Entwickler-Handbuch
V2.9.3-2099-g9f5a1c8f50, 09 Oct 2023

Inhaltsverzeichnis

Kapitel 1

Einführung

Dieses Handbuch ist noch in Arbeit. Wenn Sie beim Schreiben, Redigieren oder bei der grafischen Aufbereitung helfen können, wenden Sie sich bitte an ein Mitglied des Redaktionsteams oder schreiben Sie eine E-Mail (bevorzugt auf Englisch, aber nicht zwingend, es findet sich jemand) an emc-users@lists.sourceforge.net.

Kapitel 2

HAL General Reference

2.1 HAL Entity Names

Alle HAL-Entitäten sind über ihre Namen zugänglich und manipulierbar, daher ist die Dokumentation der Namen von Pins, Signalen, Parametern usw. sehr wichtig. Namen in HAL haben eine maximale Länge von 41 Zeichen (wie durch HAL_NAME_LEN in hal.h definiert). Viele Namen werden in der allgemeinen Form dargestellt, mit formatiertem Text <*wie-dies*>, der Felder mit verschiedenen Werten darstellt.

Wenn Pins, Signale oder Parameter zum ersten Mal beschrieben werden, wird ihrem Namen der Typ in Klammern vorangestellt (*float*), gefolgt von einer kurzen Beschreibung. Typische Pin-Definitionen sehen wie diese Beispiele aus:

(bit) parport.<portnum>.pin-<pinnum>-in
: Der HAL-Pin, der mit dem physischen Eingangspin <*pinnum*> des db25-Anschlusses verbunden ist.

(float) pid.<loopnum>.output
: Der PID-Regelkreis Ausgang

Gelegentlich kann eine abgekürzte Version des Namens verwendet werden, z. B. könnte der zweite Pin oben einfach mit *.output* aufgerufen werden, wenn dies möglich ist, ohne dass es zu Verwechslungen kommt.

2.2 HAL General Naming Conventions

Einheitliche Namenskonventionen würden die Verwendung von HAL wesentlich erleichtern. Wenn zum Beispiel jeder Encoder-Treiber den gleichen Satz von Pins zur Verfügung stellen und diese auch gleich benennen würde, dann wäre es einfach, von einem Encoder-Treiber zu einem anderen zu wechseln. Leider ist HAL, wie viele Open-Source-Projekte, eine Kombination aus Dingen, die entworfen wurden, und Dingen, die sich einfach entwickelt haben. Infolgedessen gibt es viele Inkonsistenzen. Dieser Abschnitt versucht, dieses Problem zu lösen, indem er einige Konventionen definiert, aber es wird wahrscheinlich noch eine Weile dauern, bis alle Module so konvertiert sind, dass sie ihnen folgen.

Halcmd und andere Low-Level-HAL-Utilities behandeln HAL-Namen als einzelne Einheiten ohne interne Struktur. Die meisten Module haben jedoch eine implizite Struktur. Ein Board bietet beispielsweise mehrere Funktionsblöcke, jeder Block kann mehrere Kanäle haben, und jeder Kanal hat einen oder mehrere Pins. Dies führt zu einer Struktur, die einem Verzeichnisbaum ähnelt. Auch wenn halcmd die

Baumstruktur nicht erkennt, kann es durch die richtige Wahl der Namenskonventionen verwandte Elemente zusammenfassen (da es die Namen sortiert). Darüber hinaus können Werkzeuge auf höherer Ebene so gestaltet werden, dass sie eine solche Struktur erkennen, wenn die Namen die notwendigen Informationen liefern. Um dies zu erreichen, sollten alle HAL-Komponenten diese Regeln befolgen:

- Punkte (”.”) trennen die einzelnen Ebenen der Hierarchie. Dies ist vergleichbar mit dem Schrägstrich (”/”) in einem Dateinamen.
- Bindestriche („-“) trennen Wörter oder Felder auf derselben Hierarchieebene.
- HAL-Komponenten sollten keine Unterstriche oder ”MixedCase” verwenden. [1]
- Verwenden Sie in Namen nur Kleinbuchstaben und Zahlen.

2.3 Hardware Driver Naming Conventions

Anmerkung
Die meisten Treiber folgen diesen Konventionen in Version 2.0 nicht. Dieses Kapitel ist eher ein Leitfaden für zukünftige Entwicklungen.

2.3.1 Pins/Parameters names

Hardwaretreiber sollten fünf Felder (auf drei Ebenen) verwenden, um einen PIN- oder Parameternamen wie folgt zu bilden:

```
<Gerätename>.<Gerätenummer>.<E/A-Typ>.<Kanal-Nummer>.<Spezifischer Name>
(engl. <device-name>.<device-num>.<io-type>.<chan-num>.<specific-name>)
```

Die einzelnen Felder sind:

_<Gerätename> _ (engl. device-name)
Das Gerät, mit dem der Treiber arbeiten soll. In den meisten Fällen handelt es sich dabei um eine Schnittstellenkarte, aber es gibt auch andere Möglichkeiten.

***<Gerät-Nr.>* (engl. device-num)**
Es ist möglich, mehr als eine Servokarte, eine parallele Schnittstelle oder ein anderes Hardwaregerät in einem Computer zu installieren. Die Gerätenummer identifiziert ein bestimmtes Gerät. Die Gerätenummern beginnen bei 0 und werden inkrementiert.

***<E/A-Typ>* (engl. io-type)**
Die meisten Geräte bieten mehr als eine Art von E/A. Selbst der einfache Parallelport hat sowohl digitale Eingänge als auch digitale Ausgänge. Komplexere Karten können digitale Ein- und Ausgänge, Encoderzähler, PWM- oder Schrittimpulsgeneratoren, Analog-Digital-Wandler, Digital-Analog-Wandler oder andere einzigartige Funktionen haben. Der E/A-Typ wird verwendet, um die Art der E/A zu identifizieren, mit der ein Pin oder Parameter verbunden ist. Idealerweise sollten Treiber, die denselben E/A-Typ implementieren, auch wenn sie für sehr unterschiedliche Geräte bestimmt sind, einen konsistenten Satz von Pins und Parametern und ein identisches Verhalten aufweisen. Beispielsweise sollten sich alle digitalen Eingänge von der HAL aus gesehen gleich verhalten, unabhängig vom Gerät.

[1] Die unterstrichenen Zeichen wurden entfernt, aber es gibt immer noch einige Fälle, in denen die Mischung nicht stimmt, zum Beispiel *pid.0.Pgain* anstelle von *pid.0.p-gain*.

<*Kanal-Nummer*> (engl. channel-number, kurz chan-num)
Praktisch jedes E/A-Gerät verfügt über mehrere Kanäle, und die Kanalnummer identifiziert einen dieser Kanäle. Wie die Gerätenummern beginnen auch die Kanalnummern bei Null und erhöhen sich. Fußnote:[Eine Ausnahme von der Regel ”Kanalnummern beginnen bei Null” ist der Parallelport. Seine HAL-Pins sind mit der entsprechenden Pin-Nummer auf dem DB-25-Stecker nummeriert. Dies ist praktisch für die Verdrahtung, aber inkonsistent mit anderen Treibern. Es ist umstritten, ob dies ein Fehler oder eine Funktion ist]. Wenn mehr als ein Gerät installiert ist, beginnen die Kanalnummern auf zusätzlichen Geräten wieder bei Null. Wenn es möglich ist, eine Kanalnummer größer als 9 zu haben, dann sollten die Kanalnummern zweistellig sein, mit einer führenden Null bei Nummern kleiner als 10, um die Sortierreihenfolge zu erhalten. Einige Module haben Pins und/oder Parameter, die mehr als einen Kanal betreffen. Ein PWM-Generator kann z. B. vier Kanäle mit vier unabhängigen ”Tastverhältnis”-Eingängen haben, aber einen ”Frequenz”-Parameter, der alle vier Kanäle steuert (aufgrund von Hardwarebeschränkungen). Der Frequenzparameter sollte ”0-3” als Kanalnummer verwenden.

<*Spezifischer-Name*>
Einem einzelnen E/A-Kanal kann nur ein einziger HAL-Pin zugeordnet sein, aber die meisten haben mehr als einen. Ein digitaler Eingang hat beispielsweise zwei Pins, von denen einer den Zustand des physischen Pins und der andere den gleichen Zustand invertiert darstellt. So kann der Konfigurator zwischen aktiven High und aktiven Low-Eingängen wählen. Für die meisten E/A-Typen gibt es einen Standardsatz von Pins und Parametern (die so genannte ”kanonische Schnittstelle”), die der Treiber implementieren sollte. Die kanonischen Schnittstellen sind im Kapitel Kanonische Geräte-Schnittstellen beschrieben.

Beispiele

motenc.0.encoder.2.position
Der Positionsausgang des dritten Encoderkanals auf der ersten Motenc-Platine.

stg.0.din.03.in
Der Zustand des vierten digitalen Eingangs auf der ersten Servo-to-Go-Karte.

ppmc.0.pwm.00-03.frequency
Die für die PWM-Kanäle 0 bis 3 auf der ersten Pico Systems ppmc-Karte verwendete Trägerfrequenz.

2.3.2 Function Names

Hardwaretreiber haben in der Regel nur zwei Arten von HAL-Funktionen: solche, die aus der Hardware lesen und HAL-Pins aktualisieren, und solche, die mit Daten von HAL-Pins in die Hardware schreiben. Sie sollten wie folgt benannt werden:

```
<device-name>-<device-num>.<io-type>-<chan-num-range>.read|write
```

<Gerätename> (engl. device-name)
Das gleiche wie für Pins und Parameter.

<Gerät-Nr.> (engl. device-num)
Das spezifische Gerät, auf das die Funktion zugreift.

<E/A-Typ> (engl. io-type)
Optional. Eine Funktion kann auf alle E/A auf einer Karte zugreifen oder nur auf einen bestimmten Typ. So kann es beispielsweise unabhängige Funktionen zum Lesen von Encoderzählern und

zum Lesen von digitalen E/A geben. Für solche unabhängigen Funktionen gibt das Feld <io-type> die Art der E/A an, auf die sie zugreifen. Wenn eine einzige Funktion alle von der Karte bereitgestellten E/A liest, wird <io-type> nicht verwendet. [2]

<chan-num-range>
Optional. Wird nur verwendet, wenn die <io-type> E/A in Gruppen unterteilt ist und verschiedene Funktionen darauf zugreifen.

read|write
Gibt an, ob die Funktion die Hardware liest (engl. read) oder in sie schreibt (engl. write).

Beispiele

motenc.0.encoder.read
Liest alle Encoder auf der ersten Motenc-Platine aus.

generic8255.0.din.09-15.read
Liest den zweiten 8-Bit-Port auf der ersten generischen 8255-basierten digitalen E/A-Karte.

ppmc.0.write
Schreibt alle Ausgänge (Schrittgeneratoren, PWM, DACs und Digital) auf die erste Pico Systems ppmc-Karte.

[2] Hinweis für Treiberprogrammierer: Implementieren Sie KEINE separaten Funktionen für verschiedene E/A-Typen, es sei denn, sie sind unterbrechbar und können in unabhängigen Threads arbeiten. Wenn die Unterbrechung eines Encoder-Lesevorgangs, das Lesen digitaler Eingänge und die anschließende Wiederaufnahme des Encoder-Lesevorgangs zu Problemen führen, dann implementieren Sie eine einzige Funktion, die alles erledigt.

Kapitel 3

Code Anmerkungen

3.1 Zielgruppe

Dieses Dokument ist eine Sammlung von Notizen über die Interna von LinuxCNC. Es ist in erster Linie von Interesse für Entwickler, aber viele der Informationen hier kann auch von Interesse für Systemintegratoren und andere, die einfach nur neugierig, wie LinuxCNC funktioniert sind. Viele dieser Informationen ist jetzt veraltet und wurde nie für die Richtigkeit überprüft.

3.2 Organisation

Es wird ein Kapitel für jede der Hauptkomponenten von LinuxCNC geben. Weitere Kapitel erklären, wie die Komponenten zusammenarbeiten. Dieses Dokument wird derzeit noch stark überarbeitet. Entprechend mag sich sein Layout/Zusammenstellung in der Zukunft noch ändern.

3.3 Begriffe und Definitionen

- *AXIS* - Eine Achse ist einer der neun Freiheitsgrade, die eine Werkzeugposition im dreidimensionalen kartesischen Raum definieren. Diese neun Achsen werden als X, Y, Z, A, B, C, U, V und W bezeichnet. Die linearen orthogonalen Koordinaten X, Y und Z bestimmen, wo die Werkzeugspitze positioniert ist. Die Winkelkoordinaten A, B und C bestimmen die Ausrichtung des Werkzeugs. Ein zweiter Satz linearer orthogonaler Koordinaten U, V und W ermöglicht die Bewegung des Werkzeugs (in der Regel für Schneidvorgänge) relativ zu den zuvor versetzten und gedrehten Achsen. Leider wird der Begriff ”Achse” manchmal auch für einen Freiheitsgrad der Maschine selbst verwendet, z. B. für den Schlitten, den Tisch oder die Pinole einer Bridgeport-Fräsmaschine. Bei einer Bridgeport-Maschine führt dies nicht zu Verwirrung, da die Bewegung des Tisches direkt einer Bewegung entlang der X-Achse entspricht. Die Schulter- und Ellbogengelenke eines Roboterarms und die Linearantriebe eines Hexapods entsprechen jedoch keiner Bewegung entlang einer kartesischen Achse, und im Allgemeinen ist es wichtig, zwischen den kartesischen Achsen und den Freiheitsgraden der Maschine zu unterscheiden. In diesem Dokument werden letztere als *Gelenke* und nicht als Achsen bezeichnet. Die grafischen Benutzeroberflächen und einige andere Teile des Quellcodes folgen dieser Unterscheidung vielleicht nicht immer, aber die Interna des ”Motion Controllers” schon.
- *GELENK* (engl. *JOINT*)- Ein Gelenk ist eines der beweglichen Teile der Maschine. Gelenke unterscheiden sich von Achsen, obwohl die beiden Begriffe manchmal (fälschlicherweise) für dieselbe Sache verwendet werden. In LinuxCNC ist ein Gelenk einer physikalischen Sache, die bewegt werden kann, nicht eine Koordinate im Raum. Zum Beispiel sind die Pinole, das Knie, der Sattel und der

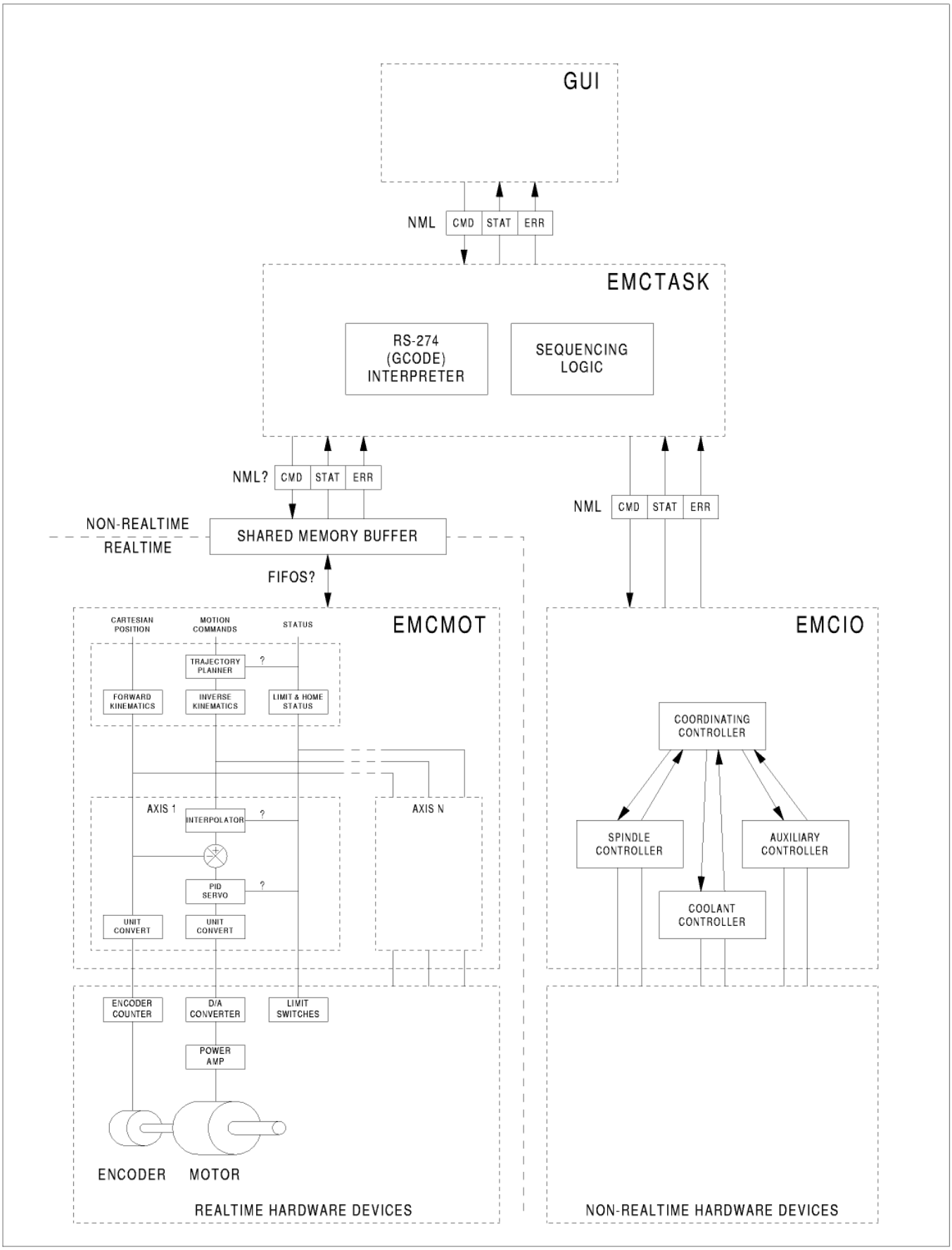
Tisch einer Bridgeport-Fräse alles Gelenke. Die Schulter, der Ellbogen und das Handgelenk eines Roboterarms sind Gelenke, ebenso wie die Linearaktuatoren eines Hexapods. Jedem Gelenk ist ein Motor oder Aktuator zugeordnet. Gelenke entsprechen nicht unbedingt den X-, Y- und Z-Achsen, obwohl dies bei Maschinen mit trivialer Kinematik der Fall sein kann. Selbst bei diesen Maschinen sind die Position des Gelenks und die Position der Achsen grundlegend verschieden. In diesem Dokument werden die Begriffe *Gelenk* und *Achse* sorgfältig verwendet, um ihre unterschiedlichen Bedeutungen zu berücksichtigen. Leider trifft das nicht unbedingt überall zu. Insbesondere GUIs für Maschinen mit trivialer Kinematik können die Unterscheidung zwischen Gelenken und Achsen beschönigen oder ganz ausblenden. Darüber hinaus wird in der .ini-Datei der Begriff *Achse* für Daten verwendet, die eher als Gelenkdaten zu bezeichnen sind, wie z. B. Eingangs- und Ausgangsskalierung usw.

Anmerkung
Diese Unterscheidung wird seit Version 2.8 von LinuxCNC vorgenommen. Die .ini-Datei bekam einen neuen Abschnitt [JOINT_<num>]. Viele der Parameter, die zuvor ordnungsgemäß zu dem [AXIS_<letter>] Abschnitt sind jetzt in dem neuen Abschnitt. Andere Abschnitte, wie [KINS], erhalten ebenfalls neue Parameter. Es wurde ein Update-Skript bereitgestellt, um alte .ini-Dateien in die neue Achsen/Gelenke-Konfiguration umzuwandeln.

- *POSE* - Eine Pose ist eine vollständig spezifizierte Position im kartesischen 3D-Raum. In der LinuxCNC Motion Controller, wenn wir uns auf eine Pose beziehen, meinen wir eine EmcPose Struktur, welche die sechs linearen Koordinaten (X, Y, Z, U, V, und W) und die drei Winkelkoordinaten (A, B, und C) enthält.
- *coord*, oder koordinierter Modus, bedeutet, dass alle Gelenke synchronisiert sind und sich zusammen bewegen, wie vom übergeordneten Code angewiesen. Dies ist der normale Modus für die Bearbeitung. Im koordinierten Modus wird davon ausgegangen, dass die Befehle im kartesischen Referenzrahmen gegeben werden. Wenn die Maschine nicht kartesisch ist, werden die Befehle von der Kinematik übersetzt, um jedes Gelenk wie erforderlich in den Gelenkraum zu bewegen.
- *frei* bedeutet, dass die Befehle im Gelenkraum interpretiert werden. Dieser wird verwendet, um einzelne Gelenke manuell zu bewegen (engl. Maschinisten-Slang: jogging), obwohl er nicht verhindert, dass mehrere Gelenke gleichzeitig bewegt werden (glaube ich). Die Referenzfahrt wird ebenfalls im freien Modus durchgeführt; Maschinen mit nicht-trivialer Kinematik müssen erst referenziert werden, bevor sie in den Koordinaten- oder Teleop-Modus wechseln können.
- *teleop* ist der Modus, den Sie wahrscheinlich brauchen, wenn Sie mit einem Hexapod joggen. Die vom Motion Controller implementierten Jog-Befehle sind Gelenk-Jogs, die im freien Modus funktionieren. Aber wenn Sie einen Hexapod oder eine ähnliche Maschine insbesondere entlang einer kartesischen Achse bewegen wollen, müssen Sie mehr als ein Gelenk betätigen. Dafür ist *teleop* gedacht.

3.4 Architekturübersicht

Die LinuxCNC-Architektur besteht aus vier Komponenten: einem Motion-Controller (EMCMOT), einem diskreten IO-Controller (EMCIO), einem Task-Executor, der diese koordiniert (EMCTASK) und mehreren textbasierten und grafischen Benutzerschnittstellen. Jede dieser Schnittstellen wird in diesem Dokument beschrieben, sowohl aus der Sicht des Designs als auch aus der Sicht der Entwickler (wo findet man benötigte Daten, wie kann man Dinge einfach erweitern/verändern, usw.).


GUI
NML
CMD
STAT
ERR
EMCTASK
RS-274
(GCODE)
INTERPRETER
SEQUENCING
LOGIC
NML?
CMD
STAT
ERR
NML
CMD
STAT
ERR
NON-REALTIME
REALTIME
SHARED MEMORY BUFFER
FIFOS?
EMCMOT
CARTESIAN
POSITION
MOTION
COMMANDS
STATUS
TRAJECTORY
PLANNER
?
FORWARD
KINEMATICS
INVERSE
KINEMATICS
LIMIT & HOME
STATUS
AXIS 1
INTERPOLATOR
?
AXIS N
PID
SERVO
?
UNIT
CONVERT
UNIT
CONVERT
ENCODER
COUNTER
D/A
CONVERTER
LIMIT
SWITCHES
POWER
AMP
ENCODER
MOTOR
REALTIME HARDWARE DEVICES
EMCIO
COORDINATING
CONTROLLER
SPINDLE
CONTROLLER
AUXILIARY
CONTROLLER
COOLANT
CONTROLLER
NON-REALTIME HARDWARE DEVICES

3.4.1 LinuxCNC-Softwarearchitektur

Auf der gröbsten Ebene ist LinuxCNC eine Hierarchie von drei Controllern: der Task-Level-Befehlshandler und Programminterpreter, der Motion-Controller und der diskrete E/A-Controller (engl. I/O). Der diskrete E/A-Controller ist als eine Hierarchie von Controllern implementiert, in diesem Fall für Spindel-, Kühlmittel- und Hilfs-Subsysteme (z. B. Notaus). Der Task-Controller koordiniert die Aktionen der Bewegungssteuerung und der diskreten E/A-Steuerung. Deren Aktionen werden in konventionellen numerischen Steuerungs- ”G- und M-Code” Programmen programmiert, die von der Aufgabensteuerung in NML-Nachrichten interpretiert und zu den entsprechenden Zeitpunkten an die Bewegungssteuerung (motion) gesendet werden.

3.5 Motion Controller Einführung

Der Motion Controller ist eine Echtzeitkomponente. Er empfängt Bewegungssteuerungsbefehle von den Nicht-Echtzeit-Teilen von LinuxCNC (d.h. dem G-Code-Interpreter/Task, GUIs usw.) und führt diese Befehle in seinem Echtzeit-Kontext aus. Die Kommunikation vom Nicht-Echtzeit-Kontext zum Echtzeit-Kontext erfolgt über einen Message-Passing-IPC-Mechanismus, der Shared Memory verwendet, und über die Hardware-Abstraktionsschicht (HAL).

Der Status des Motion Controllers ist für den Rest von LinuxCNC durch die gleiche Message-Passing Shared Memory IPC (inter-process communication), und durch HAL zur Verfügung gestellt.

Der Motion Controller interagiert mit den Motorcontrollern und anderer Echtzeit- und Nicht-Echtzeit-Hardware über HAL.

In diesem Dokument wird davon ausgegangen, dass der Leser über ein grundlegendes Verständnis des HAL verfügt, und es werden Begriffe wie HAL-Pins, HAL-Signale usw. verwendet, ohne sie zu erläutern. Weitere Informationen über den HAL finden Sie im HAL-Handbuch. Ein weiteres Kapitel dieses Dokuments wird sich mit den Interna des HAL selbst befassen, aber in diesem Kapitel verwenden wir nur die HAL-API wie sie in src/hal/hal.h definiert ist.

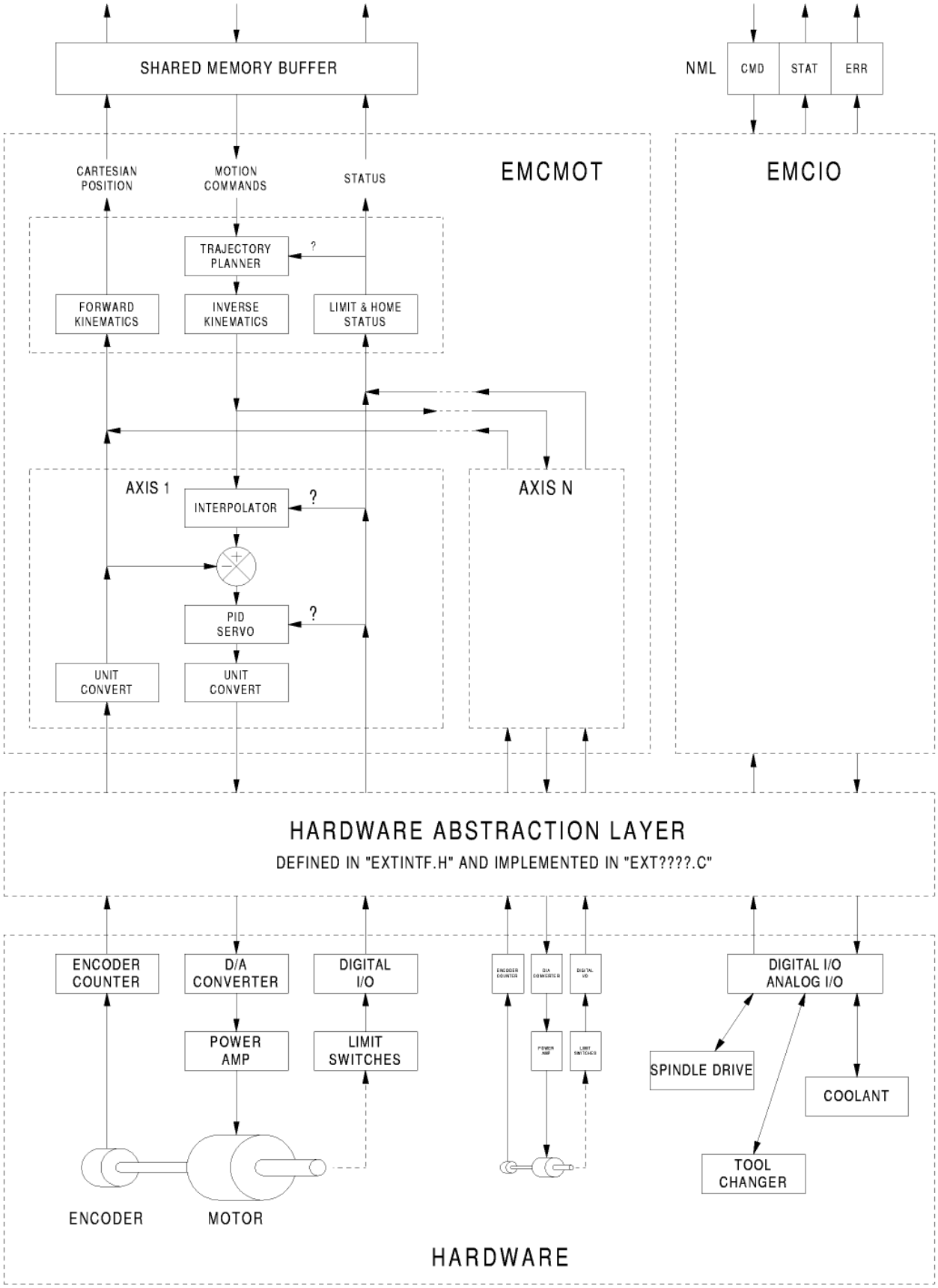
3.5.1 Bewegungssteuerungs- (engl. Motion-Controller-)Module

Die Echtzeitfunktionen des Bewegungssteuerungssystems werden mit Echtzeitmodulen implementiert - gemeinsam im userspace genutzte Objekte für Preempt-RT-Systeme oder Kernelmodule für einige Echtzeitimplementierungen im Kernelmodus wie RTAI:

- *tpmod* - Trajektorienplanung
- *homemod* - Referenzfahrtfunktionen
- *motmod* - verarbeitet NML-Befehle und steuert Hardware über HAL
- *Kinematikmodul* - führt Berechnungen der Vorwärtskinematik (bei bekannter Gelenk-Konstellation schließen auf Position der Spitze im Raum) und der Inversen Kinematik/Rückwärts-Transformation (um einem Bestimmten Punkt im Raum zu erreichen, wie müssen die Gelenke wie ausgelenkt sein) durch

LinuxCNC wird durch das **linuxcnc** Skript gestartet, welches eine Konfigurations-.ini-Datei liest und alle benötigten Prozesse startet. Für die Echtzeit-Bewegungssteuerung lädt das Skript zunächst die Standard-Module tpmod und homemod und lädt dann die Kinematik- und Bewegungsmodule entsprechend den Einstellungen in halfiles, die in der .ini-Datei angegeben sind.

Benutzerdefinierte Referenzfahrt- oder Flugbahnplanungsmodule können anstelle der Standardmodule über .ini-Datei-Einstellungen oder Befehlszeilenoptionen verwendet werden. Benutzerdefinierte Module müssen alle von den Standardmodulen verwendeten Funktionen implementieren. Das Dienstprogramm halcompile kann verwendet werden, um ein benutzerdefiniertes Modul zu erstellen.


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

3.6 Blockdiagramme und Datenfluss

Die folgende Abbildung ist das Blockdiagramm einer Gelenksteuerung. Für jedes Gelenk gibt es genau eine Gelenksteuerung. Die Gelenksteuerungen arbeiten auf einer niedrigeren Ebene als die Kinematik, einer Ebene, auf der alle Gelenke völlig unabhängig sind. Alle Daten für ein Gelenk befinden sich in einer einzigen Gelenkstruktur. Einige Elemente dieser Struktur sind im Blockdiagramm sichtbar, z. B. coarse_pos, pos_cmd und motor_pos_fb.

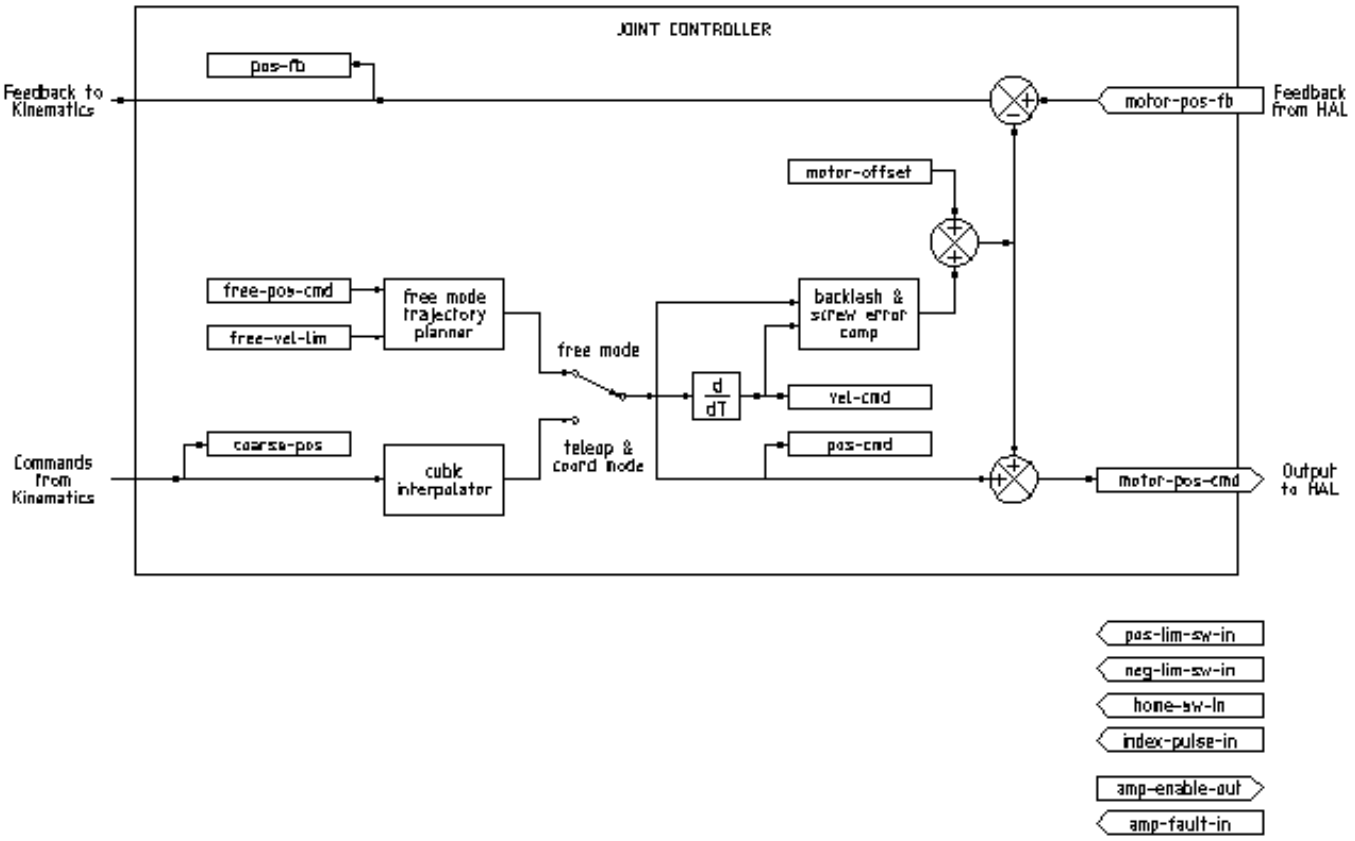


Abbildung 3.1: Blockdiagramm des Joint Controllers

Die obige Abbildung zeigt fünf der sieben Sätze von Positionsdaten, die den Hauptdatenfluss der Bewegungssteuerung darstellen. Die sieben Formen von Positionsdaten sind die folgenden:

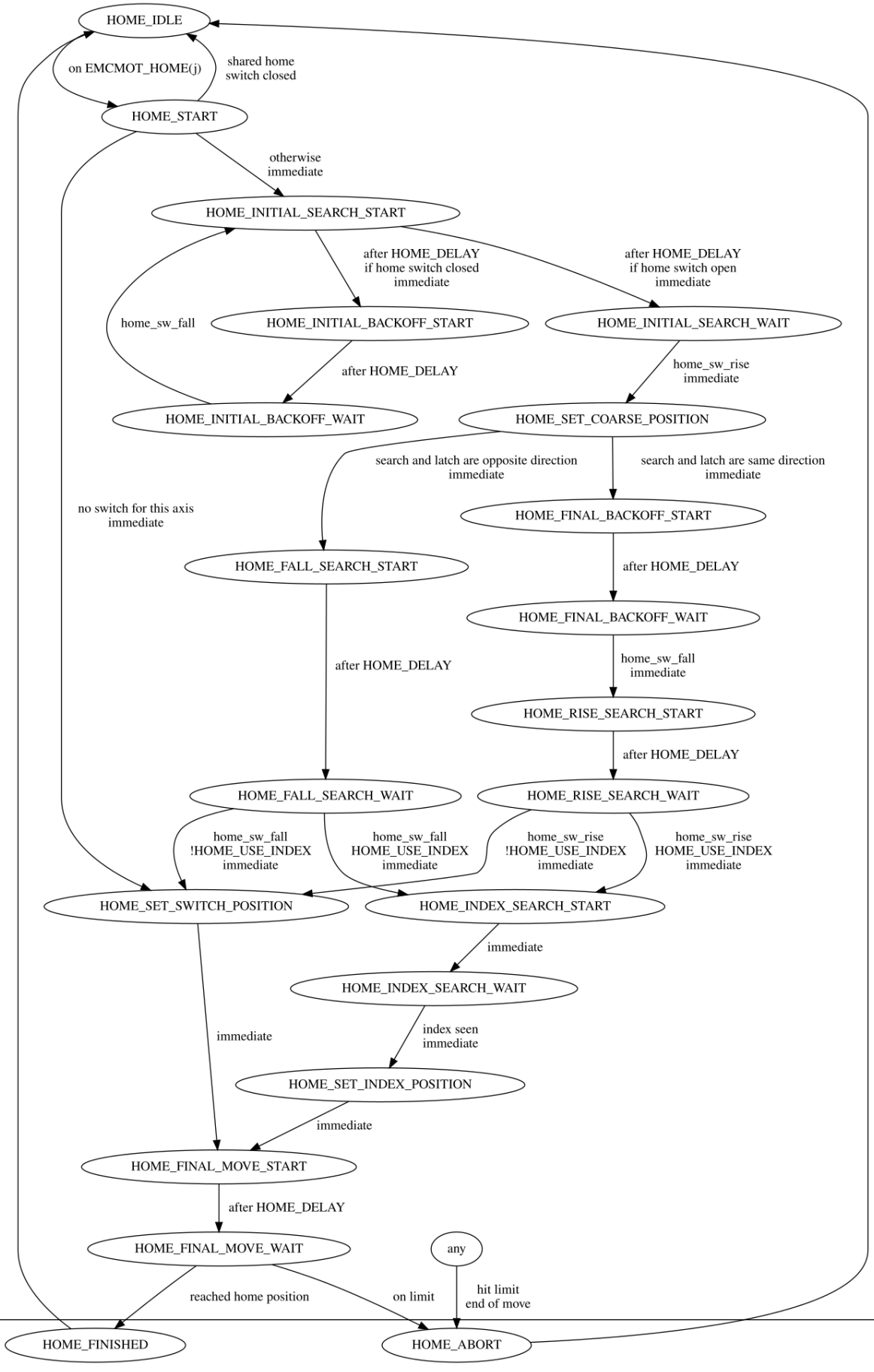
- *emcmotStatus->carte_pos_cmd* - Dies ist die gewünschte Position, in kartesischen Koordinaten. Sie wird mit der Traj-Rate aktualisiert, nicht mit der Servo-Rate. Im koordinierten Modus wird sie durch den Traj-Planer bestimmt. Im Teleop-Modus wird sie durch den Traj-Planer bestimmt? Im freien Modus sie entweder von actualPos kopiert oder durch Anwendung von Vorwärtskinematik auf (2) oder (3) erzeugt.
- *emcmotStatus->joints[n].coarse_pos* - Dies ist die gewünschte Position, in Gelenkkoordinaten, aber vor der Interpolation. Sie wird mit der traj rate aktualisiert, nicht mit der servo rate. Im Koordinatenmodus wird sie durch Anwendung von inversen Kinetiken auf (1) erzeugt. Im Teleop-Modus wird

sie durch Anwendung von inversen Kinematiken auf (1) erzeugt. Im freien Modus wird sie von (3) kopiert, glaube ich.

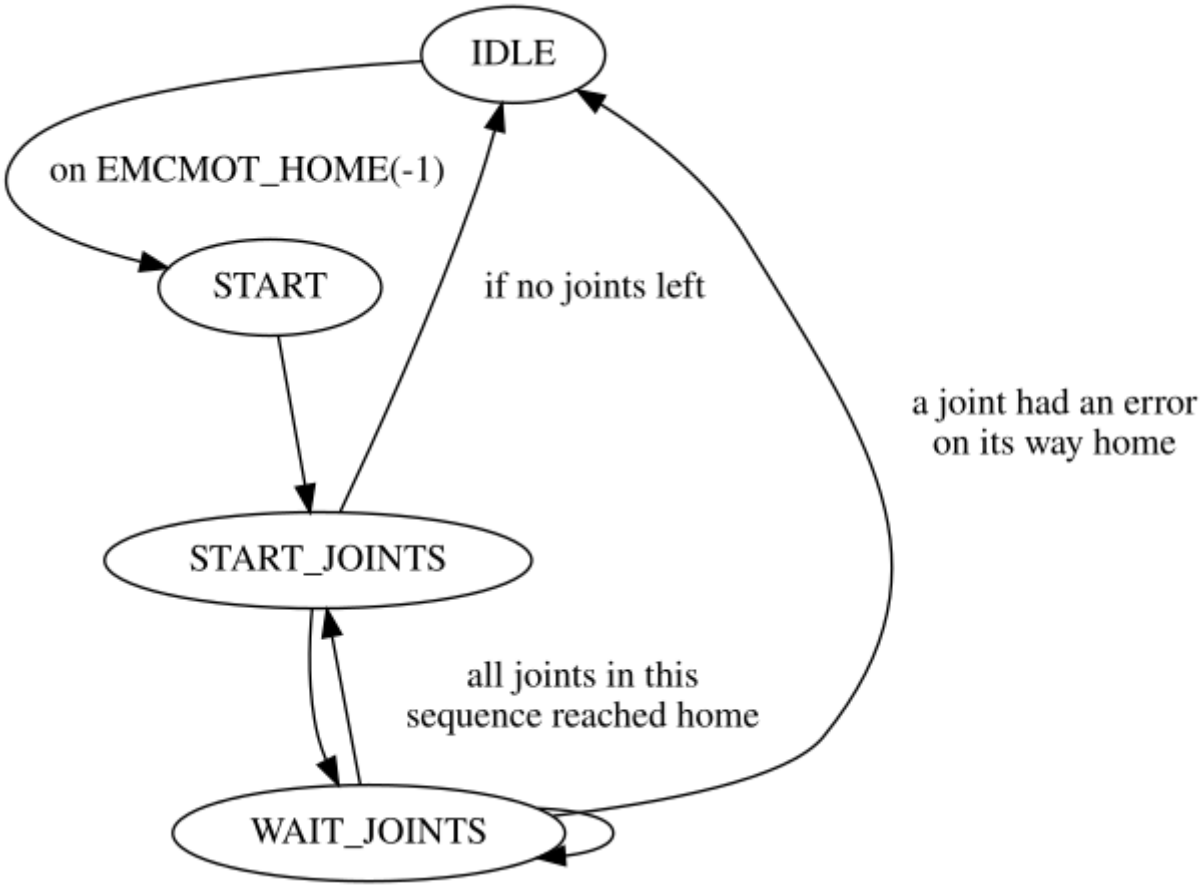
- ’emcmotStatus->joints[n].pos_cmd - Dies ist die gewünschte Position, in Gelenkkoordinaten, nach der Interpolation. Ein neuer Satz dieser Koordinaten wird in jeder Servoperiode erzeugt. Im Koordinatenmodus wird sie vom Interpolator aus (2) generiert. Im Teleop-Modus wird er durch den Interpolator aus (2) generiert. Im freien Modus wird sie durch den traj planner im freien Modus generiert.
- *emcmotStatus->joints[n].motor_pos_cmd* - Dies ist die gewünschte Position, in Motorkoordinaten. Die Motorkoordinaten werden generiert, indem die Kompensation des Umkehrspiels, die Kompensation der Spindelabweichung und der Offset (für die Referenzfahrt) zu (3) addiert werden. Diese wird unabhängig vom Modus auf die gleiche Weise erzeugt und ist die Ausgabe für die PID-Schleife oder eine andere Positionsschleife.
- *emcmotStatus->joints[n].motor_pos_fb* - Dies ist die aktuelle Position, in Motorkoordinaten. Es handelt sich um die Eingabe von Encodern oder anderen Rückkopplungsgeräten (oder von virtuellen Encodern bei Maschinen mit offenem Regelkreis). Die Geräte haben jeweils spezielle Möglichkeiten, solch feedback durch den Controller auslesen zu lassen.
- *emcmotStatus->joints[n].pos_fb* - Dies ist die tatsächliche Position, in Gelenkkoordinaten. Sie wird durch Subtraktion von Offset, Spindelkompensation und Spielkompensation von (5) erzeugt. Sie wird unabhängig von der Betriebsart auf dieselbe Weise erzeugt.
- *emcmotStatus->carte_pos_fb* - Dies ist die aktuelle Position in kartesischen Koordinaten. Sie wird mit der Traj-Rate aktualisiert, nicht mit der Servo-Rate. Idealerweise würde actualPos immer durch Anwendung der Vorwärtskinematik auf (6) berechnet werden. Es kann jedoch sein, dass die Vorwärtskinematik nicht verfügbar ist, oder dass sie unbrauchbar ist, weil eine oder mehrere Achsen nicht referenziert sind. In diesem Fall gibt es folgende Möglichkeiten: A) es durch Kopieren von (1) vorzutäuschen, oder B) zuzugeben, dass wir die kartesischen Koordinaten nicht wirklich kennen, und actualPos einfach nicht zu aktualisieren. Unabhängig davon, welcher Ansatz verwendet wird, sehe ich keinen Grund, es nicht auf die gleiche Weise zu tun, unabhängig von der Betriebsart. Ich würde das Folgende vorschlagen: Wenn es Vorwärts-Kins gibt, verwenden Sie sie, es sei denn, sie funktionieren nicht, weil die Achsen nicht beheimatet sind oder andere Probleme auftreten; in diesem Fall machen Sie (B). Wenn es keine Forward Kins gibt, dann mach (A), da sonst actualPos *nie* aktualisiert werden würde.

3.7 Referenzfahrt (engl. homing)

3.7.1 Zustandsdiagramm der Referenzfahrt



3.7.2 Ein weiteres Homing-Diagramm



3.8 Befehle

Die Befehle werden durch eine große Fallunterscheidung (Switch-Anweisung) in der Funktion emcmotCommandHandler() implementiert, die bei der Servo-Rate aufgerufen wird. Mehr zu dieser Funktion später.

Es gibt ungefähr 44 Befehle - diese Liste ist noch im Aufbau.

Anmerkung
Die Aufzählung cmd_code_t in motion.h enthält 73 Befehle, aber die switch-Anweisung in command.c berücksichtigt nur 70 Befehle (Stand: 6/5/2020). Die Befehle ENABLE_WATCHDOG / DISABLE_WATCHDOG befinden sich in motion-logger.c. Vielleicht sind sie veraltet. Der Befehl SET_TELEOP_VECTOR taucht nur in motion-logger.c auf und hat außer seinem eigenen Protokoll keine weiteren Auswirkungen.

3.8.1 Abbrechen

Der Befehl ABORT (engl. für Abbruch) stoppt einfach alle Bewegungen. Er kann jederzeit erteilt werden und wird immer akzeptiert. Er deaktiviert den Motion Controller nicht und ändert auch keine Zustandsinformationen, sondern bricht lediglich eine laufende Bewegung ab.[1]

[1] Es scheint, dass der Code auf höherer Ebene (TASK und höher) ABORT auch zum Löschen von Fehlern verwendet. Wann immer ein anhaltender Fehler auftritt (z. B. wenn die Hardware-Endschalter überschritten werden), sendet der übergeordnete

3.8.1.1 Anforderungen

Keine. Der Befehl wird immer angenommen und sofort ausgeführt.

3.8.1.2 Ergebnisse

Im freien Modus sind die Trajektorienplaner für den freien Modus deaktiviert. Das führt dazu, dass jedes Gelenk so schnell anhält, wie es seine Beschleunigungs- (Verzögerungs-) Grenze ermöglicht. Das Anhalten wird nicht koordiniert. Im Teleop-Modus wird die befohlene kartesische Geschwindigkeit auf Null gesetzt. Ich weiß nicht genau, welche Art von Stopp daraus resultiert (koordiniert, unkoordiniert, etc.), werde es aber irgendwann herausfinden. Im Koordinatenmodus wird der Koordinatenmodus-Trajektorienplaner angewiesen, die aktuelle Bewegung abzubrechen. Auch hier kenne ich das genaue Ergebnis nicht, werde es aber dokumentieren, sobald ich es herausgefunden habe.

3.8.2 FREE

Der Befehl FREE versetzt den Motion Controller in den freien Modus. Freier Modus bedeutet, dass jedes Gelenk unabhängig von allen anderen Gelenken ist. Kartesische Koordinaten, Posen und Kinematik werden im freien Modus ignoriert. Im Wesentlichen hat jedes Gelenk seinen eigenen einfachen Trajektorienplaner, und jedes Gelenk ignoriert die anderen Gelenke vollständig. Einige Befehle (wie Joint JOG und HOME) funktionieren nur im freien Modus. Andere Befehle, einschließlich aller Befehle, die mit kartesischen Koordinaten arbeiten, funktionieren im freien Modus überhaupt nicht.

3.8.2.1 Anforderungen

Der Befehlsinterpreter (engl. command handler) stellt keine Anforderungen an den FREE-Befehl, er wird immer akzeptiert. Wenn jedoch ein Gelenk in Bewegung ist (GET_MOTION_INPOS_FLAG() == FALSE), wird der Befehl ignoriert. Dieses Verhalten wird durch Code gesteuert, der sich jetzt in der Funktion *set_operating_mode()* in control.c befindet; dieser Code muss bereinigt werden. Meiner Meinung nach sollte der Befehl nicht stillschweigend ignoriert werden, sondern der Befehlsinterpreter sollte feststellen, ob er ausgeführt werden kann und einen Fehler zurückgeben, wenn dies nicht möglich ist.

3.8.2.2 Ergebnisse

Wenn sich die Maschine bereits im freien Modus befindet, geschieht nichts. Andernfalls wird die Maschine in den freien Modus versetzt. Der Trajektorienplaner jedes Gelenks im freien Modus wird mit der aktuellen Position des Gelenks initialisiert, aber die Planer sind nicht aktiviert und die Gelenke sind stationär.

3.8.3 TELEOP

Der Befehl TELEOP versetzt die Maschine in den Teleoperating-Modus. Im Teleop-Modus basiert die Bewegung der Maschine auf kartesischen Koordinaten unter Verwendung der Kinematik und nicht auf einzelnen Gelenken wie im freien Modus. Der Trajektorienplaner als solcher wird jedoch nicht verwendet, stattdessen wird die Bewegung durch einen Geschwindigkeitsvektor gesteuert. Die Bewegung im Teleop-Modus ähnelt dem Joggen, mit dem Unterschied, dass sie im kartesischen Raum und nicht im Gelenkraum erfolgt. Auf einer Maschine mit trivialer Kinematik gibt es kaum einen Unterschied zwischen dem Teleop-Modus und dem freien Modus, und die grafischen Benutzeroberflächen für diese

Code einen ständigen Strom von ABORTs an den Bewegungsregler, um den Fehler zu beheben. Tausende von ihnen.... Das bedeutet, dass die Bewegungssteuerung anhaltende Fehler vermeiden sollte. Dies muss untersucht werden.

Maschinen geben diesen Befehl möglicherweise nicht einmal aus. Bei nicht-trivialen Maschinen wie Robotern und Hexapoden wird der Teleop-Modus jedoch für die meisten vom Benutzer befohlenen Jog-Bewegungen verwendet.

3.8.3.1 Anforderungen

Der Command Handler weist den COORD-Befehl mit einer Fehlermeldung zurück, wenn die Kinematik nicht aktiviert werden kann, weil ein oder mehrere Gelenke nicht referenziert wurden. Auch wird der Befehl ignoriert (ohne Fehlermeldung), wenn sich ein Gelenk/Achse in Bewegung befindet (GET_MOTION_INPOS_FLAG() == FALSE). Dieses Verhalten wird durch Code gesteuert, der sich jetzt in der Funktion *set_operating_mode()* in control.c befindet. Meiner Meinung nach sollte der Befehl nicht stillschweigend ignoriert werden, sondern es sollte festgestellt werden, ob er ausgeführt werden kann und ein Fehler zurückgeben werden, wenn dies nicht möglich ist.

3.8.3.2 Ergebnisse

Wenn sich die Maschine bereits im Teleop-Modus befindet, geschieht nichts. Andernfalls wird die Maschine in den Teleop-Modus versetzt. Der Kinematikcode wird aktiviert, die Interpolatoren werden entleert und geleert, und die kartesischen Geschwindigkeitsbefehle werden auf Null gesetzt.

3.8.4 COORD

Mit dem Befehl COORD wird die Maschine in den Koordinaten-Modus versetzt. Im Koordinaten-Modus basiert die Bewegung der Maschine auf kartesischen Koordinaten unter Verwendung der Kinematik und nicht auf einzelnen Gelenken wie im freien Modus. Außerdem wird der Haupt-Trajektorienplaner verwendet, um die Bewegung auf der Grundlage von LINE-, CIRCLE- und/oder PROBE-Befehlen in der Warteschlange zu erzeugen. Der Koordinatenmodus ist der Modus, der bei der Ausführung eines G-Code-Programms verwendet wird.

3.8.4.1 Anforderungen

Der Command Handler weist den COORD-Befehl mit einer Fehlermeldung zurück, wenn die Kinematik nicht aktiviert werden kann, weil ein oder mehrere Gelenke nicht referenziert wurden. Auch wird der Befehl ignoriert (ohne Fehlermeldung), wenn sich ein Gelenk/Achse in Bewegung befindet (GET_MOTION_INPOS_FLAG() == FALSE). Dieses Verhalten wird durch Code gesteuert, der sich jetzt in der Funktion *set_operating_mode()* in control.c befindet. Meiner Meinung nach sollte der Befehl nicht stillschweigend ignoriert werden, sondern es sollte festgestellt werden, ob er ausgeführt werden kann und ein Fehler zurückgeben werden, wenn dies nicht möglich ist.

3.8.4.2 Ergebnisse

Wenn sich die Maschine bereits im Koordinatenmodus befindet, geschieht nichts. Andernfalls wird die Maschine in den Koordinatenmodus versetzt. Der Kinematikcode wird aktiviert, die Interpolatoren werden entleert und geleert, und die Warteschlangen des Bahnplaners sind leer. Der Trajektorienplaner ist aktiv und wartet auf einen LINE-, CIRCLE- oder PROBE-Befehl.

3.8.5 ENABLE (AKTIVIEREN)

Der Befehl ENABLE aktiviert den Motion Controller.

3.8.5.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.5.2 Ergebnisse

Wenn der Controller bereits aktiviert ist, passiert nichts. Ansonsten wird der Controller aktiviert. Warteschlangen und Interpolatoren werden geleert. Alle Bewegungs- oder Referenzfahrtvorgänge werden abgebrochen. Die mit aktiven Gelenken verbundenen Amp-Enable-Ausgänge werden eingeschaltet. Wenn keine Vorwärtskinematik verfügbar ist, wird die Maschine in den freien Modus geschaltet.

3.8.6 DISABLE (DEAKTIVIEREN)

Mit dem Befehl DISABLE wird der Motion Controller deaktiviert.

3.8.6.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.6.2 Ergebnisse

Wenn der Controller bereits deaktiviert ist, passiert nichts. Andernfalls wird der Controller deaktiviert. Warteschlangen und Interpolatoren werden geleert. Alle Bewegungs- oder Referenzfahrtvorgänge werden beendet. Die mit aktiven Gelenken verbundenen Amp-Enable-Ausgänge werden ausgeschaltet. Wenn die Vorwärtskinematik nicht verfügbar ist, wird die Maschine in den freien Modus geschaltet.

3.8.7 ENABLE_AMPLIFIER

Der Befehl ENABLE_AMPLIFIER schaltet den Verstärkerfreigabe-Ausgang (engl. amp enable output) für einen einzelnen Ausgangsverstärker ein, ohne etwas anderes zu ändern. Kann verwendet werden, um einen Spindeldrehzahlregler zu aktivieren.

3.8.7.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.7.2 Ergebnisse

Derzeit nichts. (Ein Aufruf der alten extAmpEnable-Funktion ist derzeit auskommentiert.) Eventuell wird der Amp-Enable-HAL-Pin auf true gesetzt.

3.8.8 DISABLE_AMPLIFIER

Der Befehl DISABLE_AMPLIFIER schaltet den Amp-Enable-Ausgang für einen einzelnen Verstärker aus, ohne etwas anderes zu ändern. Auch dies ist nützlich für Spindeldrehzahlregler.

3.8.8.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.8.2 Ergebnisse

Derzeit nichts. (Ein Aufruf der alten extAmpEnable Funktion ist derzeit auskommentiert.) Eventuell wird der Amp Enable HAL Pin auf false gesetzt.

3.8.9 ACTIVATE_JOINT

Der Befehl ACTIVATE_JOINT schaltet alle Berechnungen ein, die mit einem einzelnen Gelenk verbunden sind, ändert aber nicht den Amp-Enable-Ausgangsstift des Gelenks.

3.8.9.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.9.2 Ergebnisse

Die Berechnungen für das angegebene Gelenk werden aktiviert. Der Verstärker-Freigabe-Pin (amp enable pin) wird nicht verändert, jedoch werden alle nachfolgenden ENABLE- oder DISABLE-Befehle den Verstärker-Freigabe-Pin des Gelenks verändern.

3.8.10 DEACTIVATE_JOINT

Der Befehl DEACTIVATE_JOINT schaltet alle Berechnungen aus, die mit einem einzelnen Gelenk verbunden sind, ändert aber nicht dem amp enable output Pin des Gelenks.

3.8.10.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.10.2 Ergebnisse

Die Berechnungen für das angegebene Gelenk werden aktiviert. Der amp enable-Pin wird nicht verändert, und nachfolgende ENABLE- oder DISABLE-Befehle ändern den amp-enable-Pin des Gelenks nicht.

3.8.11 ENABLE_WATCHDOG

Der Befehl ENABLE_WATCHDOG aktiviert einen hardwarebasierten Watchdog (falls vorhanden).

3.8.11.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.11.2 Ergebnisse

Derzeit nichts. Der alte Watchdog war ein seltsames Ding, das eine bestimmte Soundkarte verwendete. Möglicherweise wird in Zukunft eine neue Watchdog-Schnittstelle entwickelt.

3.8.12 DISABLE_WATCHDOG

Der Befehl DISABLE_WATCHDOG deaktiviert einen hardwarebasierten Watchdog (falls vorhanden).

3.8.12.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.12.2 Ergebnisse

Derzeit nichts. Der alte Watchdog war ein seltsames Ding, das eine bestimmte Soundkarte verwendete. Möglicherweise wird in Zukunft eine neue Watchdog-Schnittstelle entwickelt.

3.8.13 PAUSE

Der Befehl PAUSE hält den Trajektorien-Planer an. Er hat keinen Effekt im freien oder Teleop-Modus. Ich weiß ich nicht, ob alle Bewegungen sofort angehalten werden, oder ob der aktuelle Bewegung abgeschlossen und dann angehalten wird, bevor ein anderer Bewegung aus der Warteschlange gezogen wird.

3.8.13.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.13.2 Ergebnisse

Der Trajektorienplaner pausiert.

3.8.14 RESUME

Der Befehl RESUME startet den Trajektorienplaner neu, wenn er angehalten wurde. Er hat keine Auswirkung im freien oder Teleop-Modus, oder wenn der Planer nicht angehalten ist.

3.8.14.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.14.2 Ergebnisse

Der Trajektorienplaner arbeitet weiter.

3.8.15 STEP

Der STEP-Befehl (engl. für Schritt) startet den Trajektorienplaner neu, wenn er angehalten wurde, und weist ihn an, wieder anzuhalten, wenn er einen bestimmten Punkt erreicht. Er hat keine Wirkung im freien oder Teleop-Modus. Zu diesem Zeitpunkt weiß ich nicht genau, wie das funktioniert. Ich werde hier mehr Dokumentation hinzufügen, wenn ich mich näher mit dem Trajektorienplaner beschäftige.

3.8.15.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.15.2 Ergebnisse

Der Trajektorienplaner wird fortgesetzt und hält wieder an, sobald er einen bestimmten Punkt erreicht.

3.8.16 SCALE

Der Befehl SCALE skaliert alle Geschwindigkeitsgrenzen und Befehle um einen bestimmten Betrag. Er wird verwendet, um die Vorschubgeschwindigkeit zu überschreiben und andere ähnliche Funktionen zu implementieren. Die Skalierung funktioniert in den Modi Free, Teleop und Coord und wirkt sich auf alles aus, einschließlich Referenzfahrtgeschwindigkeiten usw. Die Geschwindigkeitsgrenzen der einzelnen Gelenke sind jedoch nicht betroffen.

3.8.16.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert.

3.8.16.2 Ergebnisse

Alle Geschwindigkeitsbefehle werden um die angegebene Konstante skaliert.

3.8.17 OVERRIDE_LIMITS

Der Befehl OVERRIDE_LIMITS verhindert das Auslösen der Endschalter bis zum Ende des nächsten JOG-Befehls. Er wird normalerweise verwendet, um eine Maschine nach dem Auslösen von einem Endschalter zu verfahren. (Der Befehl kann auch verwendet werden, um Grenzwerte zu überschreiben oder eine vorherige Überschreibung aufzuheben.)

3.8.17.1 Anforderungen

Keine. Der Befehl kann jederzeit erteilt werden und wird immer akzeptiert. (Ich denke, es sollte nur im freien Modus funktionieren.)

3.8.17.2 Ergebnisse

Die Begrenzungen für alle Gelenke werden bis zum Ende des nächsten JOG-Befehls außer Kraft gesetzt. (Dies ist derzeit nicht möglich… sobald ein OVERRIDE_LIMITS-Befehl empfangen wird, werden die Begrenzungen ignoriert, bis ein weiterer OVERRIDE_LIMITS-Befehl sie wieder aktiviert.)

3.8.18 HOME

Der HOME-Befehl leitet eine Referenzfahrt an einem bestimmten Gelenk ein. Die tatsächliche Referenzierungssequenz wird durch eine Reihe von Konfigurationsparametern bestimmt und kann vom einfachen Setzen der aktuellen Position auf Null bis hin zu einer mehrstufigen Suche nach einem Referenzschalter und einem Indeximpuls, gefolgt von einer Bewegung zu einer beliebigen Referenzposition, reichen. Weitere Informationen über die Referenzfahrt-Sequenz finden Sie im Abschnitt Referenzfahrt des Integrator-Handbuchs.

3.8.18.1 Anforderungen

Der Befehl wird stillschweigend ignoriert, es sei denn, die Maschine befindet sich im freien Modus.

3.8.18.2 Ergebnisse

Jede Bewegung von Gelenken wird abgebrochen, und die Referenzfahrt beginnt.

3.8.19 JOG_CONT

Der Befehl JOG_CONT initiiert eine kontinuierliche Bewegung an einem einzelnen Gelenk. Eine kontinuierliche Bewegung wird erzeugt, indem die Zielposition des Free Mode Trajektorienplaners auf einen Punkt hinter dem Ende des Bewegungsbereichs des Gelenks gesetzt wird. Dadurch wird sichergestellt, dass der Planer sich ständig bewegt, bis er entweder durch die Gelenkgrenzen oder einen ABORT-Befehl gestoppt wird. Normalerweise sendet eine GUI einen JOG_CONT-Befehl, wenn der Benutzer eine Jog-Taste drückt, und ABORT, wenn die Taste losgelassen wird.

3.8.19.1 Anforderungen

Der Befehlshandler weist den JOG_CONT-Befehl mit einer Fehlermeldung zurück, wenn sich die Maschine nicht im freien Modus befindet, wenn ein Gelenk in Bewegung ist (GET_MOTION_INPOS_FLAG() == FALSE) oder wenn die Bewegung nicht aktiviert ist. Außerdem wird der Befehl ignoriert, wenn sich das Gelenk bereits an oder über seinem Limit befindet und der befohlene Jog die Situation verschlimmern würde.

3.8.19.2 Ergebnisse

Der Free-Mode-Trajektorienplaner für das durch emcmotCommand->axis identifizierte Gelenk wird aktiviert, mit einer Zielposition jenseits des Endes der Gelenkbewegung und einem Geschwindigkeitslimit von emcmotCommand->vel. Damit wird die Bewegung des Gelenks eingeleitet, und die Bewegung wird fortgesetzt, bis sie durch einen ABORT-Befehl oder durch das Erreichen eines Limits gestoppt wird. Der Planer im freien Modus beschleunigt zu Beginn der Bewegung mit der Gelenkbeschleunigungsgrenze und bremst mit der Gelenkbeschleunigungsgrenze ab, wenn er anhält.

3.8.20 JOG_INCR

Der JOG_INCR-Befehl initiiert einen inkrementellen Tippbetrieb für ein einzelnes Gelenk. Inkrementelle Verfahrbewegungen sind kumulativ, d.h. wenn Sie zwei JOG_INCR-Befehle geben, die jeweils eine Bewegung von 0,100 Zoll erfordern, ergibt dies eine Bewegung von 0,200 Zoll, auch wenn der zweite Befehl gegeben wird, bevor der erste beendet ist. Normalerweise stoppen Inkremental-Jogs, wenn sie die gewünschte Strecke zurückgelegt haben, aber sie stoppen auch, wenn sie an eine Grenze stoßen, oder bei einem ABORT-Befehl.

3.8.20.1 Anforderungen

Der Befehlshandler lehnt den JOG_INCR-Befehl stillschweigend ab, wenn sich die Maschine nicht im freien Modus befindet, wenn ein Gelenk in Bewegung ist (GET_MOTION_INPOS_FLAG() == FALSE) oder wenn die Bewegung nicht aktiviert ist. Der Befehl wird auch ignoriert, wenn das Gelenk bereits an oder über seinem Limit ist und der Jog-Befehl die Situation verschlimmern würde.

3.8.20.2 Ergebnisse

Der Free-Mode-Trajektorienplaner für das durch emcmotCommand->axis identifizierte Gelenk wird aktiviert, die Zielposition wird um emcmotCommand->offset inkrementiert/dekrementiert, und die Geschwindigkeitsgrenze wird auf emcmotCommand->vel gesetzt. Der Trajektorienplaner im freien Modus wird eine glatte trapezförmige Bewegung von der aktuellen Position zur Zielposition erzeugen. Der Planer kann Änderungen der Zielposition, die während der Bewegung auftreten, korrekt verarbeiten, so dass mehrere JOG_INCR-Befehle in schneller Folge ausgegeben werden können. Der Planer im freien Modus beschleunigt zu Beginn der Bewegung mit der Gelenkbeschleunigungsgrenze und bremst mit der Gelenkbeschleunigungsgrenze ab, um an der Zielposition anzuhalten.

3.8.21 JOG_ABS

Der Befehl JOG_ABS initiiert eine absolute Bewegung an einem einzelnen Gelenk. Eine absolute Bewegung ist eine einfache Bewegung zu einer bestimmten Position in Gelenkkoordinaten. Normalerweise stoppen absolute Bewegungsabläufe, wenn sie die gewünschte Position erreichen, aber sie stoppen auch, wenn sie eine Grenze erreichen, oder bei einem ABORT-Befehl.

3.8.21.1 Anforderungen

Der Befehlshandler (engl. command handler) lehnt den JOG_ABS-Befehl stillschweigend ab, wenn sich die Maschine nicht im freien Modus befindet, wenn ein Gelenk in Bewegung ist (GET_MOTION_INPOS_FLAG == FALSE) oder wenn Bewegungen (engl. motion) nicht aktiviert sind. Der Befehl wird auch dann stillschweigend ignoriert, wenn das Gelenk bereits an oder über seinem Limit ist und der Jog-Befehl die Situation verschlimmern würde.

3.8.21.2 Ergebnisse

Der Free-Mode-Trajektorienplaner für das durch emcmotCommand->axis identifizierte Gelenk wird aktiviert, die Zielposition wird auf emcmotCommand->offset gesetzt, und die Geschwindigkeitsgrenze wird auf emcmotCommand->vel gesetzt. Der Trajektorienplaner im freien Modus erzeugt eine gleichmäßige trapezförmige Bewegung von der aktuellen Position zur Zielposition. Der Planer kann Änderungen an der Zielposition, die während der Bewegung auftreten, korrekt verarbeiten. Wenn mehrere JOG_ABS-Befehle kurz hintereinander ausgegeben werden, ändert jeder neue Befehl die Zielposition, und die Maschine fährt zur befohlenen Endposition. Der Planer im freien Modus beschleunigt zu Beginn der Bewegung mit der Gelenkbeschleunigungsgrenze und bremst mit der Gelenkbeschleunigungsgrenze ab, um an der Zielposition anzuhalten.

3.8.22 SET_LINE

Der Befehl SET_LINE fügt eine gerade Linie in die Warteschlange des Trajektorienplaners ein.

(Mehr dazu später)

3.8.23 SET_CIRCLE

Der Befehl SET_CIRCLE fügt eine kreisförmige Bewegung in die Warteschlange des Trajektorienplaners ein.

(Mehr dazu später)

3.8.24 SET_TELEOP_VECTOR

Der Befehl SET_TELEOP_VECTOR weist den Motion Controller an, sich entlang eines bestimmten Vektors im kartesischen Raum zu bewegen.

(Mehr dazu später)

3.8.25 PROBE

Der PROBE-Befehl weist den Motion Controller an, sich zu einem bestimmten Punkt im kartesischen Raum zu bewegen und seine Position zu stoppen und aufzuzeichnen, wenn der Probe-Eingang ausgelöst wird.

(Mehr dazu später)

3.8.26 CLEAR_PROBE_FLAG

Der Befehl CLEAR_PROBE_FLAG wird verwendet, um den Sondeneingang in Vorbereitung auf einen PROBE-Befehl zurückzusetzen. (Frage: warum sollte der PROBE-Befehl den Eingang nicht automatisch zurücksetzen?)

(Mehr dazu später)

3.8.27 SET_xix

Es gibt etwa 15 SET_xxx-Befehle, wobei xxx der Name eines Konfigurationsparameters ist. Es ist davon auszugehen, dass es noch einige weitere SET-Befehle geben wird, wenn weitere Parameter hinzugefügt werden. Ich würde gerne eine sauberere Methode zum Setzen und Lesen von Konfigurationsparametern finden. Bei den vorhandenen Methoden müssen jedes Mal, wenn ein Parameter hinzugefügt wird, viele Codezeilen zu mehreren Dateien hinzugefügt werden. Ein großer Teil dieses Codes ist für jeden Parameter identisch oder fast identisch.

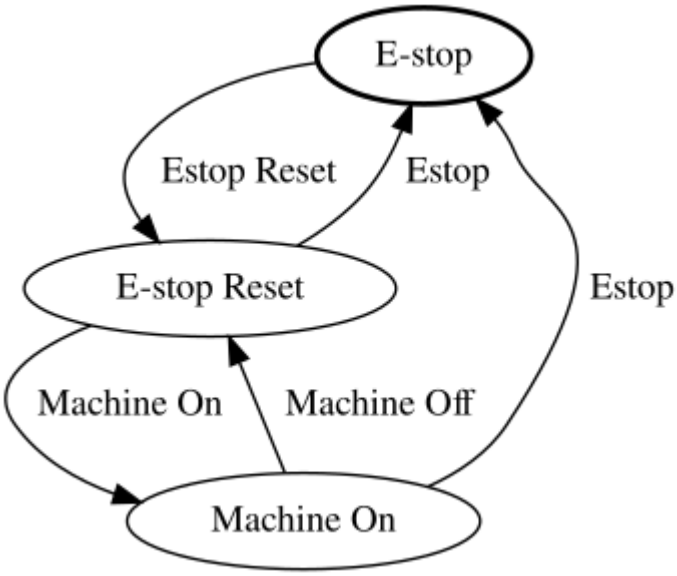
3.9 Umkehrspiel- und Schrauben-/Schneckenfehlerkompensation

```
+ Umkehrspiel- und Schraubenfehlerkompensation
```

3.10 Task-Controller (EMCTASK)

3.10.1 Zustand

Die Aufgabe hat drei mögliche interne Zustände: **Notaus** (E-Stop), **Notaus-Reset** (E-Stop Reset) und **Maschine Ein** (Machine On).



3.11 IO-Controller (EMCIO)

Der I/O controller ist Teil von TASK. Er interagiert mit externen I/O über HAL pins.

Derzeit werden NOTAUS (ESTOP)/Enable, Kühlmittel und Werkzeugwechsel von iocontrol abgewickelt. Dies sind Ereignisse mit relativ geringer Geschwindigkeit, koordinierte Hochgeschwindigkeits-E/A werden in der Bewegung (motion) abgewickelt.

emctaskmain.cc sendet I/O Befehle via taskclass.cc.

Prozess der iocontrol-Hauptschleife:

- prüft, ob sich die HAL-Eingänge geändert haben
- prüft, ob read_tool_inputs() anzeigt, dass der Werkzeugwechsel abgeschlossen ist und setzt emcioStatus.status

3.12 Benutzerschnittstellen

`FIXME Benutzerschnittstellen`

3.13 libnml Einführung

libnml ist von der NIST rcslib abgeleitet, ohne die ganze Multiplattform-Unterstützung. Viele der Wrapper um plattformspezifischen Code wurde entfernt, zusammen mit einem Großteil des Codes, der nicht für LinuxCNC erforderlich ist. Es ist zu hoffen, dass ausreichende Kompatibilität mit rcslib bleibt, so dass Anwendungen auf Nicht-Linux-Plattformen implementiert werden können und dies weiterhin in der Lage sind, mit LinuxCNC zu kommunizieren.

Dieses Kapitel ist nicht als endgültiger Leitfaden für die Verwendung von libnml (oder rcslib) gedacht. Stattdessen soll es einen Überblick über jede C++-Klasse und ihre Mitgliedsfunktionen geben. Anfänglich werden die meisten dieser Anmerkungen zufällige Kommentare sein, die bei der Überprüfung und Änderung des Codes hinzugefügt werden.

3.14 LinkedList

Basisklasse zur Verwaltung einer verknüpften Liste. Dies ist einer der wichtigsten Bausteine für die Weitergabe von NML-Nachrichten und verschiedenen internen Datenstrukturen.

3.15 LinkedListNode

Basisklasse für die Erstellung einer verknüpften Liste - Zweck, Zeiger auf den vorherigen und den nächsten Knoten, Zeiger auf die Daten und die Größe der Daten zu halten.

Es ist kein Speicher für die Datenspeicherung zugewiesen.

3.16 SharedMemory

Stellt einen Block von gemeinsamem Speicher zusammen mit einer Semaphore (geerbt von der Klasse Semaphore) bereit. Die Erstellung und Zerstörung der Semaphore wird durch den SharedMemory-Konstruktor und -Destruktor gehandhabt.

3.17 ShmBuffer

Klasse zur Weitergabe von NML-Nachrichten zwischen lokalen Prozessen unter Verwendung eines gemeinsamen Speicherpuffers. Ein Großteil der internen Arbeitsweise wird von der CMS-Klasse geerbt.

3.18 Timer

Die Klasse Timer bietet einen periodischen Timer, der nur durch die Auflösung der Systemuhr begrenzt ist. Wenn zum Beispiel ein Prozess alle 5 Sekunden ausgeführt werden muss, unabhängig von der Zeit, die für die Ausführung des Prozesses benötigt wird. Der folgende Codeschnipsel zeigt wie :

```
main()
{
    timer = new Timer(5.0);    /* Initialize a timer mit einer 5 Sekunden-Schleife (engl. ↩
        loop) */
    while(0) {
        /* hier wird irgendein beliebiger Prozess ausgeführt */
        timer.wait();    /* Warte bis zum nächsten 5 Sekunden Interval */
    }
    delete timer;
}
```

3.19 Semaphore

Die Klasse Semaphore bietet eine Methode des gegenseitigen Ausschlusses für den Zugriff auf eine gemeinsam genutzte Ressource. Die Funktion zum Abrufen einer Semaphore kann entweder blockieren bis der Zugriff möglich ist, nach einer Zeitüberschreitung zurückkehren oder sofort mit oder ohne

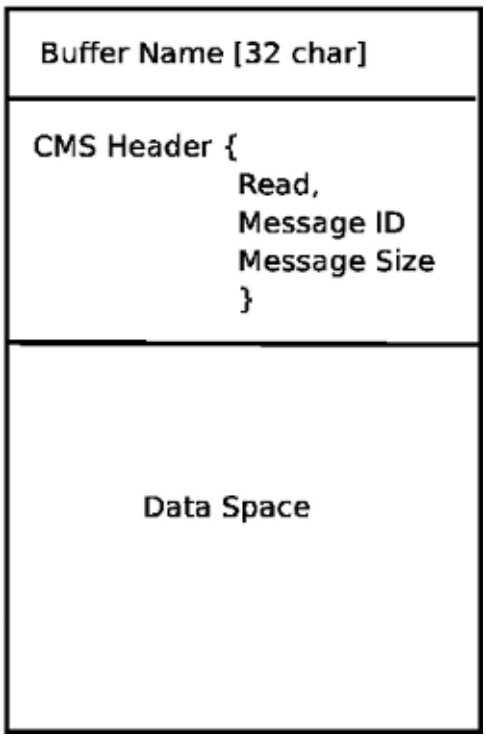
Abrufen der Semaphore zurückkehren. Der Konstruktor erstellt eine Semaphore oder hängt an eine bestehende an, wenn die ID bereits in Gebrauch ist.

Die Funktion Semaphore::destroy() darf nur vom letzten Prozess aufgerufen werden.

3.20 CMS

Das Herzstück von libnml ist die CMS-Klasse. Sie enthält die meisten der von libnml und letztlich von NML verwendeten Funktionen enthält. Viele der internen Funktionen sind überladen, um spezifische hardwareabhängige Methoden der Datenübergabe zu ermöglichen. Letztlich dreht sich alles um einen zentralen Speicherblock (der als *Nachrichtenpuffer* oder einfach *Puffer* bezeichnet wird). Dieser Puffer kann ein gemeinsam genutzter Speicherblock sein, auf den andere CMS/NML-Prozesse zugreifen, oder ein lokaler und privater Puffer für Daten, die über Netzwerk- oder serielle Schnittstellen übertragen werden.

Der Puffer wird zur Laufzeit dynamisch zugewiesen, um eine größere Flexibilität des CMS/NML-Subsystems zu ermöglichen. Die Puffergröße muss groß genug sein, um die größte Nachricht aufzunehmen, eine kleine Menge für den internen Gebrauch und die Möglichkeit, die Nachricht zu kodieren, wenn diese Option gewählt wird (auf kodierte Daten wird später eingegangen). Die folgende Abbildung zeigt eine interne Ansicht des Pufferspeichers.



CMS-Puffer Die CMS-Basisklasse ist in erster Linie für die Erstellung der Kommunikationswege und der Schnittstellen zum Betriebssystem verantwortlich.

3.21 Format der Konfigurationsdatei

Die NML-Konfiguration besteht aus zwei Arten von Zeilenformaten. Eines für Puffer und ein zweites für Prozesse, die mit den Puffern verbunden sind.

3.21.1 Pufferzeile

Das ursprüngliche NIST-Format der Pufferzeile ist:

- *B name type host size neut RPC# buffer# max_procs key [typspezifische configs]*
- *B* - kennzeichnet diese Zeile als eine Pufferkonfiguration.
- *name* - ist die Kennung des Puffers.
- *type* - beschreibt den Puffertyp - SHMEM, LOCMEM, FILEMEM, PHANTOM oder GLOBMEM.
- *host* - ist entweder eine IP-Adresse oder ein Hostname für den NML-Server
- *size* - ist die Größe des Puffers
- *neut* - ein boolescher Wert, der angibt, ob die Daten im Puffer in einem maschinenunabhängigen Format oder im Rohformat kodiert sind.
- *RPC#* - Obsolet - Platzhalter nur noch aus Gründen der Abwärtskompatibilität.
- *buffer#* - Eine eindeutige ID-Nummer, die verwendet wird, wenn ein Server mehrere Puffer kontrolliert.
- *max_procs* - ist die maximale Anzahl von Prozessen, die sich mit diesem Puffer verbinden dürfen.
- *key* - ist ein numerischer Bezeichner für einen gemeinsamen Speicherpuffer

3.21.2 Typspezifische Konfigurationen

Der Puffertyp impliziert zusätzliche Konfigurationsoptionen, während das Host-Betriebssystem bestimmte Kombinationen ausschließt. In dem Bemühen, die veröffentlichte Dokumentation in ein kohärentes Format zu bringen, wird nur der Puffertyp **SHMEM** behandelt.

- *mutex=os_sem* - Standardmodus für die Bereitstellung von Semaphore-Sperren des Pufferspeichers.
- *mutex=none* - Nicht verwendet
- *mutex=no_interrupts* - nicht anwendbar auf einem Linux-System
- *mutex=no_switching* - nicht anwendbar auf einem Linux-System
- *mutex=mao split* - Teilt den Puffer in zwei Hälften (oder mehr) und erlaubt einem Prozess, auf einen Teil des Puffers zuzugreifen, während ein zweiter Prozess in einen anderen Teil schreibt.
- *TCP=(Portnummer)* - Gibt an, welcher Netzwerkport verwendet werden soll.
- *UDP=(Portnummer)* - dito
- *STCP=(Portnummer)* - dito
- *serialPortDevName=(serielle Schnittstelle)* - Undokumentiert.
- *passwd=file_name.pwd* - Fügt dem Puffer eine Sicherheitsebene hinzu, indem jeder Prozess ein Passwort angeben muss.
- *bsem* - Die NIST-Dokumentation impliziert einen Schlüssel für einen blockierenden Semaphor, und wenn bsem=-1, werden blockierende Lesevorgänge verhindert.
- *queue* - Aktiviert die Weitergabe von Nachrichten in Warteschlangen.
- *ascii* - Nachrichten in einem einfachen Textformat kodieren
- *disp* - Nachrichten in einem für die Anzeige geeigneten Format kodieren (???)
- *xdr* - Kodierung von Nachrichten in External Data Representation. (siehe rpc/xdr.h für Details).
- *diag* - Aktiviert die im Puffer gespeicherten Diagnosen (Timings und Bytezahlen ?)

3.21.3 Prozesslinie

Das ursprüngliche NIST-Format der Prozesslinie ist:

P name buffer type host ops server timeout master c_num [type specific configs]

- *P* - kennzeichnet diese Zeile als eine Prozesskonfiguration.
- *name* - ist der Bezeichner des Prozesses.
- *buffer* - ist einer der an anderer Stelle in der Konfigurationsdatei definierten Puffer.
- *type* - legt fest, ob dieser Prozess relativ zum Puffer lokal oder entfernt ist.
- *host* - gibt an, wo im Netzwerk dieser Prozess ausgeführt wird.
- *ops* - gibt dem Prozess nur Lese-, nur Schreib- oder Lese-/Schreibzugriff auf den Puffer.
- *server* - gibt an, ob dieser Prozess einen Server für diesen Puffer betreiben wird.
- *timeout* - legt die Timeout-Eigenschaften für Zugriffe auf den Puffer fest.
- *master* - gibt an, ob dieser Prozess für die Erstellung und Löschung des Puffers verantwortlich ist.
- *c_num* - eine Ganzzahl zwischen Null und (max_procs -1)

3.21.4 Kommentare zur Konfiguration

Einige der Konfigurationskombinationen sind ungültig, während andere bestimmte Beschränkungen mit sich bringen. Auf einem Linux-System ist GLOBMEM überflüssig, während PHANTOM nur in der Testphase einer Anwendung wirklich nützlich ist, gleiches gilt für FILEMEM. LOCMEM ist für eine Multiprozess-Anwendung kaum von Nutzen und bietet nur begrenzte Leistungsvorteile gegenüber SHMEM. Damit bleibt SHMEM der einzige Puffertyp, der mit LinuxCNC verwendet werden kann.

Die Option neut ist nur in einem Multiprozessorsystem von Nutzen, in dem sich verschiedene (und inkompatible) Architekturen einen Speicherblock teilen. Die Wahrscheinlichkeit, ein solches System außerhalb eines Museums oder einer Forschungseinrichtung zu sehen, ist gering und nur für GLOBMEM-Puffer relevant.

Die RPC-Nummer ist als veraltet dokumentiert und wird nur aus Kompatibilitätsgründen beibehalten.

Mit einem eindeutigen Puffernamen scheint eine numerische Identität sinnlos zu sein. Ich muss den Code überprüfen, um die Logik zu erkennen. Auch das Schlüsselfeld scheint zunächst redundant zu sein und könnte vom Puffernamen abgeleitet werden.

Der Zweck der Begrenzung der Anzahl von Prozessen, die sich mit einem Puffer verbinden können, ist aus der vorhandenen Dokumentation und dem ursprünglichen Quellcode nicht ersichtlich. Es ist nicht schwieriger zu implementieren, wenn mehrere nicht spezifizierte Prozesse eine Verbindung zu einem Puffer herstellen können.

Die Mutex-Typen beschränken sich auf einen von zwei, den Standard ”os_sem” oder ”mao split”. Die meisten NML-Nachrichten sind relativ kurz und können mit minimalen Verzögerungen in oder aus dem Puffer kopiert werden, so dass Split-Reads nicht unbedingt erforderlich sind.

Die Datenkodierung ist nur relevant, wenn sie an einen entfernten Prozess übertragen wird - die Verwendung von TCP oder UDP impliziert XDR-Kodierung. Während die ASCII-Kodierung bei Diagnosen oder bei der Übermittlung von Daten an ein eingebettetes System, das NML nicht implementiert, von Nutzen sein kann.

Bei UDP-Protokollen werden weniger Daten überprüft und ein bestimmter Prozentsatz der Pakete kann verworfen werden. TCP ist zuverlässiger, aber geringfügig langsamer.

Wenn LinuxCNC an ein Netzwerk angeschlossen werden soll, würde man hoffen, dass es lokal und hinter einer Firewall ist. Über den einzigen Grund, den Zugriff auf LinuxCNC über das Internet zu

ermöglichen, wäre für die Ferndiagnose - Dies kann viel sicherer mit anderen Mitteln erreicht werden, vielleicht durch eine Web-Schnittstelle.

Das genaue Verhalten, wenn die Zeitüberschreitung auf Null oder einen negativen Wert gesetzt wird, geht aus den NIST-Dokumenten nicht hervor. Es werden nur INF und positive Werte erwähnt. Im Quellcode von rcslib ist jedoch ersichtlich, dass Folgendes gilt:

timeout > 0 Sperrung des Zugriffs, bis das Timeout-Intervall erreicht ist oder der Zugriff auf den Puffer möglich ist.

timeout = 0 Der Zugriff auf den Puffer ist nur möglich, wenn kein anderer Prozess gerade liest oder schreibt.

timeout < 0 oder INF Der Zugriff wird blockiert, bis der Puffer verfügbar ist.

3.22 NML-Basisklasse

Mehr zu Listen und die Beziehung zwischen NML, NMLmsg und den untergeordneten cms-Klassen.

Nicht zu verwechseln mit NMLmsg, RCS_STAT_MSG, oder RCS_CMD_MSG.

NML ist verantwortlich für das Parsen der Konfigurationsdatei, die Konfiguration der cms-Puffer und die Weiterleitung von Nachrichten an die richtigen Puffer. Zu diesem Zweck erstellt NML mehrere Listen für:

- cms-Puffer, die erstellt oder die verbunden wurden.
- Prozesse und die Puffer, mit denen sie verbunden sind
- eine lange Liste von Formatfunktionen für jeden Nachrichtentyp

Dieser letzte Punkt ist wahrscheinlich der Kern eines Großteils der schlechten Bewertung von libnml/rcslib und NML im Allgemeinen. Jede Nachricht, die über NML weitergegeben wird, erfordert neben den eigentlichen Daten auch eine gewisse Menge an Informationen, die angehängt werden müssen. Zu diesem Zweck werden nacheinander mehrere Formatierungsfunktionen aufgerufen, um Fragmente der Gesamtnachricht zusammenzusetzen. Die Formatierungsfunktionen umfassen NML_TYPE, MSG_TYPE sowie die in abgeleiteten NMLmsg-Klassen deklarierten Daten. Änderungen an der Reihenfolge, in der die Formatierungsfunktionen aufgerufen werden, und auch an den übergebenen Variablen führen zu einem Bruch der Kompatibilität mit rcslib, wenn daran herumgepfuscht wird. Es gibt Gründe für ein Beibehalten der Kompatiblität mit rcslib, aber auch für das Herumdoktoren an dem Code für eigene Anpassungen Die Frage ist, welche dieser Gründe überwiegen?

3.22.1 NML-Interna

3.22.1.1 NML-Konstruktor

NML::NML() parst die Konfigurationsdatei und speichert sie in einer verknüpften Liste, die in einzelnen Zeilen an cms-Konstruktoren übergeben wird. Es ist die Aufgabe des NML-Konstruktors, den entsprechenden cms-Konstruktor für jeden Puffer aufzurufen und eine Liste der cms-Objekte und der mit jedem Puffer verbundenen Prozesse zu führen.

Die in den Listen gespeicherten Zeiger sind die Grundlage auf der NML mit cms interagiert und warum Doxygen die tatsächlichen Beziehungen nicht aufzeigt.

Anmerkung
Die Konfiguration wird im Speicher abgelegt, bevor ein Zeiger auf eine bestimmte Zeile an den cms-Konstruktor übergeben wird. Der cms-Konstruktor analysiert dann die Zeile erneut, um einige Variablen zu extrahieren... Es wäre sinnvoller, das gesamte Parsing vorzunehmen und die Variablen in einer Struktur zu speichern, die an den CMS-Konstruktor weitergegeben wird - dies würde die Handhabung von Zeichenketten eliminieren und duplizierten Code im CMS reduzieren.

3.22.1.2 NML lesen/schreiben

Die Aufrufe von NML::read und NML::write führen beide ähnliche Aufgaben aus, nämlich die Verarbeitung der Nachricht - der einzige wirkliche Unterschied liegt in der Richtung des Datenflusses.

Ein Aufruf der Lesefunktion holt zunächst Daten aus dem Puffer und ruft dann format_output() auf, während eine Schreibfunktion format_input() aufrufen würde, bevor sie die Daten an den Puffer übergibt. In format_xxx() findet die Arbeit des Aufbaus oder Abbaus der Nachricht statt. Eine Liste verschiedener Funktionen wird nacheinander aufgerufen, um verschiedene Teile des NML-Headers (nicht zu verwechseln mit dem cms-Header) in die richtige Reihenfolge zu bringen - die letzte aufgerufene Funktion ist emcFormat() in emc.cc.

3.22.1.3 NMLmsg und NML-Beziehungen

NMLmsg ist die Basisklasse, von der alle Nachrichtenklassen abgeleitet sind. Für jede Nachrichtenklasse muss eine eindeutige ID definiert (und an den Konstruktor übergeben) werden sowie eine update(*cms)-Funktion. Die update()-Funktion wird von den NML-Lese-/Schreibfunktionen aufgerufen, wenn der NML-Formatierer aufgerufen wird - der Zeiger auf den Formatierer wird irgendwann im NML-Konstruktor deklariert worden sein. Durch die von NML erstellten verknüpften Listen ist es möglich, den cms-Zeiger auszuwählen, der an den Formatierer übergeben wird, und damit den zu verwendenden Puffer.

3.23 Hinzufügen von benutzerdefinierten NML-Befehlen

LinuxCNC ist ziemlich genial, aber einige Teile brauchen ein wenig manuelle Anpassung. Wie Sie wissen, erfolgt die Kommunikation über NML-Kanäle, die Daten werden durch einen Kanal gesendet, der in einer der Klassen in emc.hh definiert ist (und in emc.cc implementiert). Wenn jemand einen Nachrichtentyp benötigt, der nicht existiert, sollte er diese Schritte befolgen, um einen neuen hinzuzufügen. (Die Nachricht, die ich im Beispiel hinzugefügt habe, heißt EMC_IO_GENERIC (erbt von EMC_IO_CMD_MSG (erbt von RCS_CMD_MSG))

1. Hinzufügen der Definition der Klasse EMC_IO_GENERIC zu emc2/src/emc/nml_intf/emc.hh
2. Hinzufügen der Typdefinition: #define EMC_IO_GENERIC_TYPE ((NMLTYPE) 1605)
    a. (ich habe 1605 gewählt, weil es verfügbar war) nach emc2/src/emc/nml_intf/emc.hh
3. Fall EMC_IO_GENERIC_TYPE zu emcFormat in emc2/src/emc/nml_intf/emc.cc hinzufügen
4. Fall EMC_IO_GENERIC_TYPE zu emc_symbol_lookup in emc2/src/emc/nml_intf/emc.cc hinzufügen
5. EMC_IO_GENERIC::update-Funktion in emc2/src/emc/nml_intf/emc.cc hinzufügen

Neu kompilieren, und dieser neue Nachrichten-Typ sollte da sein. Der nächste Teil besteht darin, solche Nachrichten von irgendwoher zu senden und sie an einem anderen Ort zu empfangen und etwas damit anzufangen.

3.24 Tool Table and Toolchangeer

LinuxCNC Schnittstellen mit Werkzeugwechsler-Hardware, und hat eine interne Werkzeugwechsler Abstraktion. LinuxCNC verwaltet Werkzeuginformationen in einer Werkzeugtabelle.

3.24.1 Werkzeugwechsler Abstraktion in LinuxCNC

LinuxCNC unterstützt zwei Arten von Werkzeugwechsler Hardware, genannt *nonrandom* und *random*. Die INI-Einstellung [EMCIO]RANDOM_TOOLCHANGER steuert, mit welche dieser Arten von Hardware LinuxCNC denkt verbunden zu sein.

3.24.1.1 Nicht-zufällige Werkzeugwechsler

Die Hardware des nicht-zufälligen Werkzeugwechslers legt jedes Werkzeug in exakt die Tasche zurück, aus der es ursprünglich entnommen wurde.

Beispiele für nicht-zufällige Werkzeugwechsler sind der ”manuelle” Werkzeugwechsler, Drehautomaten und Regal-Werkzeugwechsler.

Wenn für einen nicht-zufälligen Werkzeugwechsler konfiguriert, ändert LinuxCNC die Platznummer in der Werkzeugtabellen-Datei nicht wenn Werkzeuge geladen bzw. entladen werden. LinuxCNC-intern werden bei einem Werkzeugwechsel die Werkzeug-Informationen von der Werkzeugtabelle **kopiert** von der jeweiligen Taschennummer der Quelle (der Ablage) zur Tasche mit der Nummer 0 (welche die Spindel darstellt). Dies ersetzt die vorherigen dort abgelegten Werkzeuginformationen.

Anmerkung
In einem für nicht-zufällige Werkzeugwechsler konfigurierten LinuxCNC hat Werkzeug 0 (T0) eine besondere Bedeutung: ”kein Werkzeug”. T0 kann nicht in der Werkzeugtabelle Datei erscheinen, und der Wechsel zu T0 wird in LinuxCNC verstehen, dass die Spindel kein Werkzeug führt.

3.24.1.2 Zufällige Werkzeugwechsler

Die Hardware des Zufallswerkzeugwechslers tauscht das Werkzeug in der Spindel (falls vorhanden) mit dem angeforderten Werkzeug beim Werkzeugwechsel aus. Somit ändert sich der Platz, in dem sich ein Werkzeug befindet, wenn es in die Spindel ein- und ausgewechselt wird.

Ein Beispiel für einen zufälligen Werkzeugwechsler ist ein Karussell-Werkzeugwechsler.

Wenn für ein zufälligen Werkzeugwechsler konfiguriert, tauscht LinuxCNC die Platznummer des alten und des neuen Werkzeugs in der Werkzeugtabellen-Datei, sobald Werkzeuge geladen werden. LinuxCNC-intern, werden bei einem Werkzeugwechsel die Werkzeug-Informationen zwischen der Werkzeugtabellen Tasche mit dem Werkzeug und Tasche 0 (welche die Spindel darstellt) ausgetauscht. Nach einem Werkzeugwechsel hat so Tasche 0 in der Werkzeugtabelle die Informationen zu einem neue Werkzeug und die Tasche, die das neue Werkzeug zuvor beherbergte, erhielt die Informationen für das alte Werkzeug (das Werkzeug, das in der Spindel vor dem Werkzeugwechsel war), wenn die Spindel überhaupt bestückt war.

Anmerkung
Wenn LinuxCNC für zufällige Werkzeugwechsler konfiguriert ist, hat Werkzeug 0 (T0) **keine** besondere Bedeutung. Es wird genau wie jedes andere Werkzeug in der Werkzeugtabelle behandelt. Es ist üblich, T0 zu verwenden, um ”kein Werkzeug” (d.h. ein Werkzeug mit Null TLO), so dass die Spindel bequem geleert werden kann.

3.24.2 Die Werkzeugtabelle

LinuxCNC verfolgt die Nutzung der Werkzeuge in einer sogenannten Werkzeug-Tabelle als Datei. Die Werkzeug-Tabelle zeichnet die folgenden Informationen für jedes Werkzeug:

Werkzeugnummer
Eine ganze Zahl, die dieses Werkzeug eindeutig identifiziert. Werkzeugnummern werden von LinuxCNC unterschiedlich behandelt, wenn sie für zufällige und nicht zufällige Werkzeugwechsler konfiguriert sind:

- Wenn LinuxCNC für einen nicht-zufälligen Werkzeugwechsler konfiguriert ist, muss diese Zahl positiv sein. T0 bekommt eine Sonderbehandlung und darf nicht in der Werkzeugtabelle erscheinen.
- Wenn LinuxCNC für einen zufälligen Werkzeugwechsler konfiguriert ist, muss diese Zahl nichtnegativ sein. T0 ist in der Werkzeugtabelle erlaubt, und wird in der Regel verwendet, um ”kein Werkzeug”, d.h. die leere Tasche darstellen.

Taschennummer
Eine ganze Zahl zur Identifikation der Tasche oder des Steckplatz in der Werkzeugwechsler-Hardware, in der sich das Werkzeug befindet. Pocket-Nummern werden unterschiedlich von LinuxCNC behandelt, abhängig davon ob für zufällige und nicht-zufällige Werkzeugwechsler konfiguriert:

- Wenn LinuxCNC für einen nicht-zufälligen Werkzeugwechsler konfiguriert ist, kann die Platznummer in der Werkzeugdatei jede positive ganze Zahl (Tasche 0 ist nicht erlaubt) sein. LinuxCNC verdichtet die Platznummern stillschweigend, wenn es die Werkzeugdatei lädt, so kann es einen Unterschied zwischen den Platznummern in der Werkzeugdatei und den internen Platznummern von LinuxCNC geben.
- Wenn LinuxCNC für einen zufälligen Werkzeugwechsler konfiguriert ist, müssen die Platznummern in der Werkzeugdatei zwischen 0 und 1000, einschließlich sein. Pockets 1-1000 sind im Werkzeugwechsler, Tasche 0 ist die Spindel.

Durchmesser
Durchmesser des Werkzeugs in Maschineneinheiten.

Werkzeuglängenkorrektur
Werkzeuglängenkorrektur (auch TLO genannt für engl. tool length offset) für bis zu 9 Achsen, in Maschineneinheiten. Achsen, die kein TLO haben, erhalten 0.

3.24.3 G-Codes mit Auswirkungen auf Werkzeuge

Die G-Codes, die Werkzeuginformationen verwenden oder beeinflussen, sind:

3.24.3.1 Txxx

Weist die Hardware des Werkzeugwechslers an, sich auf den Wechsel zu einem bestimmten Werkzeug xxx vorzubereiten.

Behandelt von `Interp::convert_tool_select()`.

1. Die Maschine wird aufgefordert, den Wechsel zum gewählten Werkzeug vorzubereiten, indem sie die Canon-Funktion `SELECT_TOOL()` mit der Werkzeugnummer des gewünschten Werkzeugs aufruft.

    a. (saicanon) ohne Funktion (no-op).

b. (emccanon) Erstellt eine EMC_TOOL_PREPARE Nachricht mit der angeforderten Taschennummer und sendet sie an Task, die sie an IO (E/A für Ein- und Ausgabe) weiterleitet. IO erhält die Nachricht und bittet HAL, die Tasche vorzubereiten, indem es iocontrol.0.tool-prep-pocket, iocontrol.0.tool-prep-number und iocontrol.0.tool-prepare setzt. IO ruft dann wiederholt read_tool_inputs() auf, um den HAL-Pin iocontrol.0.tool-prepared abzufragen, der von der Hardware des Werkzeugwechslers über HAL an IO signalisiert, dass die angeforderte Werkzeugvorbereitung abgeschlossen ist. Wenn dieser Pin auf True gesetzt wird, setzt IO emcioStatus.tool.pocketPrepped auf die angeforderte Werkzeugplatz-/taschennummer.

2. Zurück in interp, wird settings->selected_pocket dem tooldata-Index (engl. für Werkzeugdaten-Index) des angeforderten Werkzeugs *xxx* zugewiesen.

Anmerkung
Die inzwischen nicht mehr verwendeten Namen **selected_pocket** und **current_pocket** verweisen auf einen sequentiellen Werkzeugdatenindex für Werkzeugelemente, die aus einer Werkzeugtabelle ([EMCIO]TOOL_TABLE) oder über eine Werkzeugdatenbank ([EMCIO]DB_PROGRAM) geladen werden.

3.24.3.2 M6

Weist den Werkzeugwechsler an, zum aktuell ausgewählten Werkzeug zu wechseln (ausgewählt durch den vorherigen Befehl Txxx).

Wird von Interp::convert_tool_change() behandelt.

1. Die Maschine wird aufgefordert, zum ausgewählten Werkzeug zu wechseln, indem die Canon-Funktion CHANGE_TOOL() mit settings->selected_pocket (einem Werkzeugdatenindex) aufgerufen wird.

   a. (saicanon) Setzt sai's _active_slot auf die angegebene Platznummer. Die Werkzeuginformationen werden von der ausgewählten Tasche der Werkzeugtabelle (d.h. von sai's _tools[_activ zur Spindel (aka sai's _tools[0]) kopiert.

   b. (emccanon) Sendet eine EMC_TOOL_LOAD Nachricht an Task, die diese an IO sendet. IO setzt emcioStatus.tool.toolInSpindle auf die Werkzeugnummer des Werkzeugs in der Tasche, die durch emcioStatus.tool.pocketPrepped identifiziert wurde (gesetzt durch Txxx alias SELECT_TOOL()). Dann fordert es die Hardware des Werkzeugwechslers auf, einen Werkzeugwechsel durchzuführen, indem es den HAL-Pin iocontrol.0.tool-change auf True setzt. Später erkennt IO's read_tool_inputs(), dass der HAL-Pin iocontrol.0.tool_changed auf True gesetzt wurde, was anzeigt, dass der Werkzeugwechsler den Werkzeugwechsel abgeschlossen hat. Wenn dies der Fall ist, wird load_tool() aufgerufen, um den Maschinenzustand zu aktualisieren.

      i. load_tool() mit einer nicht-zufälligen Werkzeugwechsler-Konfiguration kopiert die Werkzeuginformationen von der ausgewählten Tasche auf die Spindel (Tasche 0).

      ii. load_tool() mit einer zufälligen Werkzeugwechsler-Konfiguration tauscht die Werkzeuginformationen zwischen Platz 0 (der Spindel) und dem ausgewählten Platz aus und speichert dann die Werkzeugtabelle.

2. Zurück in interp wird settings->current_pocket der neue Werkzeugdatenindex von settings->sele (gesetzt durch Txxx) zugewiesen. Die entsprechenden nummerierten Parameter (<sub:numbered-parameters,#5400-#5413>>) werden mit den neuen Werkzeuginformationen aus Tasche 0 (Spindel) aktualisiert.

3.24.3.3 G43/G43.1/G49

Werkzeuglängenkorrektur anwenden. G43 verwendet die TLO (engl. Abkürzung der Werkzeuglängenkorrektur) des aktuell geladenen Werkzeugs oder eines angegebenen Werkzeugs, wenn das H-Wort im Satz angegeben ist. G43.1 erhält die TLO von den Achsenwörtern im Satz. G49 hebt die TLO auf (es verwendet 0 für den Offset für alle Achsen).

Wird in `Interp::convert_tool_length_offset()` behandelt.

1. Es beginnt mit der Erstellung einer `EmcPose`, welche die zu verwendenden 9-Achsen-Versätze enthält. Bei `G43.1` stammen diese Werkzeugkorrekturen von den Achsenwörtern im aktuellen Satz. Bei `G43` stammen diese Versätze vom aktuellen Werkzeug (dem Werkzeug auf Platz 0) oder von dem durch das H-Wort im Satz angegebenen Werkzeug. Für G49 sind die Versätze alle 0.
2. Die Offsets werden an Canons Funktion `USE_TOOL_LENGTH_OFFSET()` übergeben.
   a. (saicanon) Zeichnet das TLO in `_tool_offset` auf.
   b. (emccanon) Erstellt eine Nachricht `EMC_TRAJ_SET_OFFSET` mit den Offsets und sendet sie an Task. Task kopiert die Offsets nach `emcStatus->task.toolOffset` und sendet sie über einen `EMCMOT_SET_OFFSET`-Befehl an Motion weiter. Motion kopiert die Offsets nach `emcmotStatus->tool` wo sie zum Offset für zukünftige Bewegungen verwendet werden.
3. Zurück in interp werden die Offsets in `settings->tool_offset` aufgezeichnet. Die effektive Tasche wird in `settings->tool_offset_index` aufgezeichnet, obwohl dieser Wert nie verwendet wird.

3.24.3.4 G10 L1/L10/L11

Ändert die Werkzeugtabelle.

Wird von `Interp::convert_setup_tool()` behandelt.

1. Wählt die Werkzeugnummer aus dem P-Wort im Block aus und findet die Tasche für dieses Werkzeug:
   a. Bei einer nicht-zufälligen Werkzeugwechsler-Konfiguration ist dies immer die Platznummer im Werkzeugwechsler (auch wenn sich das Werkzeug in der Spindel befindet).
   b. Bei einer zufälligen Werkzeugwechslerkonfiguration wird, wenn das Werkzeug gerade geladen ist, Platz 0 verwendet (Platz 0 bedeutet ”die Spindel”), und wenn das Werkzeug nicht geladen ist, wird die Platznummer im Werkzeugwechsler verwendet. (Dieser Unterschied ist wichtig.)
2. Ermittelt, wie die neuen Versätze aussehen sollen.
3. Die neuen Werkzeuginformationen (Durchmesser (engl. diameter), Versatz (engl. offset), Winkel (engl. angle) und Ausrichtung (engl. orientation)) werden zusammen mit der Werkzeugnummer und der Platznummer an den Canon-Aufruf SET_TOOL_TABLE_ENTRY() übergeben.
   a. (saicanon) Kopiert die neuen Werkzeuginformationen in die angegebene Tasche (in sai’s interne Werkzeugtabelle, `_tools`).
   b. (emccanon) Erstellt eine `EMC_TOOL_SET_OFFSET` Nachricht mit den neuen Werkzeuginformationen und sendet sie an Task, die sie an IO weitergibt. IO aktualisiert den angegebenen Platz in seiner internen Kopie der Werkzeugtabelle (`emcioStatus.tool.toolTable`), und wenn das angegebene Werkzeug gerade geladen ist (es wird mit `emcioStatus.tool.toolInSpindle` verglichen), werden die neuen Werkzeuginformationen auch in den Platz 0 (die Spindel) kopiert. (FIXME: das ist ein Buglet, sollte nur auf nicht-zufälligen Maschinen kopiert werden.) Schließlich speichert IO die neue Werkzeugtabelle.

4. Zurück in interp, wenn das geänderte Werkzeug gerade in der Spindel geladen ist und wenn die Maschine ein nicht-zufälliger Werkzeugwechsler ist, dann wird die neue Werkzeuginformation vom Stammplatz des Werkzeugs in den Platz 0 (die Spindel) in interp's Kopie der Werkzeugtabelle, `settings->tool_table` kopiert. (Diese Kopie wird auf Maschinen mit Zufallswerkzeugwechsler nicht benötigt, da die Werkzeuge dort keinen Stammplatz haben und wir stattdessen das Werkzeug in Platz 0 direkt aktualisieren.) Die relevanten nummerierten Parameter (#5400-#5413) werden aktualsiert von der Werkzeug-Information in der Spindel (durch Kopieren der Information von interps' `settings->tool_table` to `settings->parameters`)). (FIXME: Dies ist ein buglet, die Parameter sollten nur aktualisiert werden wenn das aktuelle Werkzeug modifiziert wurde).
5. Wenn das geänderte Werkzeug gerade in der Spindel geladen ist und für einen nicht zufälligen Werkzeugwechsler konfiguriert wurde, dann wird die neue Werkzeuginformation auch in den Platz 0 der Werkzeugtabelle geschrieben durch einen zweiten Aufruf von SET_TOOL_TABLE_ENTRY(). (Diese zweite Aktualisierung der Werkzeugtabelle wird bei Maschinen mit Zufallswerkzeugwechsler nicht benötigt, da die Werkzeuge dort keinen Stammplatz haben und wir stattdessen nur das Werkzeug in Platz 0 direkt aktualisieren.)

3.24.3.5 M61

Setze aktuelle Werkzeugnummer. Dies wechselt LinuxCNC's interne Darstellung, welches Werkzeug in der Spindel ist, ohne tatsächlich bewegen den Werkzeugwechsler oder Austausch von Werkzeugen.

Wird von `Interp::convert_tool_change()` behandelt.

Canon: `CHANGE_TOOL_NUMBER()`

settings->current_pocket wird dem Werkzeugdaten-Index zugewiesen, der das durch das Q-Wort-Argument angegebene Werkzeug enthält.

3.24.3.6 G41/G41.1/G42/G42.1

Aktiviert die Fräserradiuskompensation (engl. kurz *cutter comp* genannt).

Wird von `Interp::convert_cutter_compensation_on()` behandelt.

Kein Canon-Aufruf, die Fräser Kompensation erfolgt im Interpreter. Verwendet die Werkzeugtabelle in der erwarteten Weise: Wenn ein D-Wort als Werkzeugnummer angegeben wird, so wird die Platznummer der angegebenen Werkzeugnummer in der Tabelle nachgeschlagen, und wenn kein D-Wort angegeben wird, dann wird Platz 0 (die Spindel) verwendet.

3.24.3.7 G40

Fräserradiuskompensation aufheben.

Wird von `Interp::convert_cutter_compensation_off()` behandelt.

Kein Canon-Aufruf, die Fräser Kompensation erfolgt im Interpreter. Verwendet nicht die Werkzeugtabelle.

3.24.4 Interne Zustandsvariablen

Dies ist keine erschöpfende Liste! Werkzeug-Informationen sind in LinuxCNC verteilt abgelegt.

3.24.4.1 E/A (engl. I/O)

`emcioStatus` ist vom Typ `EMC_IO_STAT`

emcioStatus.tool.pocketPrepped
Wenn IO das Signal von HAL erhält, dass die Vorbereitung des Werkzeugwechslers abgeschlossen ist (nach einem `Txxx` Befehl), wird diese Variable auf den Platz des angeforderten Werkzeugs gesetzt. Wenn IO das Signal von HAL erhält, dass der Werkzeugwechsel selbst abgeschlossen ist (nach einem `M6` Befehl), wird diese Variable auf -1 zurückgesetzt.

emcioStatus.tool.toolInSpindle
Werkzeugnummer des aktuell in der Spindel installierten Werkzeugs. Wird über den HAL-Pin `iocontrol.0.tool-number` (s32) exportiert.

emcioStatus.tool.toolTable[]
Ein Array von `CANON_TOOL_TABLE` Strukturen, `CANON_POCKETS_MAX` lang. Wird beim Starten aus der Werkzeugtabellendatei geladen und danach beibehalten. Index 0 ist die Spindel, die Indizes 1-(CANON_POCKETS_MAX-1) sind die Plätze im Werkzeugwechsler. Dies ist eine vollständige Kopie der Werkzeuginformationen, die getrennt von Interp's `settings.tool_table` gepflegt wird.

3.24.4.2 interp

`settings` ist vom Typ `settings`, definiert als `struct setup_struct` ist in `src/emc/rs274ngc/interp_inter`

settings.selected_pocket
Tooldatenindex des zuletzt mit `Txxx` ausgewählten Werkzeugs.

settings.current_pocket
Ursprünglicher Werkzeugdatenindex des Werkzeugs, das sich gerade in der Spindel befindet. Mit anderen Worten: Von welchem Werkzeugdatenindex das Werkzeug geladen wurde, das sich gerade in der Spindel befindet.

settings.tool_table[]
Ein Array mit Werkzeuginformationen. Der Index im Array ist die "Platznummer" (auch "Slotnummer" genannt). Platz 0 ist die Spindel, die Plätze 1 bis (CANON_POCKETS_MAX-1) sind die Plätze des Werkzeugwechslers.

settings.tool_offset_index
Unbenutzt. FIXME: Sollte wahrscheinlich entfernt werden.

settings.toolchange_flag
Interp setzt dies auf true, wenn die Funktion CHANGE_TOOL() von Canon'aufgerufen wird. Er wird in `Interp::convert_tool_length_offset()` überprüft, um zu entscheiden, welcher Werkzeugdatenindex für G43 (ohne H-Wort) zu verwenden ist: `settings->current_pocket` wenn der Werkzeugwechsel noch im Gange ist, Werkzeugdaten-Index 0 (die Spindel), wenn der Werkzeugwechsel abgeschlossen ist.

settings.random_toolchanger
Wird beim Starten über die INI-Variable `[EMCIO]RANDOM_TOOLCHANGER` gesetzt. Steuert verschiedene Logiken zur Handhabung von Werkzeugtabellen. (IO liest auch diese INI-Variable und ändert sein Verhalten auf der Grundlage dieser Variable. Wenn beispielsweise die Werkzeugtabelle gespeichert wird, so speichert der Zufallswerkzeugwechsler das Werkzeug in der Spindel (Platz 0), während der Nicht-Zufallswerkzeugwechsler jedes Werkzeug in seinem Stammplatz speichert.)

settings.tool_offset
Dies ist eine `EmcPose` Variable.

- Dient zur Berechnung der Position an verschiedenen Orten.
- Wird über die Nachricht `EMCMOT_SET_OFFSET` an Motion gesendet. Alles, was Motion mit den Offsets macht, ist, sie in die HAL-Pins `motion.0.tooloffset.[xyzabcuvw]` zu exportieren. FIXME: die Offsets von einem Ort exportieren, der näher an der Werkzeugtabelle liegt (wahrscheinlich io oder interp) und entfernen der EMCMOT_SET_OFFSET-Nachricht.

Einstellungen. pockets_max
Wird austauschbar mit `CANON_POCKETS_MAX` verwendet (eine #definierte Konstante, ab April 2020 auf 1000 gesetzt). FIXME: Diese Einstellungsvariable ist derzeit nicht nützlich und sollte wahrscheinlich entfernt werden.

settings.tool_table
Dies ist ein Array von `CANON_TOOL_TABLE` Strukturen (definiert in `src/emc/nml_intf/emctool.h`), mit `CANON_POCKETS_MAX` Einträgen. Indiziert durch "Taschen-Nummer", auch bekannt als "Slot-Nummer". Index 0 ist die Spindel, die Indizes 1 bis (CANON_POCKETS_MAX-1) sind die Plätze im Werkzeugwechsler. Bei einem Zufallswerkzeugwechsler sind die Platznummern sinnvoll. Bei einem nicht-zufälligen Werkzeugwechsler sind die Platznummern bedeutungslos; die Platznummern in der Werkzeugtabellendatei werden ignoriert und die Werkzeuge werden den `tool_table` Plätzen nacheinander zugewiesen.

settings.tool_change_at_g30 , settings.tool_change_quill_up , settings.tool_change_with_spindle_

Diese werden über INI-Variablen im Abschnitt `[EMCIO]` gesetzt und bestimmen, wie Werkzeugwechsel durchgeführt werden.

3.25 Parameter-Bestimmung von Gelenken und Achsen

3.25.1 Im Statuspuffer

Der Statuspuffer wird vom Task-Modul und den Benutzeroberflächen verwendet.

FIXME: `axis_mask` und `axes` überspezifizieren die Anzahl der Achsen

`status.motion.traj.axis_mask`
Eine Bitmaske mit einem "1" für die Achsen, die vorhanden sind, und einem "0" für die Achsen, die nicht vorhanden sind. X ist das niederwertigste Bit 0 mit Wert $2^0 = 1$ if set , Y ist Bit 1 mit Wert $2^1 = 2$, Z ist Bit 2 mit Wert $2^2 = 4$ usw. Eine Maschine mit X- und Z-Achsen hätte zum Beispiel eine `axis_mask` von 0x5, eine XYZ-Maschine hätte 0x7 und eine XYZB-Maschine hätte eine `axis_mask` von 0x17.

`status.motion.traj.axes` (entfernt)
Dieser Wert wurde in LinuxCNC Version 2.9 entfernt. Verwenden Sie stattdessen `axis_mask`.

`status.motion.traj.joints`
Zählt die Anzahl der Gelenke einer Maschine. Eine normale Drehmaschine hat 2 Gelenke; eines treibt die X-Achse und eines die Z-Achse an. Eine XYYZ-Portalfräse hat 4 Gelenke: eines für die X-Achse, eines für eine Seite der Y-Achse, eines für die andere Seite der Y-Achse und eines für die Z-Achse. Eine XYZA-Fräse hat ebenfalls 4 Gelenke.

`status.motion.axis[EMCMOT_MAX_AXIS]`
Eine Reihe von `EMCMOT_MAX_AXIS`-Achsenstrukturen. axis[n] ist gültig, wenn `(axis_mask & (1 << n))` Wahr ist. Wenn `(axis_mask & (1 << n))` False ist, dann existiert `axis[n]` nicht auf dieser Maschine und muss ignoriert werden.

`status.motion.joint[EMCMOT_MAX_JOINTS]`
Ein Array von `EMCMOT_MAX_JOINTS` Gelenkstrukturen (engl. joint structures). `joint[0]` bis `joint[joint`
sind gültig, die anderen existieren auf diesem Rechner nicht und müssen ignoriert werden.

Im Joints-Axes-Entwicklungszweig sind die Dinge derzeit nicht so, aber Abweichungen von diesem Design werden als Fehler betrachtet. Ein Beispiel für einen solchen Fehler ist die Behandlung von Achsen in src/emc/ini/initraj.cc:loadTraj(). Es gibt zweifellos noch mehr, und ich brauche Ihre Hilfe, um sie zu finden und zu beheben.

3.25.2 In Bewegung

Die Echtzeitkomponente des Motion Controllers erhält zunächst die Anzahl der Joints aus dem Load-Time-Parameter `num_joints`. Dieser bestimmt, wie viele Gelenke mit HAL-Pins beim Start erzeugt werden.

Die Anzahl der Gelenke einer Bewegung kann zur Laufzeit mit dem Befehl `EMCMOT_SET_NUM_JOINTS` aus dem Task heraus geändert werden.

Der Motion Controller arbeitet immer mit `EMCMOT_MAX_AXIS`-Achsen. Er erstellt immer neun Sätze von ”Achsen.**.**”-Pins.

Kapitel 4

NML-Nachrichten

Liste der NML-Nachrichten.
Für Einzelheiten siehe *src/emc/nml_intf/emc.hh*.

4.1 OPERATOR (engl. für Bediener)

```
EMC_OPERATOR_ERROR_TYPE
EMC_OPERATOR_TEXT_TYPE
EMC_OPERATOR_DISPLAY_TYPE
```

4.2 JOINT (engl. für Gelenk)

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

4.3 ACHSE

```
EMC_AXIS_STAT_TYPE
```

4.4 JOG

```
EMC_JOG_CONT_TYPE
EMC_JOG_INCR_TYPE
EMC_JOG_ABS_TYPE
EMC_JOG_STOP_TYPE
```

4.5 TRAJ (engl. Kurzform für Trajektorie)

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

4.6 MOTION (engl. für Bewegung)

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

4.7 TASK (engl. für Aufgabe, auch Name des entsprechenden LinuxCNC Moduls)

```
EMC_TASK_INIT_TYPE
EMC_TASK_HALT_TYPE
EMC_TASK_ABORT_TYPE
EMC_TASK_SET_MODE_TYPE
EMC_TASK_SET_STATE_TYPE
EMC_TASK_PLAN_OPEN_TYPE
EMC_TASK_PLAN_RUN_TYPE
EMC_TASK_PLAN_READ_TYPE
EMC_TASK_PLAN_EXECUTE_TYPE
EMC_TASK_PLAN_PAUSE_TYPE
EMC_TASK_PLAN_STEP_TYPE
EMC_TASK_PLAN_RESUME_TYPE
EMC_TASK_PLAN_END_TYPE
EMC_TASK_PLAN_CLOSE_TYPE
EMC_TASK_PLAN_INIT_TYPE
EMC_TASK_PLAN_SYNCH_TYPE
EMC_TASK_PLAN_SET_OPTIONAL_STOP_TYPE
EMC_TASK_PLAN_SET_BLOCK_DELETE_TYPE
EMC_TASK_PLAN_OPTIONAL_STOP_TYPE
EMC_TASK_STAT_TYPE
```

4.8 TOOL (engl. für Werkzeug)

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

4.9 AUX (engl. Kurzform für ”andere Hilfsfunktionen”)

```
EMC_AUX_ESTOP_ON_TYPE
EMC_AUX_ESTOP_OFF_TYPE
EMC_AUX_ESTOP_RESET_TYPE
```

```
EMC_AUX_INPUT_WAIT_TYPE
EMC_AUX_STAT_TYPE
```

4.10 SPINDLE (engl. für Spindel)

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

4.11 COOLANT (engl. für Kühlflüssigkeit)

```
EMC_COOLANT_MIST_ON_TYPE
EMC_COOLANT_MIST_OFF_TYPE
EMC_COOLANT_FLOOD_ON_TYPE
EMC_COOLANT_FLOOD_OFF_TYPE
EMC_COOLANT_STAT_TYPE
```

4.12 LUBE (engl. für Schmiermittel)

```
EMC_LUBE_ON_TYPE
EMC_LUBE_OFF_TYPE
EMC_LUBE_STAT_TYPE
```

4.13 IO (engl. kurz für Eingabe/Ausgabe - Input/Output, auch Name eines LinuxCNC Moduls)

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

4.14 Andere

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

Kapitel 5

Quellcode-Stil

Dieses Kapitel beschreibt den vom LinuxCNC-Team bevorzugten Quellcode-Stil.

5.1 Keinen Schaden anrichten

Wenn Sie kleine Änderungen am Code in einem anderen als dem unten beschriebenen Stil vornehmen, beachten Sie den lokalen Kodierungsstil. Schnelle Wechsel von einem Kodierungsstil zum anderen beeinträchtigen die Lesbarkeit des Codes.

Checken Sie niemals Code ein, nachdem Sie ihn mit einem Hilfsprogramm wie ”indent” bearbeitet haben. Die durch die Einrückung verursachten Änderungen an Leerzeichen erschweren es, den Revisionsverlauf der Datei zu verfolgen.

Verwenden Sie keinen Editor, der unnötige Änderungen am Leerraum vornimmt (z. B. 8 Leerzeichen durch einen Tabstopp in einer ansonsten nicht geänderten Zeile ersetzt oder Zeilen umbricht, die ansonsten nicht geändert werden).

5.2 Tabstopps

Ein Tabstopp entspricht immer 8 Leerzeichen. Schreiben Sie keinen Code, der nur bei einer abweichenden Einstellung des Tabstopps korrekt angezeigt wird.

5.3 Einrückung

Verwenden Sie 4 Leerzeichen pro Einrückungsebene. Die Kombination von 8 Leerzeichen in einem Tabulator ist zulässig, aber nicht erforderlich.

5.4 Setzen von Klammern

Setzen Sie die öffnende Klammer zuletzt auf die Linie, und setzen Sie die schließende Klammer zuerst:

```
if (x) {
    // macht irgendetwas der Situation Angepasstes
}
```

Die schließende Klammer steht in einer eigenen Zeile, es sei denn, es folgt eine Fortsetzung derselben Anweisung, z. B. ein *while* in einer do-Anweisung oder ein *else* in einer if-Anweisung, wie hier:

```
do {
    // etwas Wichtiges
} while (x > 0);
```

und

```
if (x == y) {
    // tue das eine
} else if (x < y) {
    // tue das andere
} else {
    // tue ein Drittes
}
```

Durch diese Klammerung wird auch die Anzahl der leeren (oder fast leeren) Zeilen minimiert, wodurch eine größere Menge an Code oder Kommentaren auf einmal in einem Terminal mit fester Größe sichtbar ist.

5.5 Benennung

C ist eine spartanische Sprache, und so sollte auch Ihre Namensgebung sein. Anders als Modula-2- und Pascal-Programmierer verwenden C-Programmierer keine niedlichen Namen wie DiesIstEineTemporaereZaehlvariable. Ein C-Programmierer würde diese Variable *tmp* nennen, was viel einfacher zu schreiben und nicht im Geringsten schwieriger zu verstehen ist.

Beschreibende Namen für globale Variablen sind jedoch ein Muss. Eine globale Funktion *foo* zu nennen, wäre ein offensichtliches, schweres Vergehen.

GLOBALE Variablen (die nur verwendet werden, wenn man sie **wirklich** braucht) müssen beschreibende Namen haben, ebenso wie globale Funktionen. Wenn Sie eine Funktion haben, um die Anzahl der aktiven Benutzer zu zählen, so sollte sie *count_active_users()* oder ähnlich heißen, **nicht** *cntusr()*.

Den Typ einer Funktion in den Namen zu kodieren (so genannte ungarische Notation) ist hirnrissig - der Compiler kennt die Typen ohnehin und kann sie überprüfen, und es verwirrt den Programmierer nur. Kein Wunder, dass Microsoft fehlerhafte Programme macht.

LOCAL-Variablennamen sollten kurz und prägnant sein. Wenn Sie einen zufälligen ganzzahligen Schleifenzähler haben, sollte er wahrscheinlich *i* heißen. Der Name *loop_counter* ist unproduktiv, wenn keine Gefahr besteht, dass er missverstanden wird. In ähnlicher Weise kann *tmp* so gut wie jede Art von Variable sein, die für einen temporären Wert verwendet wird.

Wenn Sie befürchten, Ihre lokalen Variablennamen zu verwechseln, dann haben Sie ein anderes Problem, das als Funktions-Wachstums-Hormon-Gleichgewichts-Syndrom bezeichnet wird. Siehe nächstes Kapitel.

5.6 Funktionen

Funktionen sollten kurz und knapp sein und nur einen Zweck erfüllen. Sie sollten auf einen oder zwei Bildschirme voller Text passen (die ISO/ANSI-Bildschirmgröße beträgt 80x24, wie wir alle wissen) und nur eine Aufgabe erfüllen, und diese gut.

Die maximale Länge einer Funktion ist umgekehrt proportional zur Komplexität und zum Grad der Einrückung dieser Funktion. Wenn Sie also eine konzeptionell einfache Funktion haben, die nur aus

einer einzigen langen (aber einfachen) Fallanweisung besteht, bei der Sie viele kleine Dinge für viele verschiedene Fälle tun müssen, ist es in Ordnung, eine längere Funktion zu haben.

Wenn Sie jedoch eine komplexe Funktion haben und Sie vermuten, dass ein weniger begabter Anfänger nicht einmal versteht, worum es in der Funktion geht, sollten Sie sich um so mehr an die Höchstgrenzen halten. Verwenden Sie Hilfsfunktionen mit aussagekräftigen Namen (Sie können den Compiler bitten, die Funktion inline einzubinden, wenn Sie es für leistungsrelevant halten, und er wird es wahrscheinlich besser machen, als Sie es getan hätten).

Ein weiteres Maß für die Funktion ist die Anzahl der lokalen Variablen. Sie sollte 5-10 nicht überschreiten, oder Sie machen etwas falsch. Überdenken Sie die Funktion, und teilen Sie sie in kleinere Teile auf. Ein menschliches Gehirn kann in der Regel problemlos 7 verschiedene Dinge im Auge behalten, alles darüber hinaus verwirrt es. Sie wissen, dass Sie brillant sind, aber vielleicht würden Sie in zwei Wochen gerne auch noch verstehen, was Sie getan haben.

5.7 Kommentieren

Kommentare sind gut, aber es besteht auch die Gefahr, dass man zu viel kommentiert. Versuchen Sie NIEMALS, in einem Kommentar zu erklären, WIE Ihr Code funktioniert: es ist viel besser, den Code so zu schreiben, dass die **Arbeitsweise** offensichtlich ist, und es ist Zeitverschwendung, schlecht geschriebenen Code zu erklären.

Im Allgemeinen sollen Ihre Kommentare sagen, WAS Ihr Code tut, nicht WIE. Ein umrahmter Kommentar, der die Funktion, den Rückgabewert und den Aufrufer beschreibt und über dem Hauptteil steht, ist gut. Versuchen Sie auch, Kommentare innerhalb eines Funktionskörpers zu vermeiden: Wenn die Funktion derart komplex ist, dass Sie Teile davon gesondert kommentieren müssen, sollten Sie vielleicht den Abschnitt Funktionen noch einmal lesen. Sie können kleine Kommentare machen, um auf etwas besonders Kluges (oder Hässliches) hinzuweisen oder zu warnen, aber versuchen Sie, Übertreibungen zu vermeiden. Stellen Sie die Kommentare stattdessen an den Anfang der Funktion und sagen Sie, was sie tut und möglicherweise WARUM sie es tut.

Wenn Kommentare in der Art von /* Fix me */ (engl. für ”Reparier mich”) verwendet werden, dann beschreiben Sie bitte auch, warum etwas repariert werden muss. Wenn eine Änderung an dem betroffenen Teil des Codes vorgenommen wurde, entfernen Sie entweder den Kommentar oder ergänzen Sie ihn, um anzugeben, dass eine Änderung vorgenommen wurde und getestet werden muss.

5.8 Shell-Skripte & Makefiles

Nicht jeder hat die gleichen Werkzeuge und Pakete installiert. Einige Leute benutzen vi, andere emacs - Einige wenige vermeiden es sogar, eines der beiden Pakete zu installieren und bevorzugen einen leichtgewichtigen Texteditor wie nano oder den in Midnight Commander integrierten.

gawk versus mawk - Auch hier wird nicht jeder gawk installiert haben, mawk ist fast ein Zehntel so groß und entspricht dennoch dem POSIX AWK Standard. Wenn ein obskurer gawk-spezifischer Befehl benötigt wird, den mawk nicht zur Verfügung stellt, wird das Skript für einige Benutzer nicht funktionieren. Das Gleiche würde für mawk gelten. Kurz gesagt, verwenden Sie lieber den allgemeinen awk-Aufruf als gawk oder mawk.

5.9 C++-Konventionen

C++-Codierungsstile sind wiederholt Anlass für hitzige Debatten (ähnlich wie der Streit zwischen emacs und vi). Eines ist jedoch sicher: Ein gemeinsamer Stil, der von allen an einem Projekt Beteiligten verwendet wird, führt zu einheitlichem und lesbarem Code.

Namenskonventionen: Konstanten entweder aus #defines oder aus Enumerationen sollten durchgehend in Großbuchstaben geschrieben werden. Begründung: Erleichtert das Erkennen von durch den Präprozessor substituierten Ausdrücke im Quellcode, z.B. EMC_MESSAGE_TYPE.

Klassen und Namensräume (engl. namespace) sollten den ersten Buchstaben eines jeden Wortes groß schreiben und Unterstriche vermeiden. Begründung: Identifiziert Klassen, Konstruktoren und Destruktoren, z.B. GtkWidget.

Methoden (oder Funktionsnamen) sollten den obigen C-Empfehlungen folgen und nicht den Klassennamen enthalten. Begründung: Beibehaltung eines gemeinsamen Stils für C- und C++-Quellen, z.B. get_foo_bar().

Hingegen sind Boolesche Methoden leichter zu lesen, wenn sie keine Unterstriche und ein *is* Präfix verwenden (nicht zu verwechseln mit Methoden, die einen Booleschen Wert manipulieren). Begründung: Kennzeichnet den Rückgabewert als TRUE oder FALSE und nichts anderes, z.B. isOpen, isHomed.

Verwenden Sie NICHT *Not* in einem booleschen Namen, es führt nur zu Verwirrung, wenn Sie logische Tests durchführen, z.B. isNotOnLimit oder is_not_on_limit sind BÖSE.

Bei Variablennamen sollte die Verwendung von Großbuchstaben und Unterstrichen vermieden werden, außer bei lokalen oder privaten Namen. Die Verwendung von globalen Variablen sollte so weit wie möglich vermieden werden. Begründung: Stellt klar, was Variablen und was Methoden sind. Öffentlich (engl. public): z.B. axislimit Privat: z.B. maxvelocity_ .

5.9.1 Spezifische Namenskonventionen für Methoden

Die Begriffe get und set sollten verwendet werden, wenn direkt auf ein Attribut zugegriffen wird. Begründung: Gibt den Zweck der Funktion oder Methode an, z. B. get_foo set_bar.

Für Methoden, die boolesche Attribute beinhalten, ist set & reset vorzuziehen. Begründung: Wie oben. z. B. set_amp_enable reset_amp_fault

Bei rechenintensiven Methoden sollte ”compute” als Präfix verwendet werden. Begründung: Zeigt an, dass die Methode rechenintensiv ist und die CPU beansprucht. z.B. compute_PID

Abkürzungen in Namen sollten nach Möglichkeit vermieden werden - eine Ausnahme bilden die Namen lokaler Variablen. Begründung: Klarheit des Codes. z.B. Pointer wird gegenüber ptr bevorzugt compute wird gegenüber cmp bevorzugt und genauso wird compare wiederum gegenüber cmp vorgezogen.

Aufzählungen und anderen Konstanten kann ein gemeinsamer Typname vorangestellt werden, z. B. `enum COLOR { COLOR_RED, COLOR_BLUE };`.

Übermäßiger Gebrauch von Makros und Defines sollte vermieden werden - Die Verwendung einfacher Methoden oder Funktionen ist vorzuziehen. Begründung: Verbessert den Debugging-Prozess.

Include-Anweisungen Header-Dateien müssen am Anfang einer Quelldatei eingefügt werden und dürfen nicht über den gesamten Textkörper verstreut sein. Sie sollten nach ihrer hierarchischen Position innerhalb des Systems sortiert und gruppiert werden, wobei die Dateien der unteren Ebene zuerst eingebunden werden sollten. Include-Dateipfade sollten NIEMALS absolut sein - verwenden Sie stattdessen das Compiler-Flag -I zur Erweiterung des Suchpfades. Begründung: Header befinden sich nicht auf allen Systemen an der gleichen Stelle.

Bei Zeigern und Referenzen sollte das Referenzsymbol neben dem Variablennamen und nicht neben dem Typnamen stehen. Begründung: Verringert die Verwirrung, z.B. `float *x` oder `int &i` .

Implizite Tests auf Null sollten mit Ausnahme von booleschen Variablen nicht verwendet werden, z.B. `if (spindle_speed != 0)`, NICHT `if (spindle_speed)`.

In einem for()-Konstrukt dürfen nur Anweisungen zur Schleifenkontrolle enthalten sein, z.B. `sum = 0; for (i = 0; i < 10; i++) { sum += value[i]; }` `+ NICHT: for (i=0, sum=0; i<10; i++) sum += value[i];`.

Ebenso müssen ausführbare Anweisungen in Konditionalen vermieden werden, z.B. ist `if (fd = open(file_name))` böse.

Komplexe bedingte Anweisungen sollten vermieden werden - führen Sie stattdessen temporäre boolesche Variablen ein.

Klammern sollten in mathematischen Ausdrücken reichlich verwendet werden - Verlassen Sie sich nicht auf den Vorrang von Operatoren, wenn eine zusätzliche Klammer die Dinge klarer darstellen würde.

Dateinamen: C++-Quelltexte und -Header verwenden die Erweiterungen .cc und .hh. Die Verwendung von .c und .h ist für einfaches C reserviert. Header sind für Klassen-, Methoden- und Strukturdeklarationen, nicht für Code (es sei denn, die Funktionen sind inline deklariert).

5.10 Python-Codierungsstandards

Verwenden Sie den PEP 8-Stil für Python-Code.

5.11 Comp-Codierungs-Standards

Im Deklarationsteil einer .comp-Datei beginnen Sie jede Deklaration mit der ersten Spalte. Fügen Sie zusätzliche Leerzeilen ein, wenn sie zur Gruppierung verwandter Elemente beitragen.

Im Code-Teil einer .comp-Datei ist der normale C-Codierungsstil zu beachten.

Kapitel 6

Kompilieren (”bauen”) von LinuxCNC

6.1 Einführung

Dieses Dokument beschreibt, wie man die ausführbare LinuxCNC Software aus dem Quellcode erstellt. Dies ist vor allem nützlich, wenn Sie ein Entwickler sind, der LinuxCNC modifiziert. Es kann auch nützlich sein, wenn Sie ein Benutzer sind, der Entwickler-Zweige testet, obwohl Sie dann auch die Möglichkeit haben, einfach Debian-Pakete aus dem Buildbot zu installieren (http://buildbot.linuxcnc.org) oder als ein normales Paket aus Ihrer Linux Distribution (https://tracker.debian.org/pkg/linuxcnc). Zugegeben, diese Erleuterungen sollen Sie auch dazu ermutigen, zur Entwicklung von LinuxCNC beizutragen. Das Projekt lebt allein aus den Beiträgen seiner Anwender. * zu einer weiteren Entwicklung von LinuxCNC oder * einer weiteren Entwicklung, die Sie vielleicht zu LinuxCNC beisteuern möchten oder anderen helfen diese zu finalisieren.

Beispielsweise portieren Sie vielleicht LinuxCNC zu einer neuen Linux Distribution oder, und dies geschieht häufig, ein Entwickler reagiert zu einem von Ihnen berichteten Problem und bittet Sie, seinen Fix zu testen. Eine solche Veränderung wird von keinem buildbot gesehen, oder deren Hilfe trifft verzögert eine, nachdem jemand anderes die Änderung durchgesehen hat. Darauf möchten Sie aber nicht warten oder die anderen hat nicht die exakt gleiche Hardware, um den veränderten Code zu testen.

Neben dem Programm zur Steuerung der Maschine, die vom Quellbaum gebaut werden, können Sie auch die gleichen PDFs und/oder HTML Dateien bauen, denen Sie vermutlich bereits online bei https://linuxcnc.org/documents begegnet sind.

Wenn Sie zu LinuxCNC beitragen möchten aber unsicher sind wo Sie anfangen sollen, dann denken Sie bitte einmal allen Ernstes darüber nach, vielleicht bei der Dokumentation anzufangen. Jeder findet dort etwas zu verbessern - und wenn Sie nur ein ”FIXME: und noch ein Kommentar dazu” im Text als Referenz für sich selber und andere hinterlassen, um diesen Abschnitt später nochmal anzuschauen. Auch werden die bereits vorgenommenen Übersetzungen zu anderen Sprachen als Englisch sehr Wahrscheinlich von Ihrer Durchsicht profitieren bei https://hosted.weblate.org/projects/linuxcnc/ .

6.2 Herunterladen des Quellcodes

Das LinuxCNC Projekt git Repository ist zugänglich über https://github.com/LinuxCNC/linuxcnc. GitHub ist ein beliebter git Hosting-Service und Code-Sharing-Website.

Um den Quellcode herunterzuladen haben Sie zwei Optionen:

Herunterladen einer komprimierten .tar-Datei, im Englischen gern ”tarball” genannt
Auf der LinuxCNC-Projektseite in GitHub finden Sie einen Verweis auf die ”Releases” oder ”Tags”,

folgen Sie auf diesen Hyperlink zur Archivseite und laden Sie die neueste .tar-Datei herunter. Diese Datei wird als .tar.xz oder .tar.gz-Datei komprimiert angeboten. Diese Datei, allgemein als ”Tarball” bezeichnet, ist ein Archiv sehr analog zu einer .zip-Datei. Ihr Linux-Desktop wird wissen, wie Sie diese Datei behandeln, wenn Sie auf sie doppelklicken.

Erstellen Sie eine lokale Kopie des LinuxCNC-Repository
Sie würden zunächst das Programm ”git” auf Ihrem Rechner installieren, wenn es nicht bereits verfügbar ist (`sudo apt install git`). Dann bereiten Sie eine lokale Instanz des Quellbaums wie folgt vor: .

```
$ git clone https://github.com/LinuxCNC/linuxcnc.git linuxcnc-source-dir
```

. Das erste Argument des git Befehls nimmt es vorweg: Dies wird als ”Klon” (engl. clone) des LinuxCNC-Repositorys bezeichnet. Der Vorteil ist, dass dieser lokale Klon die Kommunikation über Änderungen unterstützt, die Sie am Quellbaum vornehmen.

GitHub ist eine eigene Infrastruktur und im Detail anderswo erklärt. Nur um Sie motiviert zu bekommen, wenn Sie es nicht schon wissen, GitHub bietet an einen Klon des LinuxCNC in einem allein Ihnen zugeordneten Bereich anzulegen und diese Instanz öffentlich zugänglich zu machen. GitHub nennt eine solche zusätzliche Instanz eines anderen Repositorys einen ”Fork” (engl. für ”Abzweigung”, wird aber auch im Deutschen genutzt). Sie können leicht (und ohne Kosten) einen solchen Fork des LinuxCNC git Repository bei GitHub erstellen und diesen verwenden, um Ihre Änderungen zu verfolgen und zu veröffentlichen. Nach der Erstellung Ihres eigenen GitHub Fork von LinuxCNC, klonen Sie diesen auf Ihrem Entwicklungs-Rechner und fahren Sie mit Ihrer Entwicklungsarbeit fort wie gewohnt.

Wir vom LinuxCNC-Projekt hoffen, dass Sie Ihre Änderungen mit uns teilen werden, damit die Gemeinschaft von Ihrer Arbeit profitieren kann. GitHub macht dieses Teilen sehr einfach: Nachdem Sie Ihre Änderungen aufpoliert und in Ihren Github-Fork gepusht haben, schicken Sie uns einen Pull Request.

6.2.1 Schnellstart

Wer ungeduldig ist, kann dies versuchen:

```
$ git clone https://github.com/LinuxCNC/linuxcnc.git linuxcnc-source-dir
$ cd linuxcnc-source-dir/src
$ ./autogen.sh
$ ./configure --with-realtime=uspace
$ make
```

Das wird wahrscheinlich schiefgehen! Das macht Sie nicht zu einem schlechten Menschen, es bedeutet nur, dass Sie dieses ganze Dokument lesen sollten, um herauszufinden, wie Sie Ihre Probleme lösen können. Insbesondere den Abschnitt über <Satisfying-Build-Dependencies,Bedienen von Build Abhängigkeiten (engl. dependencies)>>.

Wenn Sie auf einem Echtzeit-fähigen System arbeiten (wie z.B. eine Installation aus dem LinuxCNC Live/Install Image, siehe den Echtzeit Abschnitt unten), ist ein zusätzlicher Build-Schritt zu diesem Zeitpunkt erforderlich:

```
>$ sudo make setuid
```

Nachdem Sie erfolgreich LinuxCNC gebaut haben, ist es nun Zeit, die Tests durchzuführen:

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

Auch das kann fehlschlagen! Lesen Sie das gesamte Dokument, insbesondere den Abschnitt Einrichten der Testumgebung.

6.3 Unterstützte Plattformen

Das LinuxCNC-Projekt orientiert sich an moderne Debian-basierte Distributionen, einschließlich Debian, Ubuntu und Mint. Wir testen ständig auf den unter http://buildbot.linuxcnc.org aufgeführten Plattformen.

LinuxCNC baut auf die meisten anderen Linux-Distributionen, obwohl die Verwaltung von Abhängigkeiten mehr manuell und weniger automatisch sein wird. Patches zur Verbesserung der Portabilität auf neue Plattformen sind immer willkommen.

6.3.1 Echtzeit

LinuxCNC ist eine Werkzeugmaschinensteuerung, und es erfordert eine Echtzeit-Plattform, um diesen Anforderungen gerechtzuwerden. Diese Version von LinuxCNC unterstützt die folgenden Plattformen. Die ersten drei aufgeführten sind Echtzeit-Plattformen:

RTAI
: Von https://www.rtai.org. Ein Linux-Kernel mit dem RTAI-Patch ist aus dem Debian-Archiv unter https://linuxcnc.org erhältlich. Siehe LinuxCNC erhalten für Installationsanweisungen.

Xenomai
: Von https://xenomai.org. Sie müssen einen Xenomai-Kernel selbst kompilieren oder beziehen.

Preempt-RT
: Von https://rt.wiki.kernel.org. Ein Linux-Kernel mit dem Preempt-RT-Patch ist gelegentlich im Debian-Archiv unter https://www.debian.org und in der Wayback Machine unter https://snapshot.debian verfügbar.

Nicht in Echtzeit
: LinuxCNC kann auch auf Nicht-Echtzeit-Plattformen, wie z.B. einer normalen Installation von Debian oder Ubuntu ohne speziellen Echtzeit-Kernel, gebaut und ausgeführt werden. . In diesem Modus ist LinuxCNC nicht nützlich für die Steuerung von Werkzeugmaschinen, aber es ist nützlich für die Simulation der Ausführung von G-Code und für die Prüfung der Nicht-Echtzeit-Teile des Systems (wie die Benutzeroberflächen, und einige Arten von Komponenten und Gerätetreiber). . Um die Echtzeit-Fähigkeiten von LinuxCNC zu nutzen, müssen bestimmte Teile von LinuxCNC mit root-Rechten laufen. Um root für diese Teile zu aktivieren, führen Sie diesen zusätzlichen Befehl nach dem `make` aus, der LinuxCNC baut:

```
>$ sudo make setuid
```

6.4 Build-Modi

Es gibt zwei Möglichkeiten, LinuxCNC zu bauen: Den Entwickler-freundlichen ”run in place” Modus und den benutzerfreundlichen Debian-Packaging-Modus.

6.4.1 Kompilieren (bauen) für eine Ausführung ohne Installation (”run-in-place”)

In einem Run-In-Place-Build werden die LinuxCNC-Programme aus den Quellen kompiliert und dann direkt aus dem Build-Verzeichnis ausgeführt. Nichts wird außerhalb des Build-Verzeichnisses installiert. Dies ist schnell und einfach und geeignet für eine schnelle Iteration von Änderungen. Die LinuxCNC Testsuite läuft nur in einem Run-In-Place Build. Die meisten LinuxCNC-Entwickler bauen in erster Linie mit diesem Modus.

Das Bauen für Run-In-Place folgt den Schritten im Schneller Einstieg Abschnitt am Anfang dieses Dokuments, möglicherweise mit anderen Argumenten für `src/configure` und `make`.

6.4.1.1 `src/configure` Argumente

Das Skript `src/configure` konfiguriert, wie der Quellcode kompiliert werden soll. Es nimmt viele optionale Argumente entgegen. Um alle Argument zu `src/configure` aufzulisten, führe folgendes aus:

```
$ cd linuxcnc-source-dir/src
$ ./configure --help
```

Die am häufigsten verwendeten Argumente sind:

`--with-realtime=uspace`
: Bauen für jede Echtzeit-Plattform, oder für Nicht-Echtzeit. Die resultierenden ausführbaren Dateien von LinuxCNC laufen sowohl auf einem Linux-Kernel mit Preempt-RT-Patches (mit Echtzeit-Maschinenkontrolle) als auch auf einem reguälren (ungepatchten) Linux-Kernel (mit G-Code-Simulation, aber ohne Echtzeit-Maschinenkontrolle).

  `Werden Entwicklungsdateien für Xenomai (typischerweise aus dem Paket libxenomai-dev) oder RTAI (typischerweise aus einem Paket mit einem Namen, der mit ”rtai-modules” beginnt) installiert, wird auch die Unterstützung für diese Echtzeit-Kernels aktiviert.`

`--with-realtime=/usr/realtime-$VERSION`
: Bauen Sie für die RTAI-Echtzeitplattform unter Verwendung des älteren ”Echtzeit-Kernel” Pakets. Dies erfordert, dass Sie einen RTAI-Kernel und die RTAI-Module in `/usr/realtime-$VERSION` installiert haben. Die resultierenden ausführbare Dateien von LinuxCNC werden nur auf dem angegebenen RTAI-Kernel laufen. Ab LinuxCNC 2.7, produziert dies die beste Echtzeit-Leistung.

`--enable-build-documentation`
: Erzeuge (engl. build) die Dokumentation zusätzlich zu den ausführbaren Dateien. Diese Option erhöht den Zeitaufwand für die Kompilierung erheblich, da die Erstellung der Dokumentation recht zeitaufwendig ist. Wenn Sie nicht aktiv an der Dokumentation arbeiten, sollten Sie diese Option nicht anwählen.

`--disable-build-documentation-translation`
: Deaktivieren Sie die Erstellung der übersetzten Dokumentation für alle verfügbaren Sprachen. Die Erstellung der übersetzten Dokumentation nimmt sehr viel Zeit in Anspruch, daher ist es empfehlenswert, dies zu überspringen, wenn es nicht wirklich benötigt wird.

6.4.1.2 `make` Argumente

Der Befehl *make* akzeptiert zwei nützliche optionale Argumente.

Parallele Kompilierung
: `make` nimmt ein optionales Argument `-j` *N* (wobei *N* eine Zahl ist). Dies ermöglicht die parallele Kompilierung mit *N* gleichzeitigen Prozessen, was das Bauen erheblich beschleunigen kann.

  Ein nützlicher Wert für *N* ist die Anzahl der CPUs in Ihrem Build-System.

Sie können die Anzahl der CPUs ermitteln, indem Sie `nproc` ausführen.

Bauen nur eines bestimmten Ziels

Wenn Sie nur einen bestimmten Teil von LinuxCNC bauen wollen, können Sie das, was Sie bauen wollen, in der Kommandozeile `make` nennen. Wenn Sie zum Beispiel an einer Komponente namens "froboz" arbeiten, können Sie seine ausführbare Datei zu bauen, indem Sie:

```
$ cd linuxcnc-source-dir/src
$ make ../bin/froboz
```

6.4.2 Debian-Pakete erstellen

Beim Erstellen von Debian-Paketen werden die LinuxCNC-Programme anhand des Quellcodes kompiliert und dann in einem Debian-Paket gespeichert, komplett mit Abhängigkeitsinformationen. Dieser Prozess beinhaltet standardmäßig auch das Erstellen der Dokumentation, was aufgrund der ganzen I/O-Vorgänge für die vielen Sprachen seine Zeit dauert, aber das kann übersprungen werden. LinuxCNC wird dann als Teil dieser Pakete auf denselben Rechner oder auf jedem Rechner derselben Architektur installiert, auf welchen die .deb-Dateien kopiert wurden. LinuxCNC kann nicht ausgeführt werden, bis die Debian-Pakete auf dem gewünschen Rechner installiert sind und dann die ausführbaren Dateien in /usr/bin und /usr/lib verfügbar sind, genau wie andere reguläre Software des Systems.

Diese Art des Bauens ist vor allem dann nützlich, wenn die Software für die Auslieferung an Endbenutzer verpackt werden soll und wenn die Software für einen Rechner erstellt werden soll, auf dem die Build-Umgebung nicht installiert ist oder der keinen Internetzugang hat.

Pakete zu Bauens ist vor allem dann nützlich für eine Auslieferung der Software an Endbenutzer. Entwickler unter sich tauschen nur den Quellcode aus, der wahrscheinlich vom unten referierten LinuxCNC GitHub-Repository unterstützt wird. Auch wenn die Software für einen Rechner erstellt werden soll, auf dem die Build-Umgebung nicht installiert ist oder der keinen Internetzugang hat, werden fertig gebaute Pakete gerne akzeptiert.

Die Erstellung von Debian-Paketen wird mit dem `dpkg-buildpackage`-Tool durchgeführt, das vom `dpkg-dev`-Paket bereitgestellt wird. Dessen Ausführung geht einher mit einer Reihe von Vorarbeiten, wie nachfolgend weiter beschrieben: * die allgemeine bauinfrastruktur ist zu installieren, d.h. die compiler usw. * Paket-spezifische Abhängigkeiten für das Bauen der Pakete (engl. build dependencies) sind zu installieren, wie beschrieben im Abschnitt Den Build Abhängigkeiten genügen. * Dateien im debian Ordner müssen vollständig sein, die das Paket beschreiben

Build-Tools wurden als virtuelles Paket unter dem Namen `build-essential` zusammengefasst. Um es zu installieren, führen Sie aus:

```
$ sudo apt-get install build-essential
```

Sobald diese Voraussetzungen erfüllt sind, besteht die Erstellung der Debian-Pakete aus zwei Schritten.

Der erste Schritt ist die Generierung der Debian-Paket-Skripte und Metadaten aus dem Git-Repositorium, indem Sie dies ausführen:

```
$ cd linuxcnc-dev
$ ./debian/configure
```

Anmerkung

Das Skript `debian/configure` unterscheidet sich von dem Skript `src/configure`!
Das `debian/configure` Skript akzeptiert verschiedene Argumente, abhängig von der Plattform, auf der/für die Sie bauen, siehe den debian/configure Argumente Abschnitt. Ohne weitere Angaben ist er für die Variante von LinuxCNC im user space ("uspace") voreingestellt, einen preempt_rt Kernel erwartend um die Latenzen zu minimieren.

Sobald die Debian-Paket-Skripte und Meta-Daten konfiguriert sind, bauen Sie das Paket, indem Sie `dpkg-buildpackage` ausführen:

```
$ dpkg-buildpackage -b -uc
```

Anmerkung
`dpkg-buildpackage` muss aus der Wurzel (engl. root) des Verzeichnisses mit dem Quellcode heraus gestartet werden, dass Sie möglicherweise `linuxcnc-source-dir` genannt haben, **nicht** aus `linuxcnc-source-dir/debian` heraus.
`dpkg-buildpackage` akzeptiert ein optionales Argument ``-j``*N* (wobei *N* eine Zahl ist). Dies ermöglicht es, mehrere jobs gleichzeitig auszuführen.

6.4.2.1 LinuxCNCs Argumente für `debian/configure`

Der LinuxCNC-Quellbaum hat ein debian-Verzeichnis mit allen Informationen darüber, wie das Debian-Paket gebaut werden soll, aber einige wesentliche Dateien innerhalb werden nur als Vorlagen verteilt. Das `debian/configure`-Script bereitet die eigentlichen Dateien für die regulären Debian-Verpackungsprogra vor und muss daher auch vor `dpkg-checkbuilddeps` oder `dpkg-buildpackage` ausgeführt worden sein.

Das `debian/configure`Skript erwartet ein einziges Argument, das die Echtzeit- oder Nicht-Echtzeit-Plattform angibt, für die gebaut werden soll. Die normalen Werte für dieses Argument sind:

`no-docs`
: Erstellen der Dokumentation überspringen.

`uspace`
: Konfigurieren Sie das Debian-Paket für Preempt-RT-Echtzeit oder für Nicht-Echtzeit (diese beiden sind kompatibel).

`noauto` , `rtai` , `xenomai`
: Normalerweise wird die Liste der RTOS, die uspace Echtzeit unterstützen soll, automatisch erkannt. Wenn Sie möchten, können Sie jedoch eine oder mehrere dieser Optionen nach `uspace` angeben, um die Unterstützung für diese RTOS zu aktivieren. Um die automatische Erkennung zu deaktivieren, geben Sie `noauto` an.

  Wenn Sie nur die traditionelle RTAI ”Kernel-Modul” Echtzeit wollen, verwenden Sie stattdessen `-r` oder `$KERNEL_VERSION`.

`rtai=<package name>`
: Wenn das Entwicklungspaket für RTAI, lxrt, nicht mit ”rtai-modules” beginnt, oder wenn das erste von apt-cache search aufgelistete Paket nicht das gewünschte ist, geben Sie den Paketnamen explizit an.

`-r`
: Konfigurieren Sie das Debian-Paket für den aktuell laufenden RTAI-Kernel. Sie müssen einen RTAI-Kernel auf Ihrer Build-Maschine laufen haben, damit dies funktioniert!

`$KERNEL_VERSION`
: Konfigurieren Sie das Debian-Paket für die angegebene RTAI-Kernel-Version (zum Beispiel ”3.4.9-rtai-686-pae”). Das passende Kernel-Headers-Debian-Paket muss auf Ihrer Build-Maschine installiert sein, zum Beispiel ”linux-headers-3.4.9-rtai-686-pae”). Beachten Sie, dass Sie LinuxCNC in dieser Konfiguration *erstellen* können, aber wenn Ihr Rechner nicht auch mit dem passenden RTAI-Kernel läuft, können Sie LinuxCNC nicht *ausführen*, auch die Test-Suite nicht.

6.4.2.2 Erfüllen von Build-Abhängigkeiten

Auf Debian-basierenden Plattformen bieten wir Paketierungs-Metadaten, die wissen, welche externen Softwarepakete installiert werden müssen, um LinuxCNC zu bauen. Dies wird Build-Abhängigkeiten von LinuxCNC genannt, d.h. heißt solche Pakete, die installiert sein müssen, damit * das Bauen wurde erfolgreich beendet und * der Vorgang kann identisch reproduziert werden.

Sie können diese Meta-Daten verwenden, um die benötigten Pakete aufzulisten, die aus Ihrem Build-System fehlt. Gehen Sie zunächst zum Quellbaum von LinuxCNC und starten Sie seine Standard-Selbstkonfiguration, wenn nicht bereits ausgeführt:

```
$ cd linuxcnc-dev
$ ./debian/configure
```

Dies bereitet die Datei debian/control vor, die Listen von zu erstellenden Debian-Paketen enthält mit den Angaben von weiteren Paketen, die zur Laufzeit von LinuxCNC benötigt werden und für unseren Zweck auch eine Liste für die Pakete, die für die Kompilierung von LinuxCNC benötigt werden (engl. Build-Dependencies).

Der einfachste Weg, um alle Build-Dependencies zu installieren, ist es sie auszuführen (aus demselben Verzeichnis):

```
sudo apt-get build-dep .
```

dies wird all diejenigen erforderlichen Abhängigkeiten installieren, die noch nicht installiert sind, aber verfügbar. Der . ist Teil der Befehlszeile, d.h. eine Anweisung, die Abhängigkeiten für den gegebenen Quellbaum holen, nicht für die Abhängigkeiten eines anderen Pakets. Dies schließt die Installation von für das Bauen benötigter Pakete ab.

Der Rest dieses Abschnitts beschreibt einen halbmanuellen Ansatz. Die Liste der Abhängigkeiten in debian/control ist lang und es ist mühsam, den aktuellen Zustand der Installation mit den bereits installierten Paketen mit dieser Liste zu vergleichen. Debian-Systeme bieten ein Programm namens `dpkg-checkbuilddeps`, das die Paket-Metadaten analysiert und die als Build-Abhängigkeiten aufgelisteten Pakete mit der Liste der installierten Pakete vergleicht und Ihnen sagt, was fehlt.

Installieren Sie zuerst das Programm `dpkg-checkbuilddeps`, indem Sie es ausführen:

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

Dies erzeugt die Datei `debian/control` in einem benutzerlesbaren yaml-Format, das die Build-Abhängigkeite
nahe dem Dateianfang listet. Sie können diese Metadaten verwenden, um die benötigten Pakete, die von Ihrem Build-System fehlt, leicht aufzulisten. Sie können sich entscheiden, diese Dateien manuell zu überprüfen, wenn Sie ein gutes Verständnis haben, was bereits installiert ist.

Alternativ stellen Debian-Systeme ein Programm namens `dpkg-checkbuilddeps` zur Verfügung, das die Paket-Metadaten herausliest, die als Build-Abhängigkeiten aufgeführten Pakete gegen die Liste der installierten Pakete vergleicht und Ihnen sagt was fehlt. Auch `dpkg-buildpackage` würde Sie darüber informieren, was fehlt, und wäre ebenso nutzbar. Es berichtet jedoch fehlende Build-Deps erst nachdem von Patches aus dem Order `debian/patches` bereits automatisch eingepflegt wurden (falls vorhanden). Wenn Sie neu zu Linux und git Version Management sind, kann ein sauberer Start bevorzugt sein, um Komplikationen zu vermeiden.

Das Helfer-Werkzeug `dpkg-checkbuilddeps` (auch aus dem dpkg-dev Paket, das installiert wurde als Abhängigkeit des virtuellen Pakets `build-essential`) kann gebeten werden, seine Arbeit zu tun (beachten Sie, dass es aus dem Verzeichnis `linuxcnc-source-dir` ausgeführt werden muss, **nicht** aus `linuxcnc-source-dir/debian`):

```
$ dpkg-checkbuilddeps
```

Es wird eine Liste von Paketen ausgegeben, die erforderlich sind, um LinuxCNC auf Ihrem System zu bauen, die aber noch nicht installiert sind. Sie können nun fehlende build-dependencies installieren

händisch
: Installieren Sie sie alle mit `sudo apt-get install`, gefolgt von den Paketnamen. Sie können `dpkg-checkbuilddeps` jederzeit wieder ausführen, um fehlende Pakete aufzulisten, was keinen Einfluss auf den Quellbaum hat.

automatisiert
: Ausführen von `sudo apt build-dep .`.

Wenn Sie Zweifel daran haben, was ein bestimmtes Paket einer Build-Dep zur Verfügung stellt, überprüfen Sie die Beschreibung des Pakets mit `apt-cache show` *Paketname*.

6.4.2.3 Optionen für `dpkg-buildpackage`

Um ein typisches Debian-Paket zu erstellen, würden Sie dpkg-buildpackage ohne Argumente ausführen. Wie oben vorgestellt, bietet der Befehl zwei zusätzliche Optionen an. Wie für alle guten Linux-Tools bietet die ”man page” alle Details mit `man dpkg-buildpackage`.

`-uc`
: Anweisung, die resultierenden Pakete nicht digital zu signieren. Sie möchten Ihre Pakete nur mit einem GPG-Schlüssel von Ihnen unterschreiben, wenn Sie sie an andere verteilen wollten. Ist diese Option nicht gesetzt und scheitert dann der Versuch, das Paket zu unterzeichnen, so würde dies die .deb-Datei jedoch nicht beeinflussen.

`-b`
: Kompiliert nur die architekturabhängigen Pakete (wie die `linuxcnc` Binaries und GUIs). Dies ist sehr hilfreich, um den Aufwand für hardware-unabhängige Teile zu vermeiden. Für LinuxCNC ist dies die Dokumentation und diese ist eh bereits online verfügbar.

Wenn Sie in Schwierigkeiten bei der Paket-Erstellung kommen sollten, so kann das LinuxCNC Forum möglicherweise online weiterhelfen.

Noch in Entwicklung ist die Unterstützung der DEB_BUILD_OPTIONS-Umgebungsvariable. Setzen Sie diese auf

`nodocs`
: um das Bauen der Dokumentation zu überspringen, verwenden Sie stattdessen vorzugsweise das `-B`-Flag von dpkg-buildpackage.

`nocheck`
: to skip self-tests of the LinuxCNC build process. This saves some time and reduces the demand for a few software packages that may not be available for your system, i.e. the xvfb in particular. You should not set this option to gain some extra confidence in your build to perform as expected unless you are running into mere technical difficulties with the test-specific software dependencies.

Eine Umgebungsvariable kann zusammen mit der Ausführung des Befehls auf der Kommandozeile gesetzt werden, z.B.

```
DEB_BUILD_OPTIONS=nocheck dpkg-buildpackage -uc -B
```

würde alle in diesem Abschnitt verwendeten Optionen kombinieren.

6.4.2.4 Installieren selbst-gebauter Debian-Pakete

Ein Debian-Paket kann durch dessen .deb-Erweiterung im Dateinamen erkannt werden. Das Tool, das es installiert heißt `dpkg` und ist Teil jeder Debian-Installation. Die .deb-Dateien, die von `dpkg-buildpackage` erstellt wurden, finden sich im Verzeichnis über dem linuxcnc-source-dir, d.h. in `...` Um zu sehen, welche Dateien in einem Paket bereitgestellt werden, führen Sie aus

```
dpkg -c ../linuxcnc-uspace*.deb
```

Die Version von LinuxCNC wird Teil des Dateinamens sein an der Stelle des Sternchens. Möglicherweise werde zu viele Dateien aufgelistet, um auf Ihrem Bildschirm passen. Wenn Sie nicht in Ihrem Terminal scrollen können, dann fügen Sie `| more` zu diesem Befehl hinzu, um die Ausgabe durch einen sogenannten ”pager” zu lenken. Der ist zu verlassen mit ”q”.

Um das Paket zu installieren, führen Sie aus

```
sudo dpkg -i ../linuxcnc*.deb
```

6.5 Einrichten der Umgebung

Dieser Abschnitt beschreibt die speziellen Schritte, die benötigt werden, um eine Maschine einzurichten, um die LinuxCNC-Programme auszuführen, einschließlich der Tests.

6.5.1 Erhöhen Sie das Limit für den gesperrten Speicher

LinuxCNC versucht, seine Echtzeit-Latenz zu verbessern, indem es den verwendeten Speicher im RAM sperrt. Es tut dies, um zu verhindern, dass das Betriebssystem von den RAM LinuxCNC auf die Festplatte auslagert (slang: swap), was sich sehr stark negativ auf die Latenz auswirken würde. Normalerweise wird das Sperren des Speichers in RAM ungern gesehen, und das Betriebssystem legt eine strenge Grenze fest, wie viel Speicher ein Benutzer gesperrt haben darf.

Bei Verwendung der Preempt-RT-Echtzeitplattform läuft LinuxCNC mit genügend Privilegien, um seine Speichersperrgrenze selbst zu erhöhen. Bei Verwendung der RTAI-Echtzeit-Plattform hat es nicht genug Privilegien, und der Anwender muss die Speichersperre Grenze zu erhöhen.

Wenn LinuxCNC beim Start folgende Meldung zeigt, ist das Problem die für Ihre System konfigurierte Grenze des gesperrten Speichers:

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

Um dieses Problem zu beheben, fügen Sie eine Datei namens `/etc/security/limits.d/linuxcnc.conf` (als root) mit Ihrem bevorzugten Texteditor hinzu (z.B. `sudo gedit /etc/security/limits.d/linuxcnc.co`
Die Datei sollte die folgende Zeile enthalten:

```
* - memlock 20480
```

Melden Sie sich ab und wieder an, damit die Änderungen wirksam werden. Überprüfen Sie mit dem folgenden Befehl, ob die Speichersperrgrenze angehoben wurde:

```
$ ulimit -l
```

6.6 Kompilieren (bauen) für Gentoo

Bauen auf Gentoo ist möglich, aber nicht unterstützt. Achten Sie darauf, dass Sie ein Desktop-Profil ausführen. Dieses Projekt verwendet das Tk Widget Set, Asciidoc und hat einige andere Abhängigkeiten. Sie sollten als root installiert werden:

```
~ # euse -E tk imagequant
~ # emerge -uDNa world
~ # emerge -a dev-libs/libmodbus dev-lang/tk dev-tcltk/bwidget dev-tcltk/tclx
~ # emerge -a dev-python/pygobject dev-python/pyopengl dev-python/numpy
~ # emerge -a app-text/asciidoc app-shells/bash-completion
```

Sie können wieder zurück zu einem normalen Anwenderfür den überwiegenden restlichen Teil der Installation wechseln. Als dieser Anwender erstellen Sie eine virtuelle Umgebung für pip, dann installieren Sie die pip Pakete wie folgt:

```
~/src $ python -m venv   --system-site-packages ~/src/venv
~/src $ . ~/src/venv/bin/activate
(venv) ~/src $ pip install yapps2
(venv) ~/src $
```

Dann können Sie regulär weitermachen:

```
(venv) ~/src $ git clone https://github.com/LinuxCNC/linuxcnc.git
(venv) ~/src $ cd linuxcnc
(venv) ~/src $ cd src
(venv) ~/src $ ./autogen.sh
(venv) ~/src $ ./configure --enable-non-distributable=yes
(venv) ~/src $ make
```

Es gibt keine Notwendigkeit zur Ausführung von ”make suid”, nur stellen Sie sicher, dass Ihr UNIX-Account in der ”dialout” Gruppe ist. Um linuxcnc zu starten, müssen Sie in der Python Virtual Environment sein und die linuxcnc Umgebung einrichten:

```
~ $ . ~/src/venv/bin/activate
(venv) ~ $ . ~/src/linuxcnc/scripts/rip-environment
(venv) ~ $ ~/src/linuxcnc $ scripts/linuxcnc
```

6.7 Optionen zum Auschecken des Git-Repos

Die Schnellstart Anleitung am Anfang dieses Dokuments klont unser Git-Repository auf https://github.com/-LinuxCNC/linuxcnc.git. Dies ist der schnellste und einfachste Weg um loszulegen. Es gibt jedoch auch noch andere Optionen, die Sie in Betracht ziehen sollten.

6.7.1 Forken Sie uns auf GitHub

The LinuxCNC project git repo is at https://github.com/LinuxCNC/linuxcnc. GitHub is a popular git hosting service and code sharing website. You can easily (and at no costs) create a fork (a second instance holding a copy that you control) of the LinuxCNC git repository at GitHub. You can then use that fork of yours to track and publish your changes, receive comments to your changes and accept patches from the community. .

After creating your own GitHub fork of LinuxCNC, clone it to your development machine and proceed with your hacking as usual.

Wir vom LinuxCNC-Projekt hoffen, dass Sie Ihre Änderungen mit uns teilen werden, damit die Gemeinschaft von Ihrer Arbeit profitieren kann. GitHub macht dieses Teilen sehr einfach: Nachdem Sie Ihre Änderungen aufpoliert und in Ihren GitHub-Fork gepusht haben, schicken Sie uns einen Pull Request.

Kapitel 7

Hinzufügen von Konfigurationsauswahlelementen

Beispielkonfigurationen können der Konfigurationsauswahl auf zwei Arten hinzugefügt werden:

- Hilfsanwendungen — Anwendungen, die unabhängig mit einem deb-Paket installiert werden, können Konfigurationsunterverzeichnisse in einem bestimmten Systemverzeichnis ablegen. Der Verzeichnisname wird mit dem Shell-Skript linuxcnc_var angegeben:

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

- Laufzeiteinstellungen — der Konfigurationsselektor kann auch Konfigurationsunterverzeichnisse anbieten, die zur Laufzeit mit einer exportierten Umgebungsvariablen (LINUXCNC_AUX_CONFIGS) angegeben werden. Diese Variable sollte eine Pfadliste von einem oder mehreren Konfigurationsverzeichnissen sein, die durch ein (:) getrennt sind. Typischerweise wird diese Variable in einer Shell gesetzt, die `linuxcnc` startet, oder in einem Startskript des Benutzers `~/.profile`. Beispiel:

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

Kapitel 8

Mitwirkung an LinuxCNC

8.1 Einführung

Dieses Dokument enthält Informationen für Entwickler über die LinuxCNC-Infrastruktur und beschreibt die besten Praktiken für das Einbringen von Code- und Dokumentations-Updates in das LinuxCNC-Projekt.

In diesem Dokument bedeutet ”Quelle” sowohl den Quellcode der Programme und Bibliotheken als auch den Quelltext der Dokumentation.

8.2 Kommunikation unter LinuxCNC-Entwicklern

Die Projektentwickler kommunizieren hauptsächlich auf zwei Arten miteinander:

- Über IRC, unter #linuxcnc-devel auf Libera.chat.
- Per E-Mail, auf der developers’ Mailingliste

8.3 Das LinuxCNC Source Forge-Projekt

Wir verwenden Source Forge für Mailinglisten.

8.4 Das Git Revisionskontrollsystem

Der gesamte Quellcode von LinuxCNC wird im Git Revisionskontrollsystem verwaltet.

8.4.1 LinuxCNC offizielles Git Repository

Das offizielle LinuxCNC git repo ist zugänglich über https://github.com/linuxcnc/linuxcnc/

Jeder kann eine schreibgeschützte Kopie des LinuxCNC-Quellbaums über git erhalten:

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

Wenn Sie ein Entwickler mit Schreibrechten (genannt ”push” in Anlehnung an das git-Kommando) sind, folgen Sie den Anweisungen in github, um ein Repository einzurichten, von dem aus Sie pushen können.

Beachten Sie, dass der Clone-Befehl das lokale LinuxCNC Repo in ein Verzeichnis namens `linuxcnc-dev` legt, anstatt des Standardverzeichnisses `linuxcnc`. Dies wurde so gewählt, da die LinuxCNC-Software standardmäßig die Configurations-Dateien (.ini) und G-Code-Programme in einem Verzeichnis in dem Pfad `$HOME/linuxcnc` erwartet. Dies soll Verwirrungen zwischen der Entwickung im Git Repository und der Ausführung helfen zu vermeiden.

Problem-Meldungen (engl. *issues*) und Anforderungen zur Integration eigener Änderungen (genannt *pull requests*, kurz: PR) sind auf GitHub willkommen: https://github.com/LinuxCNC/linuxcnc/issues https://github.com/LinuxCNC/linuxcnc/pulls

8.4.2 Verwendung von Git im LinuxCNC-Projekt

Wir verwenden die hier beschriebenen Git-Workflows ”merging upwards” und ”topic branches”:

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

Wir haben einen Entwicklungszweig namens `master` und einen oder mehrere stabile Entwicklungszweige (engl. branches) mit Namen wie `2.6` und `2.7`, welche die Versionsnummer der daraus jeweils hervorgehenden Veröffentlichung angeben.

Fehlerbehebungen gehen in den ältesten anwendbaren stabilen branch, und dieser branch wird in den nächst neueren stabilen branch zusammengeführt, und so weiter bis zu `master`. Der Committer des Bugfixes kann die Zusammenführung selbst durchführen, oder sie jemand anderem überlassen.

Neue Funktionen gehen im Allgemeinen in den ”Master”-branch, aber einige Arten von Funktionen (insbesondere gut isolierte Gerätetreiber und Dokumentationen) können (nach dem Ermessen der Verantwortlichen für den stabilen branch) in den stabilen branch wandern und dort zusammengeführt werden, genau wie Bugfixes.

8.4.3 Git-Tutorials

Es gibt viele ausgezeichnete, kostenlose Git-Tutorials im Internet.

Die erste Anlaufstelle ist wahrscheinlich die Manpage ”gittutorial”. Auf diese Manpage kann man zugreifen, bspw. durch die Ausführung von ”man gittutorial” in einem Terminal (wenn man die git-Manpages installiert hat, bei Debian im Paket ”git-man”). Das gittutorial und die dazugehörige Dokumentation sind auch hier online verfügbar:

- git tutorial: https://www.kernel.org/pub/software/scm/git/docs/gittutorial.html
- git tutorial 2: https://www.kernel.org/pub/software/scm/git/docs/gittutorial-2.html
- Alltägliches Git mit 20 Befehlen oder so: https://www.kernel.org/pub/software/scm/git/docs/giteveryday.ht
- Git-Benutzerhandbuch: https://www.kernel.org/pub/software/scm/git/docs/user-manual.html

Eine ausführlichere Dokumentation von Git finden Sie in dem Buch ”Pro Git”: https://git-scm.com/book

Ein weiteres Online-Tutorial, das empfohlen wurde, ist ”Git for the Lazy”: https://wiki.spheredev.org/-index.php/Git_for_the_lazy

8.5 Überblick über den Prozess

Ein Überblick darüber, wie man Änderungen am Quelltext vornimmt, sieht folgendermaßen aus:

- Kommunizieren Sie mit den Projektentwicklern und lassen Sie uns wissen, woran Sie hacken. Erklären Sie, was Sie tun und warum.
- Klonen des Git-Repositories.
- Nehmen Sie Ihre Änderungen in einer lokalen Entwicklungszweig (engl. branch, wie orchestriert durch das Entwicklungswerkzeug für code management ”git”) vor.
- Das Hinzufügen von Dokumentation und das Schreiben von Tests ist ein wichtiger Teil des Hinzufügens einer neuen Funktion. Andernfalls werden andere nicht wissen, wie Ihre Funktion zu verwenden ist, und wenn andere Änderungen Ihre Funktion zerstören, kann dies ohne einen Test unbemerkt bleiben.
- Kommunizieren Sie Ihre Änderungen mit den anderen Projektentwicklern auf eine der folgenden Arten:
  - Pushen Sie Ihren Entwicklungszweig (engl. branch) auf Github und erstellen Sie einen Github-Pull-Request auf https://github.com/linuxcnc/linuxcnc (dies erfordert einen Github-Account), oder
  - Verschieben Sie Ihren Entwicklungszweig in ein öffentlich sichtbares Git-Repository (z. B. Github, Ihren eigenen öffentlich zugänglichen Server usw.) oder teilen Sie diesen Ort der emc-developers Mailingliste mit, oder
  - Senden Sie Ihre Übertragungen an die LinuxCNC-Entwickler-Mailingliste (<emc-developers@lists.sour
    (verwenden Sie `git format-patch`, um die Patches zu erstellen).
- Setzen Sie sich für Ihren Änderungswunsch (engl. Patch) ein:
  - Erläutern Sie, welches Problem gelöst wird und warum dies in LinuxCNC enthalten sein sollte.
  - Seien Sie offen für Fragen und Rückmeldungen aus der Entwicklergemeinschaft.
  - Es ist nicht ungewöhnlich, dass ein Patch mehrere Überarbeitungen durchläuft, bevor er angenommen wird.

8.6 Git-Konfiguration

Um für die Aufnahme in den LinuxCNC-Quellcode in Betracht gezogen zu werden, müssen Commits korrekte Author-Felder haben, die den Autor des Commits identifizieren. Eine gute Möglichkeit, dies sicherzustellen, ist die Einstellung der globalen Git-Konfiguration:

```
git config --global user.name ”Your full name”
git config --global user.email ”you@example.com”
```

Benutzen Sie Ihren richtigen Namen und eine direkt nutzbare E-Mail Adresse.

8.7 Effektive Nutzung von Git

8.7.1 Commit-Inhalte

Halten Sie Ihre Commits klein und auf den Punkt. Jeder Commit sollte eine logische Änderung am Projektarchiv bewirken.

8.7.2 Schreiben Sie gute Commit-Nachrichten

Halten Sie die Commit-Meldungen etwa 72 Spalten breit (damit sie in einem Terminalfenster mit Standardgröße nicht umbrechen, wenn sie von `git log` angezeigt werden).

Verwenden Sie die erste Zeile als Zusammenfassung der Absicht der Änderung (fast wie die Betreffzeile einer E-Mail). Danach folgt eine Leerzeile und dann eine längere Nachricht, in der die Änderung erläutert wird. Beispiel:

8.7.3 Ein git commit muss im richtigen Entwicklungszweig ausgeführt werden

Fehlerbehebungen sollten im ältesten anwendbaren branch durchgeführt werden. Neue Funktionalität sollte in den Master-Zweig kommen. Wenn Sie sich nicht sicher sind, wo eine Änderung hingehört, fragen Sie im IRC oder auf der Mailingliste.

8.7.4 Verwenden Sie mehrere Commits, um Änderungen zu organisieren

Organisieren Sie Ihre Änderungen gegebenenfalls in einem Zweig (engl. branch, eine zusammengehörige Folge von Commits), wobei jeder Commit ein logischer Schritt auf dem Weg zum endgültigen Ziel ist. Beispielsweise sollten Sie zunächst einen komplexen Code in eine neue Funktion umwandeln. Beheben Sie dann in einem zweiten Commit einen zugrunde liegenden Fehler. Dann, im dritten Commit, fügen Sie eine neue Funktion hinzu, die durch das Refactoring einfacher geworden ist und die ohne die Behebung des Fehlers nicht funktioniert hätte.

Das ist hilfreich für die Prüfer, denn es ist einfacher zu sehen, dass der Schritt ”Code in eine neue Funktion umwandeln” richtig war, wenn keine anderen Änderungen darin enthalten sind; es ist einfacher zu sehen, dass der Fehler behoben ist, wenn die Änderung, die ihn behebt, getrennt von der neuen Funktion ist; und so weiter.

8.7.5 Folgen Sie den Stil des umgebenden Codes

Bemühen Sie sich, dem vorherrschenden Einrückungsstil des umgebenden Codes zu folgen. Insbesondere Änderungen an Leerzeichen erschweren es anderen Entwicklern, Änderungen im Laufe der Zeit zu verfolgen. Wenn eine Neuformatierung des Codes erforderlich ist, sollten Sie diese getrennt von semantischen Änderungen vornehmen.

8.7.6 Werden Sie RTAPI_SUCCESS los, verwenden Sie stattdessen 0

Der Test ”retval < 0” sollte Ihnen bekannt vorkommen; es ist die gleiche Art von Test, den Sie im Userspace (gibt -1 für Fehler zurück) und im Kernelspace (gibt -ERRNO für Fehler zurück).

8.7.7 Vereinfachen Sie den komplizierten Verlauf, bevor Sie ihn mit anderen Entwicklern teilen

Mit Git ist es möglich, jede Bearbeitung und jeden Fehlstart als separaten Commit aufzuzeichnen. Dies ist sehr praktisch, um während der Entwicklung Kontrollpunkte zu setzen, aber oft möchte man diese Fehlstarts nicht mit anderen teilen.

Git bietet zwei Möglichkeiten, die Historie zu bereinigen, die beide frei durchgeführt werden können, bevor Sie die Änderung freigeben:

Mit `git commit --amend` können Sie zusätzliche Änderungen an dem letzten Commit vornehmen und optional auch die Commit-Nachricht ändern. Benutzen Sie dies, wenn Sie sofort merken, dass Sie etwas in der Commit-Nachricht vergessen haben, oder wenn Sie sich in der Commit-Nachricht vertippt haben.

Mit `git rebase --interactive` upstream-branch können Sie jeden Commit, den Sie seit dem Abzweig (engl. fork) Ihres Entwicklungszweigs vom Upstream-Branch (der Zweig im repository von dem Sie abgezweigt sind) gemacht haben, zurückgehen und dabei möglicherweise Commits bearbeiten, Commits verwerfen oder mit anderen Commits zusammenlegen (Squash). Rebase kann auch verwendet werden, um einzelne Commits in mehrere neue Commits aufzuteilen.

8.7.8 Stellen Sie sicher, dass jeder commit auch kompiliert werden kann

Wenn Ihre Änderung aus mehreren Patches besteht, kann `git rebase -i` verwendet werden, um diese Patches in eine Folge von Commits umzuordnen, die Schritte Ihrer Arbeit klarer darstellt. Eine mögliche Folge des Umordnens solcher Patches ist, dass man Abhängigkeiten falsch einordnen kann - zum Beispiel, wenn man die Verwendung einer Variable einführt und die Deklaration dieser Variable erst in einem späteren Patch folgt.

Beim Bauen des HEAD branches würde in einem solchen Fall nicht jeder Commit gebaut. Das machte dann `git bisect` kaputt - etwas, das jemand anderes später benutzen könnte, um den Commit zu finden, der einen Fehler eingeführte. Es ist also nicht nur wichtig, sicherzustellen, dass Ihr Zweig gebaut wird, sondern auch, dass jeder einzelne Commit erfolgreich kompiliert.

Es gibt einen automatischen Weg, einen Zweig darauf zu prüfen, ob jeder Commit baubar ist - siehe https://dustin.sallings.org/2010/03/28/git-test-sequence.html und den Code unter https://github.com/-dustin/bindir/blob/master/git-test-sequence. Verwenden Sie wie folgt (in diesem Fall testen Sie jeden Commit von origin/master bis HEAD, einschließlich der Durchführung von Regressionstests):

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

Dies meldet entweder *All is well* (engl. für *Alles in Ordnung*) oder *Broke on <commit>* (engl. für *Fehler bei <commit>*)

8.7.9 Umbenennen von Dateien

Bitte verwenden Sie die Möglichkeit Dateien umzubenennen mit Bedacht. Wie bei der Einrückung einzelner Dateien erschweren auch Umbenennungen die Verfolgung von Änderungen im Laufe der Zeit. Zumindest sollten Sie im IRC oder auf der Mailingliste einen Konsens darüber finden, dass die Umbenennung eine Verbesserung darstellt.

8.7.10 ”Rebase” bevorzugen

Verwenden Sie `git pull --rebase` anstelle von `git pull`, um eine schöne lineare Historie zu erhalten. Wenn Sie rebasen, behalten Sie Ihre Arbeit immer als Revisionen, die vor origin/master liegen, so dass Sie Dinge wie `git format-patch` ausführen können, um Entwicklungen mit anderen zu teilen, ohne sie in das zentrale Repository zu pushen.

8.8 Übersetzungen

Das LinuxCNC-Projekt verwendet `gettext`, um die Software in viele Sprachen zu übersetzen. Wir begrüßen Beiträge und Hilfe in diesem Bereich! Das Verbessern und Erweitern der Übersetzungen ist

einfach: Sie müssen keine Programmierkenntnisse haben und Sie müssen keine speziellen Übersetzungsprogramme oder andere Software installieren.

Der einfachste Weg, bei Übersetzungen zu helfen, ist die Nutzung von Weblate, einem Open-Source-Webdienst. Unser Übersetzungsprojekt finden Sie hier:

https://hosted.weblate.org/projects/linuxcnc/

Die Dokumentation zur Verwendung von Weblate finden Sie hier: https://docs.weblate.org/en/latest/-user/basic.html

8.9 Andere Möglichkeiten, einen Beitrag zu leisten

Es gibt viele Möglichkeiten, zu LinuxCNC beizutragen, die in diesem Dokument nicht behandelt werden. Diese Wege umfassen:

- Beantwortung von Fragen im Forum, auf Mailinglisten und im IRC
- Melden von Fehlern im Bug-Tracker, im Forum, auf Mailinglisten oder im IRC
- Hilfe beim Testen experimenteller Funktionen

Kapitel 9

Glossar

Eine Auflistung von Begriffen und deren Bedeutung. Einige Begriffe haben eine allgemeine Bedeutung und mehrere zusätzliche Bedeutungen für Benutzer, Installateure und Entwickler.

Acme-Schraube
: 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.

Achse
: 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.

GMOCCAPY (GUI)
: A Graphical User Interfaces available to users of LinuxCNC. It features the use and feel of an industrial control and can be used with touch screen, mouse and keyboard. It support embedded tabs and hal driven user messages, it offers a lot of hal beens to be controlled with hardware. GMOCCAPY is highly customizable.

Umkehrspiel
: 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.

Umkehrspiel-Kompensation
: 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.

Kugelumlaufspindel
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.

Kugelmutter
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.

Konfiguration(n)
Ein Verzeichnis, das eine Reihe von Konfigurationsdateien enthält. Benutzerdefinierte Konfigurationen sind in der Regel in den Benutzer home/linuxcnc/configs Verzeichnis gespeichert. Diese Dateien enthalten LinuxCNC's traditionelle INI-Datei und HAL-Dateien. Eine Konfiguration kann auch mehrere allgemeine Dateien enthalten, die Werkzeuge, Parameter und NML-Verbindungen beschreiben.

Konfiguration(v)
Die Aufgabe, LinuxCNC so einzustellen, dass es mit der Hardware einer Werkzeugmaschine übereinstimmt.

Koordinatenmessmaschine
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 handmade prototype needs to be digitized for moldmaking, or to check the accuracy of machined or molded parts.

Anzeigeeinheiten
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.

EMC
The Enhanced Machine Controller. Initially a NIST project. Renamed to LinuxCNC in 2012.

EMCIO
The module within LinuxCNC that handles general purpose I/O, unrelated to the actual motion of the axes.

EMCMOT
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.

Vorschub
Relatively slow, controlled motion of the tool used when making a cut.

Vorschubgeschwindigkeit
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.

Rückmeldung
A method (e.g., quadrature encoder signals) by which LinuxCNC receives information about the position of motors.

Vorschubgeschwindigkeit-Anpassung (engl. 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”.

Gleitkommazahl
Eine Zahl, die einen Dezimalpunkt hat, bsw. 12.3. In HAL wird sie (engl.) als Float bezeichnet.

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.

Allgemeines
Eine Art von Schnittstelle zur Kommunikation zwischen einem Computer und einem Menschen (in den meisten Fällen) über die Manipulation von Symbolen und anderen Elementen (Widgets) auf einem Computerbildschirm.

LinuxCNC
Eine Anwendung, die dem Maschinenbediener einen grafischen Bildschirm präsentiert zur Bedienung der Maschine und des Steuerungsprogramms.

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.

Pos1
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-Datei
A text file that contains most of the information that configures LinuxCNC for a particular machine.

Instanz
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.

Gelenk-Koordinaten
These specify the angles between the individual joints of the machine. See also Kinematics

Jog (manuelle Bewegung)
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
Code, der innerhalb des Kernels läuft, im Gegensatz zu Code, der im Userspace läuft. Einige Echtzeitsysteme (wie RTAI) führen Echtzeitcode im Kernel und Nicht-Echtzeitcode im Userspace aus, während andere Echtzeitsysteme (wie Preempt-RT) sowohl Echtzeit- als auch Nicht-Echtzeitcode im Userspace ausführen.

Kinematik
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.

Leitspindel
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.

Maschineneinheiten
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.

NML
Neutral Message Language provides a mechanism for handling multiple types of messages in the same buffer as well as simplifying the interface for encoding and decoding buffers in neutral format and the configuration mechanism.

Versätze
An arbitrary amount, added to the value of something to make it equal to some desired value. For example, G-code 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 G-code 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.

Werkstück Programm
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.

Programm-Einheiten
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
Allzweck-, sehr High-Level-Programmiersprache. Wird in LinuxCNC verwendet für die Axis GUI, das StepConf Konfigurationswerkzeug, und mehrere G-Code-Programmierung Skripte.

Schnell
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!

Schnellauf-Geschwindigkeit
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.

Echtzeit
Software that is intended to meet very strict timing deadlines. On Linux, in order to meet these requirements it is necessary to install a realtime kernel such as RTAI or Preempt-RT, and build the LinuxCNC software to run in the special real-time environment. Realtime software can run in the kernel or in userspace, depending on the facilities offered by the system.

RTAI
Real Time Application Interface, see https://www.rtai.org/, the real-time extensions for Linux that LinuxCNC can use to achieve real-time performance.

RTLINUX
See https://en.wikipedia.org/wiki/RTLinux, an older real-time extension for Linux that LinuxCNC used to use to achieve real-time performance. Obsolete, replaced by RTAI.

RTAPI
A portable interface to real-time operating systems including RTAI and POSIX pthreads with realtime extensions.

RS-274/NGC
The formal name for the language used by LinuxCNC part programs.

Servomotor
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.

Ganze Zahl mit Vorzeichen
A whole number that can have a positive or negative sign. In HAL it is usually a s32, but could be also a s64.

Spindel
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.

Spindeldrehzahl-Anpassung
Eine manuelle, vom Bediener gesteuerte Änderung der Geschwindigkeit, mit der sich das Werkzeug während des Schneidens dreht. Oft verwendet, um dem Bediener zu ermöglichen, für Ratter verursacht durch die cutter's Zähne anzupassen. Spindeldrehzahl Override setzt voraus, dass die LinuxCNC-Software dafür konfiguriert wurde, die Spindeldrehzahl zu steuern.

StepConf
Ein LinuxCNC Konfigurations-Assistent. Es ist in der Lage, viele Schritt-und-Richtung Bewegung Befehl basierte Maschinen zu behandeln. Er schreibt eine vollständige Konfiguration, nachdem der Benutzer ein paar Fragen über den Computer und die LinuxCNC-ausführenden Maschine beantwortet hat.

Schrittmotor
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 (engl. für Aufgabe, auch Name des entsprechenden LinuxCNC Moduls)
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 Bewegung
A move in a straight line from the start point to the end point.

Einheiten
See ”Machine Units”, ”Display Units”, or ”Program Units”.

Ganzzahl ohne Vorzeichen
A whole number that has no sign. In HAL it is usually a u32 but could be also a u64.

Weltkoordinaten
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.

Kapitel 10

Juristischer Abschnitt

Die Übersetzungen dieser Datei im Quellbaum sind nicht rechtsverbindlich.

10.1 Copyright-Bedingungen

10.2 GNU Free Documentation License

GNU Free Documentation License Version 1.1, March 2000

0. PREAMBLE

Der Zweck dieser Lizenz ist es, ein Handbuch, ein Lehrbuch oder ein anderes schriftliches Dokument ”frei” im Sinne von Freiheit zu machen: jedem die effektive Freiheit zu sichern, es zu kopieren und weiterzugeben, mit oder ohne Modifikation, entweder kommerziell oder nicht-kommerziell. In zweiter Linie bewahrt diese Lizenz dem Autor und dem Herausgeber eine Möglichkeit, Anerkennung für ihre Arbeit zu erhalten, während sie nicht für die von anderen vorgenommenen Änderungen verantwortlich gemacht werden.

Diese Lizenz ist eine Art ”Copyleft”, was bedeutet, dass abgeleitete Werke des Dokuments selbst im gleichen Sinne frei sein müssen. Sie ergänzt die GNU General Public License, die eine Copyleft-Lizenz für freie Software ist.

Wir haben diese Lizenz entworfen, um sie für Handbücher für freie Software zu verwenden, weil freie Software freie Dokumentation braucht: ein freies Programm sollte mit Handbüchern geliefert werden, welche die gleichen Freiheiten bieten wie die Software. Aber diese Lizenz ist nicht auf Software-Handbücher beschränkt; sie kann für jedes textliche Werk verwendet werden, unabhängig vom Thema oder ob es als gedrucktes Buch veröffentlicht wird. Wir empfehlen diese Lizenz in erster Linie für Werke, deren Zweck die Anleitung oder das Nachschlagen ist.

1. APPLICABILITY AND DEFINITIONS

Diese Lizenz gilt für jedes Handbuch oder andere Werk, das einen Hinweis des Urheberrechtsinhabers enthält, der besagt, dass es unter den Bedingungen dieser Lizenz verbreitet werden darf. Das ”Dokument”, unten, bezieht sich auf ein solches Handbuch oder Werk. Jedes Mitglied der Öffentlichkeit ist ein Lizenznehmer und wird als ”Sie” angesprochen.

Eine ”modifizierte Version” des Dokuments ist jedes Werk, welches das Dokument oder einen Teil davon enthält, entweder wortwörtlich kopiert oder mit Änderungen und/oder in eine andere Sprache übersetzt.

Ein ”sekundärer Abschnitt” ist ein benannter Anhang oder ein vorderer Abschnitt des Dokuments, der sich ausschließlich mit der Beziehung der Herausgeber oder Autoren des Dokuments zum Gesamtthema des Dokuments (oder zu verwandten Themen) befasst und nichts enthält, was direkt in dieses Gesamtthema fallen könnte. (Wenn das Dokument zum Beispiel teilweise ein Lehrbuch der Mathematik ist, darf ein sekundärer Abschnitt keine Mathematik erklären). Die Beziehung könnte eine Frage des historischen Zusammenhangs mit dem Thema oder mit verwandten Themen oder der rechtlichen, kommerziellen, philosophischen, ethischen oder politischen Position dazu sein.

Die unveränderlichen Abschnitte” sind bestimmte sekundäre Abschnitte, deren Titel in der Mitteilung, die besagt, dass das Dokument unter dieser Lizenz freigegeben ist, als die der unveränderlichen Abschnitte bezeichnet werden.

Die ”Covertexte” sind bestimmte kurze Textpassagen, die als Front-Cover-Texte oder Back-Cover-Texte in dem Hinweis aufgeführt sind, der besagt, dass das Dokument unter dieser Lizenz freigegeben ist.

Eine ”transparente” Kopie des Dokuments ist eine maschinenlesbare Kopie, die in einem Format dargestellt wird, dessen Spezifikation der Allgemeinheit zur Verfügung steht, dessen Inhalt direkt und unkompliziert mit allgemeinen Texteditoren oder (für Bilder, die aus Pixeln bestehen) mit allgemeinen Malprogrammen oder (für Zeichnungen) mit einem weit verbreiteten Zeichnungseditor betrachtet und bearbeitet werden kann, und die für die Eingabe in Textformatierer oder für die automatische Übersetzung in eine Vielzahl von Formaten geeignet ist, die für die Eingabe in Textformatierer geeignet sind. Eine Kopie, die in einem ansonsten transparenten Dateiformat erstellt wurde, dessen Markup so gestaltet wurde, dass eine nachträgliche Änderung durch Leser vereitelt oder erschwert wird, ist nicht transparent. Eine Kopie, die nicht ”Transparent” ist, wird als ”Opak” bezeichnet.

Geeignete Formate für transparente Kopien sind z. B. ASCII ohne Markup, Texinfo-Eingabeformat, LaTeX-Eingabeformat, SGML oder XML mit einer öffentlich zugänglichen DTD und standardkonformes einfaches HTML, das für die Bearbeitung durch den Menschen ausgelegt ist. Zu den undurchsichtigen Formaten gehören PostScript, PDF, proprietäre Formate, die nur von proprietären Textverarbeitungsprogrammen gelesen und bearbeitet werden können, SGML oder XML, für die eine DTD und/oder die Verarbeitungswerkzeuge nicht allgemein verfügbar sind, und das maschinell erzeugte HTML, das von einigen Textverarbeitungsprogrammen nur zu Ausgabezwecken erzeugt wird.

Die ”Titelseite” bedeutet bei einem gedruckten Buch die Titelseite selbst sowie die Folgeseiten, die benötigt werden, um das Material, das nach dieser Lizenz auf der Titelseite erscheinen soll, lesbar zu halten. Für Werke in Formaten, die kein Titelblatt als solches haben, bedeutet ”Titelblatt” den Text in der Nähe des auffälligsten Erscheinens des Werktitels, der dem Beginn des Textes vorausgeht.

2. VERBATIM COPYING

Sie dürfen das Dokument in jedem beliebigen Medium kopieren und verbreiten, sei es kommerziell oder nicht kommerziell, vorausgesetzt, dass diese Lizenz, die Urheberrechtsvermerke und der Lizenzvermerk, der besagt, dass diese Lizenz für das Dokument gilt, in allen Kopien wiedergegeben werden, und dass Sie keine weiteren Bedingungen zu denen dieser Lizenz hinzufügen. Sie dürfen keine technischen Maßnahmen anwenden, um das Lesen oder weitere Kopieren der von Ihnen erstellten oder verbreiteten Kopien zu behindern oder zu kontrollieren. Sie dürfen jedoch eine Vergütung im Austausch für Kopien annehmen. Wenn Sie eine ausreichend große Anzahl von Kopien verbreiten, müssen Sie auch die Bedingungen in Abschnitt 3 einhalten.

Sie können auch Kopien unter den oben genannten Bedingungen ausleihen und öffentlich ausstellen.

3. COPYING IN QUANTITY

Wenn Sie mehr als 100 gedruckte Exemplare des Dokuments veröffentlichen und der Lizenzhinweis des Dokuments Umschlagtexte verlangt, müssen Sie die Exemplare in Umschläge einlegen, die deutlich und lesbar alle diese Umschlagtexte enthalten: Vorderseitentexte auf dem vorderen Umschlag und Rückseitentexte auf dem hinteren Umschlag. Auf beiden Umschlägen müssen Sie außerdem deutlich und leserlich als Verleger dieser Exemplare ausgewiesen sein. Der vordere Umschlag muss den vollständigen Titel enthalten, wobei alle Wörter des Titels gleichmäßig hervorgehoben und sichtbar sein müssen. Sie können die Umschläge zusätzlich mit anderem Material versehen. Kopien mit Änderungen, die sich auf die Umschläge beschränken, können als wortgetreue Kopien behandelt werden, solange der Titel des Dokuments erhalten bleibt und diese Bedingungen erfüllt sind.

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. Verwenden Sie auf der Titelseite (und auf den Umschlägen, falls vorhanden) einen Titel, der sich von dem des Dokuments und von denen früherer Versionen unterscheidet (die, falls es welche gab, im Abschnitt ”Historie” des Dokuments aufgeführt sein sollten). Sie können denselben Titel wie eine frühere Version verwenden, wenn der ursprüngliche Herausgeber dieser Version seine Zustimmung gibt. B. Führen Sie auf der Titelseite als Autoren eine oder mehrere Personen oder Organisationen auf, die für die Urheberschaft der Änderungen in der geänderten Version verantwortlich sind, zusammen mit mindestens fünf der Hauptautoren des Dokuments (alle seine Hauptautoren, wenn es weniger als fünf hat). C. Geben Sie auf der Titelseite den Namen des Herausgebers der geänderten Version als Herausgeber an. D. Behalten Sie alle Urheberrechtsvermerke des Dokuments bei. E. Fügen Sie einen angemessenen Urheberrechtsvermerk für Ihre Änderungen neben den anderen Urheberrechtsvermerken ein. F. Fügen Sie unmittelbar nach den Urheberrechtsvermerken einen Lizenzhinweis ein, welcher der Öffentlichkeit die Erlaubnis gibt, die modifizierte Version unter den Bedingungen dieser Lizenz zu benutzen, und zwar in der Form, die im Anhang unten gezeigt wird. G. Behalten Sie in diesem Lizenzhinweis die vollständigen Listen der unveränderlichen Abschnitte und der erforderlichen Umschlagtexte bei, die im Lizenzhinweis des Dokuments angegeben sind. H. Fügen Sie eine unveränderte Kopie dieser Lizenz bei. I. Behalten Sie den Abschnitt mit dem Titel Geschichte” und seinen Titel bei und fügen Sie ihm einen Punkt hinzu, der mindestens den Titel, das Jahr, die neuen Autoren und den Herausgeber der modifizierten Version angibt, wie auf der Titelseite angegeben. Wenn es keinen Abschnitt mit dem Titel ”Geschichte” in dem Dokument gibt, erstellen Sie einen, der den Titel, das Jahr, die Autoren und den Herausgeber des Dokuments angibt, wie auf der Titelseite angegeben, und fügen Sie dann einen Punkt hinzu, der die geänderte Version beschreibt, wie im vorherigen Satz angegeben. J. Bewahren Sie den im Dokument angegebenen Netzwerkstandort, falls vorhanden,

für den öffentlichen Zugang zu einer transparenten Kopie des Dokuments auf, und ebenso die im Dokument angegebenen Netzwerkstandorte für frühere Versionen, auf denen es basierte. Diese können im Abschnitt ”Historie” abgelegt werden. Sie können eine Netzwerkadresse für ein Werk weglassen, das mindestens vier Jahre vor dem Dokument selbst veröffentlicht wurde, oder wenn der ursprüngliche Herausgeber der Version, auf die es sich bezieht, die Erlaubnis gibt. K. In jedem Abschnitt, der mit ”Danksagungen” oder ”Widmungen” betitelt ist, bewahren Sie den Titel des Abschnitts, und bewahren Sie in dem Abschnitt den gesamten Inhalt und Ton der Danksagungen und/oder Widmungen, die darin enthalten sind. L. Bewahren Sie alle unveränderlichen Abschnitte des Dokuments, unverändert in ihrem Text und in ihren Titeln. Abschnittsnummern oder das Äquivalent werden nicht als Teil der Abschnittstitel betrachtet. M. Streichen Sie jeden Abschnitt mit der Überschrift ”Vermerke”. Ein solcher Abschnitt darf nicht in die geänderte Fassung aufgenommen werden. N. Vorhandene Abschnitte dürfen nicht in ”Vermerke” umbenannt werden oder im Titel mit einem unveränderlichen Abschnitt kollidieren.

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

Die Free Software Foundation kann von Zeit zu Zeit neue, überarbeitete Versionen der GNU Free Documentation License veröffentlichen. Solche neuen Versionen werden im Geiste der gegenwärtigen Version ähnlich sein, können sich aber im Detail unterscheiden, um neue Probleme oder Anliegen zu behandeln. Siehe https://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.