnalité peut être utilisée pour créer les pins seulement quand c'est nécessaire. Par exemple, la personnalité peut être utilisée dans un composant logique, pour donner un nombre variable de broches d'entrée à chaque porte logique et permettre la sélection de n'importe quelle fonction de logique booléenne de base and, or et xor.

14.9.11 Compiler un fichier .comp dans l'arborescence

Placer le fichier .comp dans le répertoire linuxcnc/src/hal/components et lancer/relancer make. Les fichiers Comp sont automatiquement détectés par le système de compilation.

Si un fichier .comp est un pilote de périphérique, il peut être placé dans linuxcnc/src/hal/components et il y sera construit excepté si LinuxCNC est configuré en mode simulation.

14.9.12 Compiler un composant temps réel hors de l'arborescence

comp peut traiter, compiler et installer un composant temps réel en une seule étape, en plaçant rtexample.ko dans le répertoire du module temps réel de LinuxCNC:

```
comp --install rtexample.comp
```

Ou il peut aussi être traité et compilé en une seule étape en laissant example.ko (ou example.so pour la simulation) dans le répertoire courant:

```
comp --compile rtexample.comp
```

Ou il peut simplement être traité en laissant example.c dans le répertoire courant:

```
comp rtexample.comp
```

comp peut aussi compiler et installer un composant écrit en C, en utilisant les options --install et --compile comme ci-dessous:

```
comp --install rtexample2.c
```

La documentation au format man peut être créée à partir des informations de la section declaration:

```
comp --document rtexample.comp
```

La manpage résultante, exemple.9 peut être lue avec:

```
man ./exemple.9
```

ou copiée à un emplacement standard pour une page de manuel.

14.9.13 Compiler un composant de l'espace utilisateur hors de l'arborescence

comp peut traiter, compiler et installer un document de l'espace utilisateur:

```
comp usrexample.comp
comp --compile usrexample.comp
comp --install usrexample.comp
comp --document usrexample.comp
```

Cela fonctionne seulement pour les fichiers .comp mais pas pour les fichiers .c.

14.9.14 Exemples

14.9.14.1 constant

Noter que la déclaration "function _" crée les fonctions nommées "constant.0" , etc. Le nom du fichier doit correspondre au nom du composant.

```
component constant;
pin out float out;
param r float value = 1.0;
function _;
license "GPL"; // indique la GPL v2 ou suivantes
;;
FUNCTION(_) { out = value; }
```

14.9.14.2 sincos

Ce composant calcule le sinus et le cosinus d'un angle entré en radians. Il a différentes possibilités comme les sorties sinus et cosinus de siggen, parce que l'entrée est un angle au lieu d'être librement basé sur un paramètre frequency.

Les pins sont déclarées avec les noms sin' et cos' dans le code source pour que ça n'interfère pas avec les fonctions sin() et cos(). Les pins de HAL sont toujours appelées sincos.<num>.sin.

```
component sincos;
pin out float sin_;
pin out float cos_;
pin in float theta;
function _;
license "GPL"; // indique la GPL v2 ou suivantes
;;
#include <rtapi_math.h>
FUNCTION(_) { sin_ = sin(theta); cos_ = cos(theta); }
```

14.9.14.3 out8

Ce composant est un pilote pour une carte imaginaire appelée out8, qui a 8 pins de sortie digitales qui sont traitées comme une simple valeur sur 8 bits. Il peut y avoir un nombre quelconque de ces cartes dans le système et elles peuvent avoir des adresses variées. La pin est appelée out' parce que out est un identifiant utilisé dans <asm/io.h>. Il illustre l'utilisation de EXTRA_SETUP et de EXTRA_CLEANUP pour sa requête de région d'entrées/sorties et libère cette région en cas d'erreur ou quand le module est déchargé.

```
component out8;
pin out unsigned out_ "Output value; only low 8 bits are used";
param r unsigned ioaddr;

function _;

option count_function;
option extra_setup;
option extra_cleanup;
option constructable no;

license "GPL";
;;
#include <asm/io.h>

#define MAX 8
int io[MAX] = {0,};
RTAPI_MP_ARRAY_INT(io, MAX, "I/O addresses of out8 boards");

int get_count(void) {
   int i = 0;
```

```
   for(i=0; i<MAX && io[i]; i++) { /* Nothing */ }
   return i;
}

EXTRA_SETUP() {
    if(!rtapi_request_region(io[extra_arg], 1, "out8")) {
        // set this I/O port to 0 so that EXTRA_CLEANUP does not release the IO
        // ports that were never requested.
        io[extra_arg] = 0;
        return -EBUSY;
    }
    ioaddr = io[extra_arg];
    return 0;
}

EXTRA_CLEANUP() {
    int i;
    for(i=0; i < MAX && io[i]; i++) {
        rtapi_release_region(io[i], 1);
    }
}

FUNCTION(_) { outb(out_, ioaddr); }
```

14.9.14.4 hal_loop

```
component hal_loop;
pin out float example;
```

Ce fragment de composant illustre l'utilisation du préfixe hal_ dans un nom de composant. loop est le nom d'un module standard du kernel Linux, donc un composant loop ne pourrait pas être chargé si le module loop de Linux est également présent.

Quand il le charge, halcmd montre un composant appelé hal_loop. Cependant, les pins affichées par halcmd sont loop.0.example et non hal-loop.0.example.

14.9.14.5 arraydemo

Ce composant temps réel illustre l'utilisation d'un tableau de taille fixe:

```
component arraydemo "4-bit Shift register";
pin in bit in;
pin out bit out-# [4];
function _ nofp;
license "GPL"; // indique la GPL v2 ou ultérieures
;;
int i;
for(i=3; i>0; i--) out(i) = out(i-1);
out(0) = in;
```

14.9.14.6 rand

Ce composant de l'espace utilisateur modifie la valeur de ses pins de sortie vers une nouvelle valeur aléatoire dans l'étendue (0,1) à chaque 1ms.

```
component rand;
option userspace;

pin out float out;
license "GPL";
;;
#include <unistd.h>

void user_mainloop(void) {
    while(1) {
        usleep(1000);
        FOR_ALL_INSTS() out = drand48();
    }
}
```

14.9.14.7 logic

Ce composant temps réel montre l'utilisation de la personnalité pour créer un tableau de taille variable et des pins optionnelles.

```
component logic "LinuxCNC HAL component providing experimental logic functions";
pin in bit in-##[16 : personality & 0xff];
pin out bit and if personality & 0x100;
pin out bit or if personality & 0x200;
pin out bit xor if personality & 0x400;
function _ nofp;
description """
Experimental general logic function component.  Can perform and, or
and xor of up to 16 inputs.  Determine the proper value for personality
by adding:
.IP \\(bu 4
The number of input pins, usually from 2 to 16
.IP \\(bu
256 (0x100)  if the and output is desired
.IP \\(bu
512 (0x200)  if the or output is desired
.IP \\(bu
1024 (0x400)  if the xor (exclusive or) output is desired""";
license "GPL";
;;
FUNCTION(_) {
    int i, a=1, o=0, x=0;
    for(i=0; i < (personality & 0xff); i++) {
        if(in(i)) { o = 1; x = !x; }
        else { a = 0; }
    }
    if(personality & 0x100) and = a;
    if(personality & 0x200) or = o;
    if(personality & 0x400) xor = x;
}
```

Une ligne de chargement typique pourrait être:

```
    loadrt logic count=3 personality=0x102,0x305,0x503
```

qui créerait les pins suivantes:

- Une porte AND à 2 entrées: logic.0.and, logic.0.in-00, logic.0.in-01

- des portes AND et OR à 5 entrées: logic.1.and, logic.1.or, logic.1.in-00, logic.1.in-01, logic.1.in-02, logic.1.in-03, logic.1.in-04,
- des portes AND et XOR à 3 entrées: logic.2.and, logic.2.xor, logic.2.in-00, logic.2.in-01, logic.2.in-02

14.10 Créer des composants de l’espace utilisateur

14.10.1 Utilisation de base en Python

Un composant de l’espace utilisateur commence par créer ses pins et ses paramètres, puis il entre dans une boucle dans laquelle il va positionner périodiquement toutes ses sorties en fonction de ses entrées. Le composant suivant, un passe-tout, copie la valeur vue sur ses pins d’entrée (passe_tout.in) vers ses pins de sortie (passe_tout.out) approximativement une fois par seconde.

```
#!/usr/bin/env python3
import hal, time
h = hal.component(”passe_tout”)
h.newpin(”in”, hal.HAL_FLOAT, hal.HAL_IN)
h.newpin(”out”, hal.HAL_FLOAT, hal.HAL_OUT)
h.ready()
try:
    while 1:
        time.sleep(1)
        h[’out’] = h[’in’]
except KeyboardInterrupt:
    raise SystemExit
```

Copier le listing précédent dans un fichier nommé passe_tout, le rendre exécutable par un chmod +x et le placer dans son $PATH. On peut alors l’essayer en faisant:

```
halrun
halcmd: loadusr passe_tout
halcmd: show pin
Component Pins:
Owner Type  Dir     Value  Name
 03   float IN          0  passe_tout.in
 03   float OUT         0  passe_tout.out

halcmd: setp passe_tout.in 3.14
halcmd: show pin
Component Pins:
Owner Type  Dir     Value  Name
 03   float IN       3.14  passe_tout.in
 03   float OUT      3.14  passe_tout.out
```

14.10.2 Composants de l’espace utilisateur et délais

Si vous tapez rapidement show pin, vous pourrez voir que passe_tout.out conserve un moment son ancienne valeur de 0. Ceci est dû à l’appel de la fonction time.sleep(1), qui fait que les pins de sortie changent d’état, au plus, une fois par seconde. Parce-que ce composant appartient à l’espace utilisateur, ce délai peut apparaître plus long, par exemple si la mémoire utilisée par le composant

pass_tout est échangée avec le disque dur, le délai peut être allongé jusqu'au rafraîchissement de la mémoire.

Ces composants de l'espace utilisateur conviennent parfaitement pour des éléments tels que des panneaux de contrôle pour lesquels des délais de l'ordre de quelques millisecondes sont imperceptibles. Ils ne conviennent pas, en revanche, pour envoyer des impulsions de pas vers une carte de pilotage de périphériques pour lesquelles les délais doivent rester de l'ordre de quelques microsecondes, dans tous les cas.

14.10.3 Créer les pins et les paramètres

```
h = hal.component("passe_tout")
```

Le composant lui-même est créé par l'appel du constructeur hal.component. Les arguments sont le nom du composant HAL et optionnellement, le préfixe utilisé pour les noms de pin et de paramètre. Si le préfixe n'est pas spécifié, le nom du composant est utilisé.

```
h.newpin("in", hal.HAL_FLOAT, hal.HAL_IN)
```

Puis les pins sont créées par appels des méthodes sur l'objet composant. Les arguments sont: pin nom suffixe, type de pin et direction de la pin. Pour les paramètres, les arguments sont: paramètre nom suffixe, type de paramètre et direction du paramètre.

Table 14.3: Noms des options de HAL

Types de Pin et Paramètre:	**HAL_BIT**	**HAL_FLOAT**	**HAL_S32**	**HAL_U32**
Directions des pins:	HAL_IN	HAL_OUT	HAL_IO	
Directions des paramètres:	HAL_RO	HAL_RW		

Le nom complet d'une pin ou d'un paramètre est formé en joignant le préfixe avec le suffixe par un **.**, comme dans l'exemple où la pin créée est appelée passe_tout.in.

```
h.ready()
```

Une fois toutes les pins et les paramètres créés, la méthode .ready() est appelée.

14.10.3.1 Changer le préfixe

Le préfixe peut être changé en appelant la méthode .setprefix(). Le préfixe courant peut être retrouvé en appelant la méthode .getprefix().

14.10.4 Lire et écrire les pins et les paramètres

Pour les pins et les paramètres qui sont aussi des identifiants Python, la valeur est accessible ou ajustable en utilisant la syntaxe des attributs suivante:

```
h.out = h.in
```

Pour les pins et les paramètres qui sont aussi des identifiants Python, la valeur est accessible ou ajustable en utilisant la syntaxe de sous-script suivante:

```
h[out] = h[in]
```

14.10.4.1 Pilotage des pins de sortie (HAL_OUT)

Périodiquement, habituellement dans le temps de réponse de l'horloge, toutes les pins HAL_OUT doivent être pilotées en leur assignant une nouvelle valeur. Ceci doit être fait que la valeur soit différente ou non de la valeur précédemment assignée. Quand la pin est connectée au signal, l'ancienne valeur de sortie n'est pas copiée vers le signal, la valeur correcte n'apparaîtra donc sur le signal qu'une fois que le composant lui aura assigné une nouvelle valeur.

14.10.4.2 Pilotage des pins bidirectionnelles (HAL_IO)

La règle mentionnée ci-dessus ne s'applique pas aux pins bidirectionnelles. Au lieux de cela, une pin bidirectionnelle doit seulement être pilotée par le composant et quand le composant souhaite changer sa valeur. Par exemple, dans l'interface codeur, le composant codeur positionne seulement la pin index-enable à FALSE quand une impulsion d'index est vue et que l'ancienne valeur est TRUE, mais ne la positionne jamais à TRUE. Positionner de manière répétitive la pin à FALSE pourrait faire qu'un autre composant connecté agisse comme si une nouvelle impulsion d'index avait été vue.

14.10.5 Quitter

Une requête halcmd unload pour le composant est délivrée comme une exception KeyboardInterrupt . Quand une requête de déchargement arrive, le processus doit quitter dans un court laps de temps ou appeler la méthode .exit() sur le composant si un travail substantiel, comme la lecture ou l'écriture de fichiers, doit être fourni pour terminer le processus d'arrêt.

14.10.6 Idées de projets

- Créer un panneau de contrôle extérieur avec boutons poussoirs, interrupteurs et voyants. Connecter le tout à un microcontrôleur et raccorder le microcontrôleur à un PC en utilisant une liaison série. Python est vraiment capable d'interfacer une liaison série grâce à son module pyserial (Paquet python-serial, dans les dépôts universe d'Ubuntu)
- Relier un module d'affichage à LCD LCDProc et l'utiliser pour afficher les informations de votre choix (Paquet lcdproc, dans les dépôts universe d'Ubuntu)
- Créer un panneau de contrôle virtuel utilisant n'importe quelle librairie d'interface graphique supportée par Python (gtk, qt, wxwindows, etc)

Part V

Advanced Topics

Chapter 15

La cinématique dans LinuxCNC

15.1 Introduction

Habituellement quand nous parlons de machines CNC, nous pensons à des machines programmées pour effectuer certains mouvements et effectuer diverses tâches. Pour avoir une représentation unifiée dans l'espace de ces machines, nous la faisons correspondre à la vision humaine de l'espace en 3D, la plupart des machines (sinon toutes) utilisent un système de coordonnées courant, le système Cartésien.

Le système de coordonnées Cartésiennes est composé de 3 axes (X, Y, Z) chacun perpendiculaire aux 2 autres. [1]

Quand nous parlons d'un programme G-code (RS274/NGC) nous parlons d'un certain nombre de commandes (G0, G1, etc.) qui ont comme paramètres (X- Y- Z-). Ces positions se réfèrent exactement à des positions Cartésiennes. Une partie du contrôleur de mouvements de LinuxCNC est responsable de la translation entre ces positions et les positions correspondantes de la cinématique de la machine[2].

15.1.1 Les articulations par rapport aux axes

Une articulation, pour une machine CNC est un des degrés physiques de liberté de la machine. Elle peut être linéaire (vis à billes) ou rotative (table tournante, articulations d'un bras robotisé). Il peut y avoir n'importe quel nombre d'articulations sur une machine. Par exemple, un robot classique dispose de 6 articulations et une fraiseuse classique n'en a que 3.

Sur certaines machines, les articulations sont placées de manière à correspondre aux axes cinématiques (articulation 0 le long de l'axe X, articulation 1 le long de l'axe Y et articulation 2 le long de l'axe Z), ces machines sont appelées machines Cartésiennes (ou encore machines à cinématiques triviales . Ce sont les machines les plus courantes parmi les machines-outils mais elles ne sont pas courantes dans d'autres domaines comme les machines de soudage (ex: robots de soudage de type puma).

15.2 Cinématiques triviales

Comme nous l'avons vu, il y a un groupe de machines sur lesquelles chacun des axes est placé le long d'un des axes Cartésiens. Sur ces machines le passage, du plan de l'espace Cartésien (le programme G-code) au plan de l'espace articulation (l'actuateur actuel de la machine), est trivial. C'est un simple plan 1:1:

[1] Le mot axes est aussi communément (et incorrectement) utilisé à propos des machines CNC, il fait référence aux directions des mouvements de la machine.

[2] Cinématique: une fonction à deux voies pour transformer un espace Cartésien en espace à articulations

```
pos->tran.x = joints[0];
pos->tran.y = joints[1];
pos->tran.z = joints[2];
pos->a = joints[3];
pos->b = joints[4];
pos->c = joints[5];
```

Dans l'extrait de code ci-dessus, nous pouvons voir comment le plan est fait: la position X est identique avec la articulation 0, Y avec la articulation 1 etc. Nous nous référons dans ce cas à une cinématique directe (une transformation avant), tandis que dans l'extrait de code suivant il est fait référence à une cinématique inverse (ou une transformation inverse):

```
joints[0] = pos->tran.x;
joints[1] = pos->tran.y;
joints[2] = pos->tran.z;
joints[3] = pos->a;
joints[4] = pos->b;
joints[5] = pos->c;
```

Comme on peut le voir, c'est assez simple de faire la transformation d'une machine à cinématique banale (ou Cartésienne). Cela devient un peu plus compliqué si il manque un axe à la machine.[3][4]

15.3 Cinématiques non triviales

Il peut y avoir un certain nombre de types de configurations de machine (robots: puma, scara; hexapodes etc.) Chacun d'eux est mis en place en utilisant des articulations linéaires et rotatives. Ces articulations ne correspondent pas habituellement avec les coordonnées Cartésiennes, cela nécessite une fonction cinématique qui fasse la conversion (en fait 2 fonctions: fonction en avant et inverse de la cinématique).

Pour illustrer ce qui précède, nous analyserons une simple cinématique appelée bipode (une version simplifiée du tripode, qui est déjà une version simplifiée de l'hexapode).

[3] Si la machine (par exemple un tour) est montée avec seulement les axes X, Z et A et que le fichier d'init de LinuxCNC contient uniquement la définition de ces 3 articulations, alors l'assertion précédente est fausse. Parce-que nous avons actuellement (joint0=x, joint1=Z, joint2=A) ce qui suppose que joint1=Y. Pour faire en sorte que cela fonctionne dans LinuxCNC il suffit de définir tous les axes (XYZA), LinuxCNC utilisera alors une simple boucle dans HAL pour l'axe Y inutilisé.

[4] Une autre façon de le faire fonctionner, est de changer le code correspondant et recompiler le logiciel.

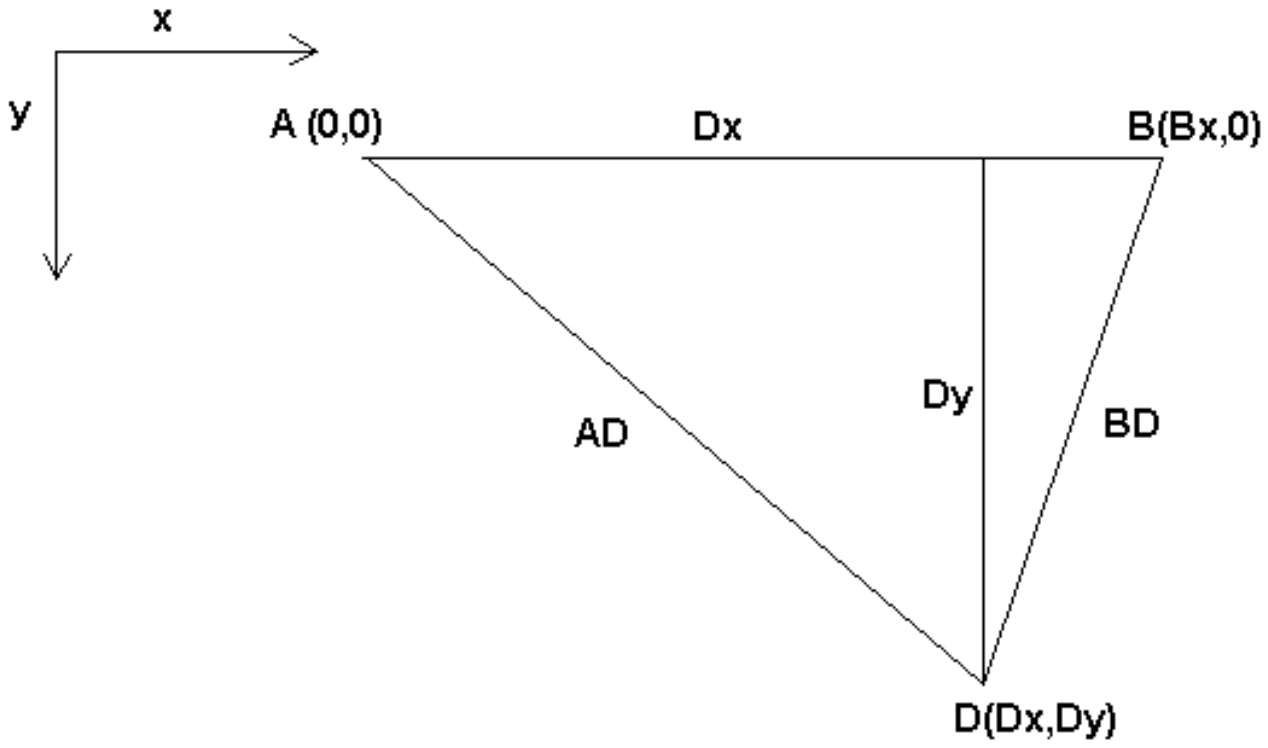


Figure 15.1: Définir un bipode

Le bipode dont nous parlons est un appareil, composé de deux moteurs placés sur un mur, à cet appareil un mobile est suspendu par des fils. Les articulations dans ce cas sont les distances entre le mobile et les moteurs de l'appareil (nommées AD et BD sur la figure ci-dessus.

La position des moteurs est fixée par convention. Le moteur A est en (0,0), qui signifie que sa coordonnée X est 0 et sa coordonnée Y également 0. Le moteur B est placé en (Bx, 0), se qui veut dire que sa coordonnée X est Bx.

Notre pointe mobile se trouvera au point D défini par les distances AD et BD, et par les coordonnées Cartésiennes Dx, Dy.

La tâche de la cinématique consistera à transformer les longueurs des articulations en (AD, BD) en coordonnées Cartésiennes (Dx, Dy) et vice-versa.

15.3.1 Transformation avant

Pour effectuer la transformation de l'espace articulation en espace Cartésien nous allons utiliser quelques règles de trigonomètrie (le triangle rectangle déterminé par les points (0,0), (Dx,0), (Dx,Dy) et le triangle rectangle (Dx,0), (Bx,0) et (Dx,Dy).

Nous pouvons voir aisément que $\mathbf{AD^2=x^2+y^2}$, de même que $\mathbf{BD^2=(Bx-x)^2+y^2}$.

Si nous soustrayons l'un de l'autre nous aurons: $\mathbf{AD^2-BD^2=x^2+y^2-x^2+2*x*Bx-Bx^2-y^2}$

et par conséquent: $\mathbf{x=(AD^2-BD^2+Bx^2)/(2*Bx)}$

De là nous calculons: $\mathbf{y=sqrt(AD^2-x^2)}$

Noter que le calcul inclus la racine carrée de la différence, mais qu'il n'en résulte pas un nombre réel. Si il n'y a aucune coordonnée Cartésienne pour la position de cette articulation, alors la position est dite singulière. Dans ce cas, la cinématique inverse retourne -1.

Traduction en code:

```
double AD2 = joints[0] * joints[0];
double BD2 = joints[1] * joints[1];
double x = (AD2 - BD2 + Bx * Bx) / (2 * Bx);
double y2 = AD2 - x * x;
if(y2 < 0) return -1;
pos->tran.x = x;
pos->tran.y = sqrt(y2);
return 0;
```

15.3.2 Transformation inverse

La cinématique inverse est beaucoup plus simple dans notre exemple, de sorte que nous pouvons l'écrire directement:

AD=sqrt(x^2+y^2)

BD=sqrt($(Bx-x)^2+y^2$)

ou traduite en code:

```
double x2 = pos->tran.x * pos->tran.x;
double y2 = pos->tran.y * pos->tran.y;
joints[0] = sqrt(x2 + y2);
joints[1] = sqrt((Bx - pos->tran.x)*(Bx - pos->tran.x) + y2);
return 0;
```

15.4 Détails d'implémentation

Un module cinématique est implémenté comme un composant de HAL, et il est permis d'exporter ses pins et ses paramètres. Il consiste en quelques fonctions "C" (par opposition au fonctions de HAL):

int kinematicsForward(const double *joint, EmcPose *world, const KINEMATICS_FORWARD_FLA

Implémente la fonction cinématique avant.

int kinematicsInverse(const EmcPose * world, double *joints, const KINEMATICS_INVERSE_FLA

Implémente la fonction cinématique inverse.

KINEMATICS_TYPE kinematicsType(void)_
Retourne l'identificateur de type de la cinématique, typiquement KINEMATICS_BOTH.

int kinematicsHome(EmcPose *world, double *joint, KINEMATICS_FORWARD_FLAGS *fflags, KI

La fonction prise d'origine de la cinématique ajuste tous ses arguments à leur propre valeur à une position d'origine connue. Quand elle est appelée, cette position doit être ajustée, quand elle est connue, comme valeurs initiales, par exemple depuis un fichier INI. Si la prise d'origine de la cinématique peut accepter des points de départ arbitraires, ces valeurs initiales doivent être utilisées.

int rtapi_app_main(void) , void rtapi_app_exit(void)
Il s'agit des fonctions standards d'installation et de la désinstallation des modules RTAPI.

Quand ils sont contenus dans un seul fichier source, les modules de la cinématique peuvent être compilés et installés par comp. Voir la manpage comp(1) pour d'autres informations.

Chapter 16

Réglages d’une boucle PID

16.1 Régulation à PID

Un régulateur Proportionnel Intégral Dérivé (PID) est un organe de contrôle qui permet d’effectuer une régulation en boucle fermée d’un procédé.

Le régulateur compare une valeur mesurée sur le procédé avec une valeur de consigne. La différence entre ces deux valeurs (le signal d’erreur) est alors utilisée pour calculer une nouvelle valeur d’entrée du procédé tendant à réduire au maximum l’écart entre la mesure et la consigne (signal d’erreur le plus faible possible).

Contrairement aux algorithmes de régulation simples, le contrôle par PID peut ajuster les sorties du procédé, en fonction de l’amplitude du signal d’erreur, et en fonction du temps. Il donne des résultats plus précis et un contrôle plus stable. (Il est montré mathématiquement qu’une boucle PID donne un contrôle plus stable qu’un contrôle proportionnel seul et qu’il est plus précis que ce dernier qui laissera le procédé osciller).

16.1.1 Les bases du contrôle en boucle

Intuitivement, une boucle PID essaye d’automatiser ce que fait un opérateur muni d’un multimètre et d’un potentiomètre de contrôle. L’opérateur lit la mesure de sortie du procédé, affichée sur le multimètre et utilise le bouton du potentiomètre pour ajuster l’entrée du procédé (l’action) jusqu’à stabiliser la mesure de la sortie souhaitée, affichée sur le multimètre.

Un boucle de régulation est composée de trois parties:

1. La mesure, effectuée par un capteur connecté à un procédé, par exemple un codeur.
2. La décision, prise par les éléments du régulateur.
3. L’action sur le dispositif de sortie, par exemple: un moteur.

Quand le régulateur lit le capteur, il soustrait la valeur lue à la valeur de la consigne et ainsi, obtient l’«erreur de mesure». Il peut alors utiliser cette erreur pour calculer une correction à appliquer sur la variable d’entrée du procédé (l’action) de sorte que cette correction tende à supprimer l’erreur mesurée en sortie de procédé.

Dans une boucle PID, la correction à partir de l’erreur est calculée de trois façons: P) l’erreur de mesure courante est soustraite directement (effet proportionnel), I) l’erreur est intégrée pendant un laps de temps (effet intégral), D) l’erreur est dérivée pendant un laps de temps (effet dérivé).

Une régulation à PID peut être utilisée dans n'importe quel procédé pour contrôler une variable mesurable, en manipulant d'autres variables de ce procédé. Par exemple, elle peut être utilisée pour contrôler: température, pression, débit, composition chimique, vitesse et autres variables.

Dans certains systèmes de régulation, les régulateurs sont placés en série ou en parallèle. Dans ces cas, le régulateur maître produit les signaux utilisés par les régulateurs esclaves. Une situation courante dans le contrôle des moteurs, la régulation de vitesse, qui peut demander que la vitesse du moteur soit contrôlée par un régulateur esclave (souvent intégré dans le variateur de fréquence du moteur) recevant en entrée une valeur proportionnelle à la vitesse. Cette entrée de l'esclave est alors fournie par la sortie du régulateur maître, lequel reçoit la variable de consigne.

16.1.2 Théorie

Le PID représente les abréviations des trois actions qu'il utilise pour effectuer ses corrections, ce sont des ajouts d'un signal à un autre. Tous agissent sur la quantité régulée. Les actions aboutissent finalement à des soustractions de l'erreur de mesure, parce que le signal proportionnel est habituellement négatif.

16.1.2.1 Action Proportionnelle

Pour cette action, l'erreur est multipliée par la constante P (pour Proportionnel) qui est négative, puis ajoutée (soustraction de l'erreur de mesure) à la quantité régulée. P est valide uniquement sur la bande dans laquelle le signal de sortie du régulateur est proportionnel à l'erreur du système. Noter que si l'erreur de mesure est égale à zéro, la partie proportionnelle de la sortie du régulateur est également à zéro.

16.1.2.2 Action Intégrale

L'action intégrale fait intervenir la notion de temps. Elle tire profit du signal d'erreur passé qui est intégré (additionné) pendant un laps de temps, puis multiplié par la constante I (négative) ce qui en fait une moyenne, elle est enfin additionnée (soustraction de l'erreur de mesure) à la quantité régulée. La moyenne de l'erreur de mesure permet de trouver l'erreur moyenne entre la sortie du régulateur et la valeur de la consigne. Un système seulement proportionnel oscille en plus et en moins autour de la consigne du fait qu'en arrivant vers la consigne, l'erreur est à zéro, il n'enlève alors plus rien est dépasse la consigne, ou oscille et/ou se stabilise à une valeur trop basse ou trop élevée. L'addition sur l'entrée d'une proportion négative (soustraction) de l'erreur de mesure moyennée, permet toujours de réduire l'écart moyen entre la mesure en sortie et la consigne. Donc finalement, une boucle PI bien réglée verra sa sortie redescendre lentement à la valeur de la consigne.

16.1.2.3 Action Dérivée

L'action dérivée utilise aussi la notion de temps. Elle cherche à anticiper l'erreur future. La dérivée première (la pente de l'erreur) est calculée pour un laps de temps et multipliée par la constante (négative) D, puis elle est additionnée (soustraction de l'erreur de mesure) à la quantité régulée. L'action dérivée de la régulation fourni une réponse aux perturbations agissant sur le système. Plus important est le terme dérivé, plus rapide sera la réponse en sortie à une perturbation sur l'entrée.

Plus techniquement, une boucle PID peut être caractérisée comme un filtre appliqué sur un système complexe d'un domaine fréquentiel. C'est utilisé pour calculer si le système atteindra une valeur stable. Si les valeurs sont choisies incorrectement, le procédé entrera en oscillation et sa sortie n'atteindra jamais la consigne.

16.1.3 Réglage d'une boucle

Régler une boucle de régulation consiste à agir sur les paramètres des différentes actions (gain du proportionnel, gain de l'intégral, gain de la dérivée) sur des valeurs optimales pour obtenir la réponse désirée sur la sortie du procédé. Le comportement des procédés varient selon les applications lors d'un changement de consigne. Certains procédés ne permettent aucun dépassement de la consigne. D'autres doivent minimiser l'énergie nécessaire pour atteindre un nouveau point de consigne. Généralement la stabilité de la réponse est requise, le procédé ne doit pas osciller quels que soient les conditions du procédé et le point de consigne.

Régler une boucle est rendu plus compliqué si le temps de réponse du procédé est long; il peut prendre plusieurs minutes, voir plusieurs heures pour qu'une modification de consigne produise un effet stable. Certains procédés ne sont pas linéaires et les paramètres qui fonctionnent bien à pleine charge ne marchent plus lors du démarrage hors charge du procédé. Cette section décrit quelques méthodes manuelles traditionnelles pour régler ces boucles.

Il existe plusieurs méthodes pour régler une boucle PID. Le choix de la méthode dépendra en grande partie de la possibilité ou non de mettre la boucle «hors production» pour la mise au point ainsi que de la vitesse de réponse du système. Si le système peut être mis hors production, la meilleure méthode de réglage consiste souvent à soumettre le système à un changement de consigne, à mesurer la réponse en fonction du temps et à l'aide de cette réponse à déterminer les paramètres de la régulation.

16.1.3.1 Méthode simple

Si le système doit rester en production, une méthode de réglage consiste à mettre les valeurs I et D à zéro. Augmenter ensuite le gain P jusqu'à ce que la sortie de la boucle oscille. Puis, augmenter le gain I jusqu'à ce que cesse l'oscillation. Enfin, augmenter le gain D jusqu'à ce que la boucle soit suffisamment rapide pour atteindre rapidement sa consigne. Le réglage d'une boucle PID rapide provoque habituellement un léger dépassement de consigne pour avoir une montée plus rapide, mais certains systèmes ne le permettent pas.

Paramètre	**Temps de montée**	**Dépassement**	**Temps de réglage**	**S.S. Error**
P	Augmente	Augmente	Chang. faible	Diminue
I	Diminue	Augmente	Augmente	Eliminate
D	Chang. faible	Diminue	Diminue	Chang. faible

Effets de l'augmentation des paramètres

16.1.3.2 Méthode de Ziegler-Nichols

Une autre méthode de réglage est la méthode dite de "Ziegler-Nichols", introduite par John G. Ziegler et Nathaniel B. Nichols. Elle commence comme la méthode précédente: réglage des gains I et D à zéro et accroissement du gain P jusqu'à ce que la sortie du procédé commence à osciller. Noter alors le gain critique (K_c) et la période d'oscillation de la sortie (P_c). Ajuster alors les termes P, I et D de la boucle comme sur la table ci-dessous:

Type de régulation	**P**	**I**	**D**
P	$.5K_c$		
PI	$.45K_c$	$P_c/1.2$	
PID	$.6K_c$	$P_c/2$	$P_c/8$

Part VI

Glossary

A listing of terms and what they mean. Some terms have a general meaning and several additional meanings for users, installers, and developers.

Acme Screw
A type of lead-screw that uses an Acme thread form. Acme threads have somewhat lower friction and wear than simple triangular threads, but ball-screws are lower yet. Most manual machine tools use acme lead-screws.

Axis
One of the computer controlled movable parts of the machine. For a typical vertical mill, the table is the X axis, the saddle is the Y axis, and the quill or knee is the Z axis. Angular axes like rotary tables are referred to as A, B, and C. Additional linear axes relative to the tool are called U, V, and W respectively.

Axis(GUI)
One of the Graphical User Interfaces available to users of LinuxCNC. It features the modern use of menus and mouse buttons while automating and hiding some of the more traditional LinuxCNC controls. It is the only open-source interface that displays the entire tool path as soon as a file is opened.

Backlash
The amount of ”play” or lost motion that occurs when direction is reversed in a lead screw. or other mechanical motion driving system. It can result from nuts that are loose on leadscrews, slippage in belts, cable slack, ”wind-up” in rotary couplings, and other places where the mechanical system is not ”tight”. Backlash will result in inaccurate motion, or in the case of motion caused by external forces (think cutting tool pulling on the work piece) the result can be broken cutting tools. This can happen because of the sudden increase in chip load on the cutter as the work piece is pulled across the backlash distance by the cutting tool.

Backlash Compensation
Any technique that attempts to reduce the effect of backlash without actually removing it from the mechanical system. This is typically done in software in the controller. This can correct the final resting place of the part in motion but fails to solve problems related to direction changes while in motion (think circular interpolation) and motion that is caused when external forces (think cutting tool pulling on the work piece) are the source of the motion.

Ball Screw
A type of lead-screw that uses small hardened steel balls between the nut and screw to reduce friction. Ball-screws have very low friction and backlash, but are usually quite expensive.

Ball Nut
A special nut designed for use with a ball-screw. It contains an internal passage to re-circulate the balls from one end of the screw to the other.

CNC
Computer Numerical Control. The general term used to refer to computer control of machinery. Instead of a human operator turning cranks to move a cutting tool, CNC uses a computer and motors to move the tool, based on a part program.

Comp
A tool used to build, compile and install LinuxCNC HAL components.

Configuration(n)
A directory containing a set of configuration files. Custom configurations are normally saved in the users home/LinuxCNC/configs directory. These files include LinuxCNC’s traditional INI file and HAL files. A configuration may also contain several general files that describe tools, parameters, and NML connections.

Configuration(v)
The task of setting up LinuxCNC so that it matches the hardware on a machine tool.

Coordinate Measuring Machine
A Coordinate Measuring Machine is used to make many accurate measurements on parts. These machines can be used to create CAD data for parts where no drawings can be found, when a hand-made prototype needs to be digitized for moldmaking, or to check the accuracy of machined or molded parts.

Display units
The linear and angular units used for onscreen display.

DRO
A Digital Read Out is a system of position-measuring devices attached to the slides of a machine tool, which are connected to a numeric display showing the current location of the tool with respect to some reference position. DROs are very popular on hand-operated machine tools because they measure the true tool position without backlash, even if the machine has very loose Acme screws. Some DROs use linear quadrature encoders to pick up position information from the machine, and some use methods similar to a resolver which keeps rolling over.

EDM
EDM is a method of removing metal in hard or difficult to machine or tough metals, or where rotating tools would not be able to produce the desired shape in a cost-effective manner. An excellent example is rectangular punch dies, where sharp internal corners are desired. Milling operations can not give sharp internal corners with finite diameter tools. A wire EDM machine can make internal corners with a radius only slightly larger than the wire's radius. A sinker EDM can make internal corners with a radius only slightly larger than the radius on the corner of the sinking electrode.

LinuxCNC
The Enhanced Machine Controller. Initially a NIST project. LinuxCNC is able to run a wide range of motion devices.

LinuxCNCIO
The module within LinuxCNC that handles general purpose I/O, unrelated to the actual motion of the axes.

LinuxCNCMOT
The module within LinuxCNC that handles the actual motion of the cutting tool. It runs as a real-time program and directly controls the motors.

Encoder
A device to measure position. Usually a mechanical-optical device, which outputs a quadrature signal. The signal can be counted by special hardware, or directly by the parport with LinuxCNC.

Feed
Relatively slow, controlled motion of the tool used when making a cut.

Feed rate
The speed at which a cutting motion occurs. In auto or mdi mode, feed rate is commanded using an F word. F10 would mean ten machine units per minute.

Feedback
A method (e.g., quadrature encoder signals) by which LinuxCNC receives information about the position of motors

Feedrate Override
A manual, operator controlled change in the rate at which the tool moves while cutting. Often used to allow the operator to adjust for tools that are a little dull, or anything else that requires the feed rate to be "tweaked".

Floating Point Number
A number that has a decimal point. (12.300) In HAL it is known as float.

G-Code
The generic term used to refer to the most common part programming language. There are several dialects of G-code, LinuxCNC uses RS274/NGC.

GUI
Graphical User Interface.

General
A type of interface that allows communications between a computer and a human (in most cases) via the manipulation of icons and other elements (widgets) on a computer screen.

LinuxCNC
An application that presents a graphical screen to the machine operator allowing manipulation of the machine and the corresponding controlling program.

HAL
Hardware Abstraction Layer. At the highest level, it is simply a way to allow a number of building blocks to be loaded and interconnected to assemble a complex system. Many of the building blocks are drivers for hardware devices. However, HAL can do more than just configure hardware drivers.

Home
A specific location in the machine's work envelope that is used to make sure the computer and the actual machine both agree on the tool position.

ini file
A text file that contains most of the information that configures LinuxCNC for a particular machine

Instance
One can have an instance of a class or a particular object. The instance is the actual object created at runtime. In programmer jargon, the Lassie object is an instance of the Dog class.

Joint Coordinates
These specify the angles between the individual joints of the machine. See also Kinematics

Jog
Manually moving an axis of a machine. Jogging either moves the axis a fixed amount for each key-press, or moves the axis at a constant speed as long as you hold down the key. In manual mode, jog speed can be set from the graphical interface.

kernel-space
See real-time.

Kinematics
The position relationship between world coordinates and joint coordinates of a machine. There are two types of kinematics. Forward kinematics is used to calculate world coordinates from joint coordinates. Inverse kinematics is used for exactly the opposite purpose. Note that kinematics does not take into account, the forces, moments etc. on the machine. It is for positioning only.

Lead-screw
An screw that is rotated by a motor to move a table or other part of a machine. Lead-screws are usually either ball-screws or acme screws, although conventional triangular threaded screws may be used where accuracy and long life are not as important as low cost.

Machine units
The linear and angular units used for machine configuration. These units are specified and used in the ini file. HAL pins and parameters are also generally in machine units.

MDI
Manual Data Input. This is a mode of operation where the controller executes single lines of G-code as they are typed by the operator.

NIST
National Institute of Standards and Technology. An agency of the Department of Commerce in the United States.

Offsets
An arbitrary amount, added to the value of something to make it equal to some desired value. For example, gcode programs are often written around some convenient point, such as X0, Y0. Fixture offsets can be used to shift the actual execution point of that gcode program to properly fit the true location of the vice and jaws. Tool offsets can be used to shift the ”uncorrected” length of a tool to equal that tool’s actual length.

Part Program
A description of a part, in a language that the controller can understand. For LinuxCNC, that language is RS-274/NGC, commonly known as G-code.

Program Units
The linear and angular units used in a part program. The linear program units do not have to be the same as the linear machine units. See G20 and G21 for more information. The angular program units are always measured in degrees.

Python
General-purpose, very high-level programming language. Used in LinuxCNC for the Axis GUI, the Stepconf configuration tool, and several G-code programming scripts.

Rapid
Fast, possibly less precise motion of the tool, commonly used to move between cuts. If the tool meets the workpiece or the fixturing during a rapid, it is probably a bad thing!

Rapid rate
The speed at which a rapid motion occurs. In auto or mdi mode, rapid rate is usually the maximum speed of the machine. It is often desirable to limit the rapid rate when testing a g-code program for the first time.

Real-time
Software that is intended to meet very strict timing deadlines. Under Linux, in order to meet these requirements it is necessary to install RTAI or RTLINUX and build the software to run in those special environments. For this reason real-time software runs in kernel-space.

RTAI
Real Time Application Interface, see https://www.rtai.org/, one of two real-time extensions for Linux that LinuxCNC can use to achieve real-time performance.

RTLINUX
See http://www.rtlinux.org, one of two real-time extensions for Linux that LinuxCNC can use to achieve real-time performance.

RTAPI
A portable interface to real-time operating systems including RTAI and RTLINUX

RS-274/NGC
The formal name for the language used by LinuxCNC part programs.

Servo Motor
Generally, any motor that is used with error-sensing feedback to correct the position of an actuator. Also, a motor which is specially-designed to provide improved performance in such applications.

Servo Loop
A control loop used to control position or velocity of an motor equipped with a feedback device.

Signed Integer
A whole number that can have a positive or negative sign. In HAL it is known as s32. (A signed 32-bit integer has a usable range of -2,147,483,647 to +2,147,483,647.)

Spindle
The part of a machine tool that spins to do the cutting. On a mill or drill, the spindle holds the cutting tool. On a lathe, the spindle holds the workpiece.

Spindle Speed Override
A manual, operator controlled change in the rate at which the tool rotates while cutting. Often used to allow the operator to adjust for chatter caused by the cutter's teeth. Spindle Speed Override assumes that the LinuxCNC software has been configured to control spindle speed.

Stepconf
An LinuxCNC configuration wizard. It is able to handle many step-and-direction motion command based machines. It writes a full configuration after the user answers a few questions about the computer and machine that LinuxCNC is to run on.

Stepper Motor
A type of motor that turns in fixed steps. By counting steps, it is possible to determine how far the motor has turned. If the load exceeds the torque capability of the motor, it will skip one or more steps, causing position errors.

TASK
The module within LinuxCNC that coordinates the overall execution and interprets the part program.

Tcl/Tk
A scripting language and graphical widget toolkit with which several of LinuxCNCs GUIs and selection wizards were written.

Traverse Move
A move in a straight line from the start point to the end point.

Units
See "Machine Units", "Display Units", or "Program Units".

Unsigned Integer
A whole number that has no sign. In HAL it is known as u32. (An unsigned 32-bit integer has a usable range of zero to 4,294,967,296.)

World Coordinates
This is the absolute frame of reference. It gives coordinates in terms of a fixed reference frame that is attached to some point (generally the base) of the machine tool.

Part VII

Legal Section

Ce document n’est pas traduit en raison de la complexité de la validation des documents juridiques. Vous trouverez sur ce site plus d’information: http://www.gnu.org/licenses/licenses.fr.html

Chapter 17

Copyright Terms

Chapter 18

GNU Free Documentation License

GNU Free Documentation License Version 1.1, March 2000

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document ”free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of ”copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The ”Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as ”you”.

A ”Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A ”Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The ”Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The ”Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A ”Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not ”Transparent” is called ”Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The ”Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, ”Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version: A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission. B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five). C. State on the Title page the name of the publisher of the Modified Version, as the publisher. D. Preserve all the copyright notices of the Document. E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices. F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below. G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice. H. Include an unaltered copy of this License. I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence. J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission. K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein. L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles. M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version. N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name

but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled ”History” in the various original documents, forming one section entitled ”History”; likewise combine any sections entitled ”Acknowledgements”, and any sections entitled ”Dedications”. You must delete all sections entitled ”Endorsements.”

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an ”aggregate”, and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document’s Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License ”or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled ”GNU Free Documentation License”.

If you have no Invariant Sections, write ”with no Invariant Sections” instead of saying which ones are invariant. If you have no Front-Cover Texts, write ”no Front-Cover Texts” instead of ”Front-Cover Texts being LIST”; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Chapter 19

Index