

Homing Configuration

原点復帰の設定

目次 Table of Contents

- [1. 概要 Overview](#)
- [2. 原点復帰シーケンス Homing Sequence](#)
- [3. 設定 Configuration](#)
 - [3.1. HOME_SEARCH_VEL](#)
 - [3.2. HOME_LATCH_VEL](#)
 - [3.3. HOME_FINAL_VEL](#)
 - [3.4. HOME_IGNORE_LIMITS](#)
 - [3.5. HOME_USE_INDEX](#)
 - [3.6. HOME_OFFSET](#)
 - [3.7. HOME](#)
 - [3.8. HOME_IS_SHARED](#)
 - [3.9. HOME_SEQUENCE](#)
 - [3.10. VOLATILE_HOME](#)
 - [3.11. LOCKING_INDEXER](#)
 - [3.12. Immediate Homing](#)

※この日本語版は非公式なものです。内容の完全性は保証されません。
気になる箇所はlinuxcnc.org公式のドキュメントを参照して確認してください。
(ドキュメントのフッタにオリジナルのURLがあります)

1. はじめに Overview

原点復帰はとても簡単な事にみえます—各軸をわかっている場所にただ動かして、LinuxCNCの内部変数を設定し直すだけ。しかし、機械が違えば要求も変化しますし、原点復帰は、実は大変複雑なものです。

2. 原点復帰シーケンス Homing Sequence

ここに「*SEARCH_VEL*」と「*LATCH_VEL*」と「*USE_INDEX*」との組み合わせで分類できる4種類の原点復帰シーケンスがあります。補助的な設定パラメータの様子はその下の表に示してあります。それぞれの設定パラメータがどのように動作するかの詳細な説明は次のセクションをみてください。

図中の用語

Home switch releases: 原点スイッチが離れたところ

Home switch trips: 原点スイッチが作動したところ

Overshoot: 減速中の行き過ぎ

そのほかは次のセクションに解説されています

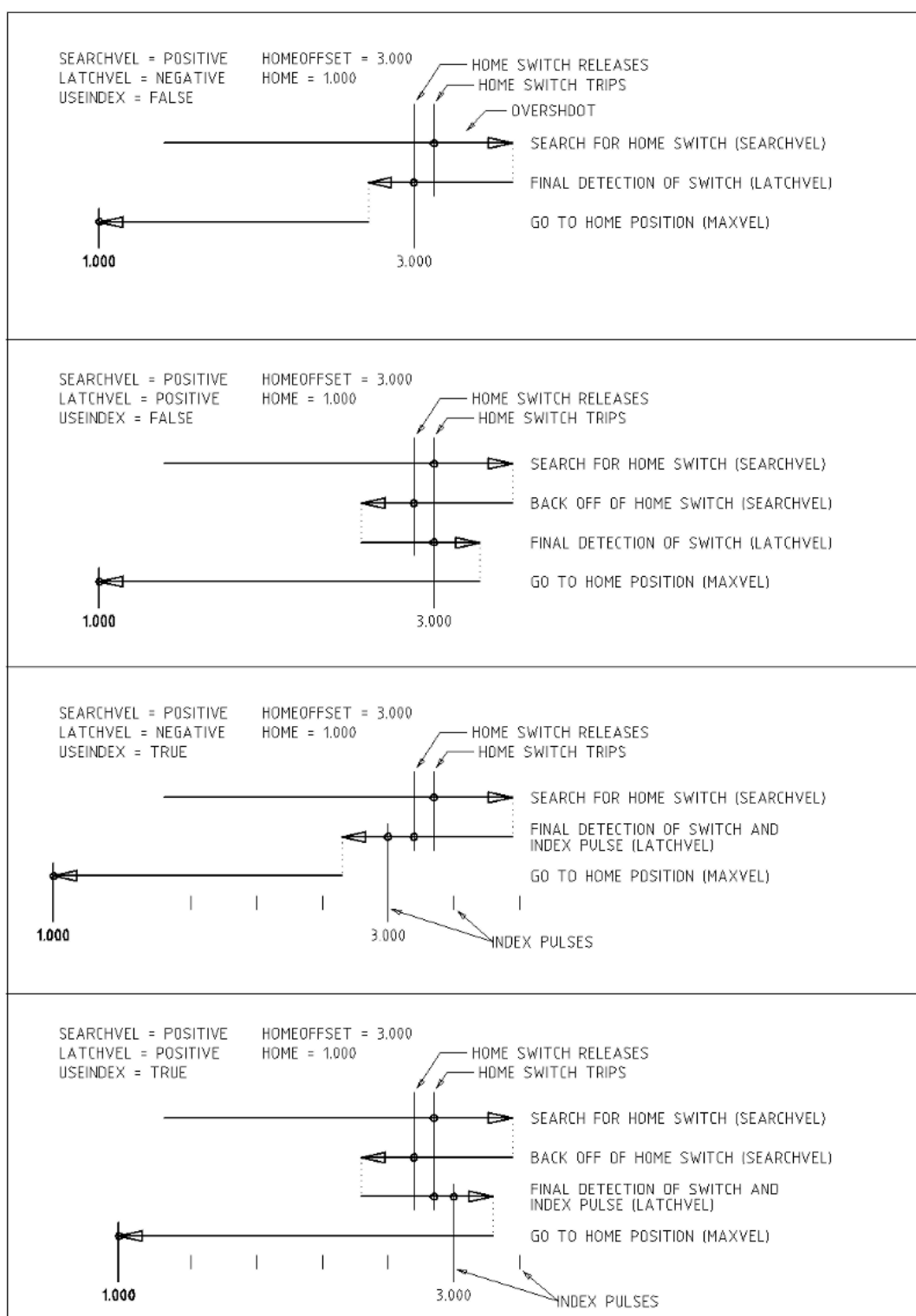


Figure 1. 4種の原点復帰シーケンス Homing Sequences

3. 設定 Configuration

以下の表のように原点シーケンスの具体的な挙動を定めています。これらの値はINIファイルの「*AXIS*」セクションで定義されます。

Homing Type	SEARCH_VEL	LATCH_VEL	USE_INDEX
Immediate	0	0	NO
Index-only	0	nonzero	YES
Switch-only	nonzero	nonzero	NO
Switch and Index	nonzero	nonzero	YES

注:

ここに示す組み合わせ以外ではエラーとなる可能性があります。

3.1. 原点サーチ速度 HOME_SEARCH_VEL

デフォルトの値は「0」です。「0」の値はLinuxCNCに原点が存在しないとみなし、原点サーチをスキップして次の動作に移ります。

「HOME_SEARCH_VEL」がゼロ以外の数値であった時、LinuxCNCは原点センサが存在しているとみなし(復帰動作を開始し)ます。まずはじめに、既に原点スイッチが作動していないかどうかを確認にいきます。もし作動していた場合、「HOME_SEARCH_VEL」の速度で一旦原点スイッチから後退します。このときの進行方向は「HOME_SEARCH_VEL」で定めた向きと反対になります。それから、「HOME_SEARCH_VEL」の符号で定めた向きと速度(値は絶対値)で動き原点スイッチを探し始めます。原点センサを発見すると、可能な限り早く軸を停止させますが、反応速度や減速度の関係で常に幾らかのオーバーシュート、行き過ぎが発生します。オーバーシュートの大半はサーチの速度によって引き起こされます。あまりにも速すぎると、原点センサを発見したあととまりきれずにリミットセンサや、ひどい時には軸端に衝突するほど行き過ぎることもありえます。一方で、「HOME_SEARCH_VEL」が遅すぎると、原点復帰動作にとっても時間がかかります。

3.2. ラッチ速度 HOME_LATCH_VEL

デフォルトの値は「0」です。

最終的な原点スイッチのより精密な原点位置(あれば)を決定し、そこからインデックス送り(設定していれば)するための方向と速度を指定します。通常は精度を最重要視しますのでサーチ速度「HOME_SEARCH_VEL」より遅い値にします。

「HOME_SEARCH_VEL」と「HOME_LATCH_VEL」が同じ符号のとき、ラッチング完了までサーチと同じ方向に移動します(その場合、LinuxCNCは、まず後退して原点スイッチから離れてから、ラッチ速度でその方向に移動し直します)。「HOME_SEARCH_VEL」と「HOME_LATCH_VEL」とが逆の符号のときは、ラッチ動作はサーチ動作と反対方向になります。これはLinuxCNCが原点スイッチのoffを確認したあと1パルス分だけラッチしたところで停止しようとするという意味です。

「HOME_SEARCH_VEL」がゼロ(物理的な原点スイッチがないことを意味する)かつ

「HOME_LATCH_VEL」がゼロ以外の数値のとき、LinuxCNCはインデックス送り動作に進みます。

「HOME_SEARCH_VEL」がゼロでなく「HOME_LATCH_VEL」がゼロの時はエラーとなり原点復帰動作は失敗します。

3.3. ファイナル速度 HOME_FINAL_VEL

センサで決定した原点位置から「HOME_OFFSET」まで移動するときの速度を指定します。

「HOME_FINAL_VEL」がINIファイルに見つからない時は、[AXIS]セクションに記載されているその軸の最高移動速度がかわりに使用されます。この値は必ず正の値でなければなりません。

3.4. 原点復帰時リミット無視 HOME_IGNORE_LIMITS

「YES」か「NO」のいずれかが入力できます。デフォルトは「NO」です。

このフラグはこの軸の原点復帰中にリミットスイッチからの入力を無視するかどうかを決めます。「YES」としてもこの軸以外のリミットスイッチからの信号は有効に作用します。もし単独の原点を持たない装置なら、この変数を「YES」にし、リミットスイッチからの信号をHALピンの原点スイッチ入力にも分岐させてください。LinuxCNCは、この軸の原点復帰シーケンスの間はこの軸のリミットスイッチ入力を無視します。

ひとつの入力にすべてのリミットスイッチと原点センサの信号をまとめるために、すべての軸のリミットセンサをブロックしなければならないときは、HALの機能では原点復帰できないので、1度に1軸ずつ個別に原点復帰してください。

3.5. インデックスパルスを使う HOME_USE_INDEX

インデックスパルスがあるかどうかを指定します。デフォルトは「NO」です。

フラグの値が真なら(「HOME_USE_INDEX = YES」)、LinuxCNCインデックスパルスの立ち上がりエッジでラッチします。偽なら、LinuxCNC原点センサの立ち上がり(または立ち下がり)エッジでラッチします。(「HOME_SEARCH_VEL」と「HOME_LATCH_VEL」の符号の組み合わせにより変化します)。

3.6. 原点オフセット HOME_OFFSET

デフォルトの値は「0」です。

原点センサやインデックスパルスの位置の座標データを入力します。

原点スイッチやインデックスパルスがラッチしたところと実際の軸のゼロ点がずれているときに距離補正するのにも使えます。インデックスパルス検知後、LinuxCNCは現在位置の座標を「HOME_OFFSET」にセットします。

3.7. ホーム位置 HOME

デフォルトの値は「0」です。

原点シーケンス完了時に軸が移動する場所です。

インデックスパルスを検知し現在位置の座標を「HOME_OFFSET」にセットした後、LinuxCNC原点シーケンスの最終工程として「HOME」に示す座標まで軸を移動させます。この変数の値を「HOME_OFFSET」と同じ値に指定したときでさえ、軸はラッチ位置で停止するときわずかにオーバーシュートしていて位置にずれがあることに注意してください。したがってこのとき、毎回ごくわずかにだけ移動することになります。(「HOME_SEARCH_VEL」がゼロでサーチ動作もラッチもスキップされていない限りは)。この最終動作は軸の最高速度で移動します。軸はすでに機械原点を取っていて、動作中に軸端に衝突するリスクがなくなったので、最高速での送りが原点復帰シーケンスの最終段では早くて合理的です。^[1]

3.8. シェアード原点 HOME_IS_SHARED

軸ごとに分離された原点スイッチ入力ピンはないが、モーメンタリタイプの原点スイッチは装備されていて同じ入力ピンにまとめて配線されている構成のときは、この変数の値を「1」にすると共有された原典入力品が既に作動しているとき原点復帰のスタートを防ぎます。この変数の値を「0」とすると原点センサが既に閉じていても原点復帰を許可します。

3.9. 原点復帰シーケンス HOME_SEQUENCE

「HOME_ALL」で複数軸の原点復帰シーケンスを定義するとき、復帰順を強制したい場合に使用します(例えば、X軸はZ軸の復帰が終わってからにしたいとき)。ある軸が復帰するには、それより小さい「HOME_SEQUENCE」の値を持つ軸すべてが原点復帰し「HOME_OFFSET」にいないければなりません。もし複数の軸の「HOME_SEQUENCE」が同じ値となっていたら、それらは同時に復帰動作します。「HOME_SEQUENCE」を「-1」としたり、指定がない場合は、その軸は「HOME_ALL」シーケンスから除外されます。「HOME_SEQUENCE」の値は0はじまりで、使用できない数はありません。

3.10. 原点の自動解除 **VOLATILE_HOME**

「*TRUE*」のとき、この軸は機械の電源を切ると原点をクリアします。この設定は軸の駆動が切られると保持力がなくなって位置が維持できない箇所に適しています。いくつかのステッピングモータドライバ、とくにマイクロステップ機能を利用するドライバで必要となる場合があります。

3.11. インデックステーブル ロック **LOCKING_INDEXER**

原点復帰しようとする軸がロック機能付きの割り出しテーブルのとき、この変数を「？」にセットすると原点シーケンス前にロックを解除し、完了後にロックを掛けます。

3.12. 即時原点化 **Immediate Homing**

軸に原点センサが無かったり、回転軸であるためにどこを原点としても問題なく、「**AXIS**」UIの「Home All」ボタンをクリックしたとき現在位置を原点としたい場合、iniファイルの設定を以下のようにする必要があります。

1. `SEARCH_VEL = 0`
2. `LATCH_VEL = 0`
3. `USE_INDEX = NO`
4. `HOME_SEQUENCE = 0`

1. 「*home_offset*」と「*home*」の違いは、「*home_offset*」は、見つかった原点位置に「*home_offset*」の値を適用することで機械の定規の目盛の位置を先に確立し、それから「*home*」はその目盛のどこに最終的に移動したら良いのかを示しています。

Last updated 2013-11-04 00:28:04 MST

日本語版最終更新:2013-12-14 JST