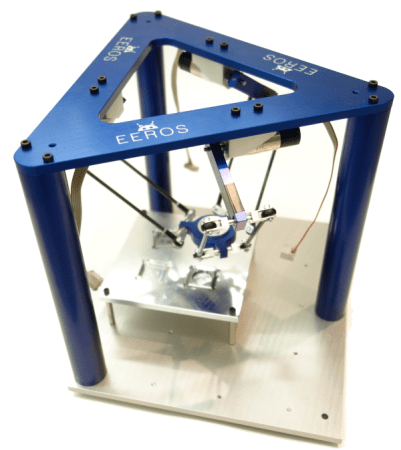
Delta Redesign



Verfasser: Simon Fink

Partner: INF Institut NTB Buchs

Referent: Prof. Dr. sc. Techn. Urs Graf

Korreferent: Prof. Einar Nielsen

Datum: 12.08.2018

Inhalt

[Aufgabenstellung 2](#_Toc521866112)

[Allgemeine Beschreibung 2](#_Toc521866113)

[Evaluation 3](#_Toc521866114)

[Hardware 3](#_Toc521866115)

[Beagle Bone Blue 3](#_Toc521866116)

[EEROS - Software 4](#_Toc521866117)

[Control System 4](#_Toc521866118)

[Safety System 4](#_Toc521866119)

[Sequencer 4](#_Toc521866120)

[HAL 5](#_Toc521866121)

[Theorie 6](#_Toc521866122)

[Berechnung Hardware 6](#_Toc521866123)

[Auslegung des Transistors für den Elektromagneten 6](#_Toc521866124)

[Spannungsteiler der Encoder Signale 6](#_Toc521866125)

[Berechnung HAL 7](#_Toc521866126)

[Encoder IEH2-4096 7](#_Toc521866127)

[Encoder HEM3-256 8](#_Toc521866128)

[Auslastung 9](#_Toc521866129)

[RTTest 9](#_Toc521866130)

[Simple Motor Controller 9](#_Toc521866131)

[Delta Roboter 10](#_Toc521866132)

[Abbildungsverzeichnis 11](#_Toc521866133)

[Eigenständigkeitserklärung 11](#_Toc521866134)

# Aufgabenstellung

Im Rahmen einer MSE Vertiefungsarbeit soll die Software des bestehenden EEDURO-Delta Roboters auf die aktuelle EEROS Version angepasst werden. Da dieser Roboter als Vorzeigebeispiel des EEROS-Frameworks dient, soll ein ausführliches Tutorial auf der EEROS Homepage erstellt werden.

# Allgemeine Beschreibung

Diese Arbeit beinhaltet die Ergänzung und Anpassung des bestehenden Source-Codes des Delta Roboters auf die aktuelle Version von EEROS. Dieser ist auf GitHub unter <https://github.com/simonfing/VTDelta> zu finden.

Die Tutorials befinden sich unter <http://hw.eeros.org/eeduro> . Die Step-by-Step Anleitung um sich einen Delta Roboter zu bauen befindet sich unter <http://hw.eeros.org/eeduro/delta/build_your_own>.

# Evaluation

## Hardware

In dieser Arbeit wird nur die Hardware sowie die Software des Delta Roboters behandelt. Folgende Hardware soll wiederverwendet werden:

1. 3 Stück Faulhaber 1524-012SR Motoren mit IEH2-4096 Encoder und 15-8 76:1 Getriebe.
2. 1 Stück Faulhaber 0816-012SR Motor mit HEM3-256 Encoder und 08/3 120:1 Getriebe
3. Elektromagnet GTO-14-0.5000 12VDC
4. 3 beleuchtete Drucktaster
5. LED Beleuchtung
6. Maus für manuelle Steuerung

## Beagle Bone Blue[[1]](#footnote-1)

Das Beagle Bone Blue ist ein Linux basierter “Single-Board-Computer”. Es verfügt über einen 1GHz ARM Cortex-A8 Prozessor, einen 512 MB DDR3 RAM sowie über zwei 32-bit programmierbare Real-Time Units (PRU).

Um den Delta Roboter zu steuern besitzt das Beagle Bone Blue:

* 4 Motorenausgänge,
* 4 Quadratur Encoder,
* 8 GPIOs und
* einen USB Anschluss.

Um dies mit dem bestehenden Delta Roboter verwenden zu können ist jedoch ein zusätzlicher Adapterprint nötig. Die Berechnungen für diesen Print sind in Kapitel Theorie – Berechnung Hardware zu finden. Das Schema des Adapterprints ist unter <http://hw.eeros.org/eeduro/delta/hardware> zu finden.

Die vier Motoren des Delta Roboters können direkt an den Motorenausgängen des Beagle Bone Blue angeschlossen werden. Diese können eine Spannung im Bereich von -8.3V bis +8.3V und einen Strom von bis zu 1.5A liefern.

Für die vier Encoder muss je Signalleitung ein Pegelwandler oder ein Spannungsteiler verbaut werden. Die Berechnung hierzu ist in Kapitel Theorie – Berechnung Hardware zu finden.

Der Elektromagnet wird über einen Transistor geschalten. Auch hier ist die Berechnung in Kapitel Theorie – Berechnung Hardware zu finden.

Die drei beleuchteten Drucktaster werden direkt an die GPIOs des Beagle Bone Blue angeschlossen. Insgesamt werden 7 GPIOs benötigt, drei für die LEDs in den Drucktastern, drei für die Taster und einen für den Elektromagneten. Die LEDs der Drucktaster benötigen 3.3V, daher können diese direkt mit dem GPIO gesteuert werden.

Die verbauten LED Bänder benötigen 12V. Da das Beagle Bone Blue ebenfalls mit 12V betrieben werden kann, können die LED Bänder direkt an diese angeschlossen werden.

Da das Beagle Bone Blue über einen USB 2.0 host verfügt, kann eine Maus an diesem angeschlossen werden.

## EEROS - Software

EEROS ist ein open source Real-Time Robotics Software Framework, entwickelt an der Interstaatlichen Hochschule für Technik in Buchs. Dieses Framework wurde und wird für die Regelung des Delta Roboters verwendet.

Es besteht aus dem Control System, dem Sequencer, dem Safety System und dem Hardware Abstraction Layer (HAL). Da der Sourcecode bereits besteht, jedoch mit einer älteren Version von EEROS entwickelt wurde, wird dieser auf den neuesten Stand gebracht. Nachfolgend werden die Änderungen der einzelnen Komponenten erklärt. Die genauen Funktionen der einzelnen Systeme werden nicht erklärt. Die Beschreibung zu diesen sind unter <http://wiki.eeros.org/eeros_architecture/start> zu finden.

### Control System

Im Control System gibt es nur kleinere Änderungen. So wurde der «Board»-Block gegen einen «Mux»-Block und einen «DeMux»-Block ausgetauscht. Der Mux nimmt die vier Encoder Signale entgegen und packt diese in einen AxisVector. Dem entgegengesetzt wandelt der DeMux einen AxisVector in vier «double»-Signale um und gibt diese Werte den entsprechenden Motoren weiter. Das Diagramm sowie die Beschreibung des Control Systems befindet sich unter <http://hw.eeros.org/eeduro/delta/software#control_system>.

### Safety System

Das Safety System wurde aus dem bestehenden Code übernommen. Es wurden die Zuweisungen der Safety Levels und Safety Events angepasst. Das Safety Level “slJoystickTeaching” wurde entfernt, da der Xbox-Controller nicht mehr verwendet wird. Es wurde jedoch ein neues Safety Level “slCalibrating” mit dem entsprechenden Safety Event “doCalibrating” eingefügt. Somit ist es während des laufenden Betriebes möglich eine Neukalibrierung durchzuführen. Die genaue Beschreibung des Safety Systems sowie dessen Diagramm ist unter <http://hw.eeros.org/eeduro/delta/software#safety_system> zu finden.

### Sequencer

Sequenzen waren im bestehenden Code schon vorhanden, jedoch hat sich der Sequencer in letzter Zeit geändert. Die Sequenzen wurden nun unterteilt in “Main Sequence”, “Sort Sequence”, “Shuffle Sequence”, “Mouse Sequence”, “MoveBlock Sequence” und “Calibration Sequence”. Die einzelnen Schritte der Sequenzen wurden in “Steps” unterteilt, hinter welchen ein einzelner Arbeitsschritt steht. Zudem wurden noch Monitore und “Exception Sequences” eingeführt. Der genaue Ablauf der Sequenzen ist dem Diagramm und der Beschreibung unter <http://hw.eeros.org/eeduro/delta/software#sequencer_&_sequences> zu entnehmen.

Die “Main Sequence” wird immer durchlaufen. Je nach aktuellem Safety Level schaltet die “Main Sequence” die entsprechende Sequenz hinzu.

Der “Sort Sequence” sowie der “Shuffle Sequence” wurde ein “Mouse Move Monitor” angehängt. Dieser überprüft laufend die Bewegung der Maus. Sobald diese bewegt wird, ruft der Monitor die “Move Mouse Exception Sequence” auf, welche die laufende Sequenz beendet und das Safety Event “doMouseTeaching” auslöst. Die “Main Sequence” ruft nun die “Mouse Sequence” auf.

Der “Mouse Sequence” wurde ein “Time Out Monitor” angehängt. Dieser überprüft wie lange es dauert, bis die “Mouse Sequence” durchlaufen ist. Bei bewegen der Maus, oder drücken einer Taste, wird dieser zurückgesetzt. Sobald die Zeit überschritten ist, wird die “Mouse Sequence Time Out Exception Sequence” aufgerufen, welche die “Mouse Sequence” beendet und das Safety Event “doAutoMoving” auslöst.

Um in die “Calibration Sequence” zu gelangen muss zuerst das Safety Event “doEmergency” ausgelöst werden. Dies geschieht, indem der rote Drucktaster betätigt wird. Die LED dieses Tasters beginnt nun zu leuchten. Durch 2 Sekunden langes drücken auf den roten Drucktaster wird nun das Safety Event “doCalibrating” ausgelöst. Die “Main Sequence” ruft nun die “Calibration Sequence” auf.

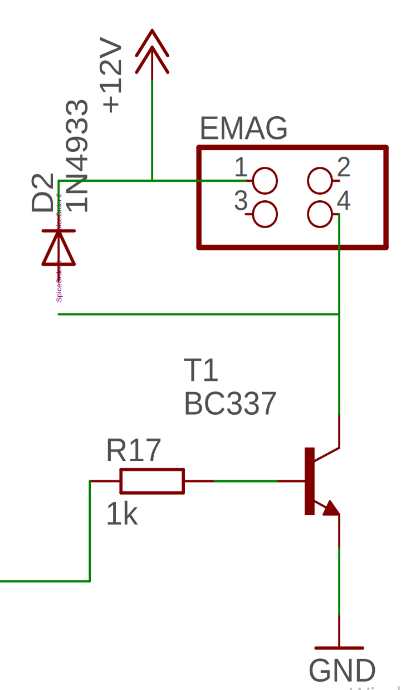
### HAL

Der Hardware Abstraction Layer wurde erst zu einem späteren Zeitpunkt als eigenes Element in EEROS hinzugefügt. Dementsprechend wurde für die neue Software des Delta Roboters ein HAL-Konfigurations-File erzeugt, in welchem die Motoren, Encoder und GPIOs des Delta Roboters beschrieben sind. Die Berechnung des HAL-Konfigurations-File befinden sich in Kapitel Theorie – Berechnung – HAL oder unter <http://hw.eeros.org/eeduro/delta/software#hal>.

# Theorie

## Berechnung Hardware

### Auslegung des Transistors für den Elektromagneten



GPIO

Abbildung : Schema Anschluss Elektromagnet

Da der Elektromagnet mit 12V betrieben werden kann, ein GPIO des Beagle Bone Blue jedoch nur 3.3V liefert, ist eine kleine Schaltung nötig.

Der Elektromagnet wird mit 12V betrieben und hat eine Leistung von 1.3W. Somit ergibt sich für den Kollektorstrom einen Wert von

.

Der BC337 [[2]](#footnote-2)liefert einen Kollektorstrom von bis zu 800 mA. Für einen Kollektorstrom von 108 mA ist laut Datenblatt ein Basisstrom von 0,8 mA bis 3 mA nötig. Da die GPIOs des Beagle Bone Blue 3.3V liefern ist hierfür ein Vorwiderstand von 1kΩ bis 4kΩ nötig.

Parallel zum Elektromagneten wurde in entgegengesetzter Stromrichtung eine Diode verbaut. Diese dient zum Schutz des Transistors und dem GPIO Port des Beagle Bone Blue. Sobald der Elektromagnet ausgeschaltet wird, erzeugt das noch vorhandene Magnetfeld einen Strom im Elektromagneten, welcher dann über Diode und den Elektromagnet fließt, bis er verschwindet.

### Spannungsteiler der Encoder Signale

Um die Spannung der Encoder Signale von 5V auf 3.3V zu bekommen wurde ein Spannungsteiler, mit je einem 330Ω und einem 680Ω Widerstand, verwendet. Die dadurch am 680Ω Widerstand und am Encoder Eingang des Beagle Bone Blue anliegende Spannung ist

.

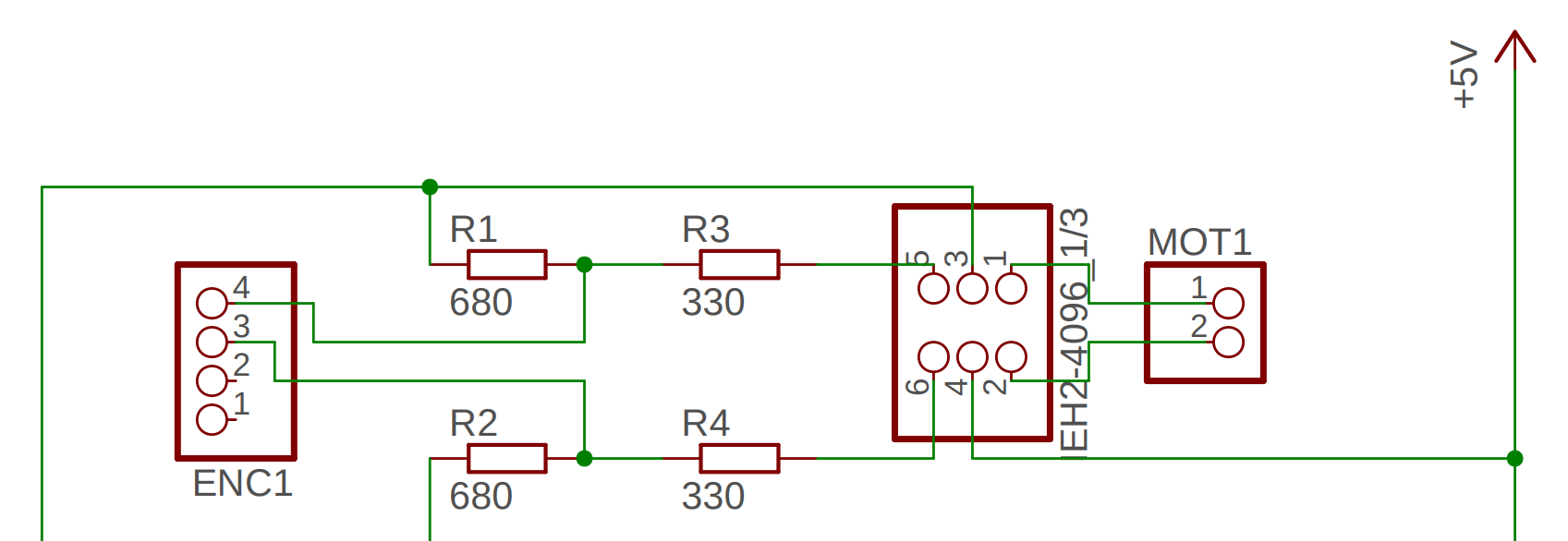
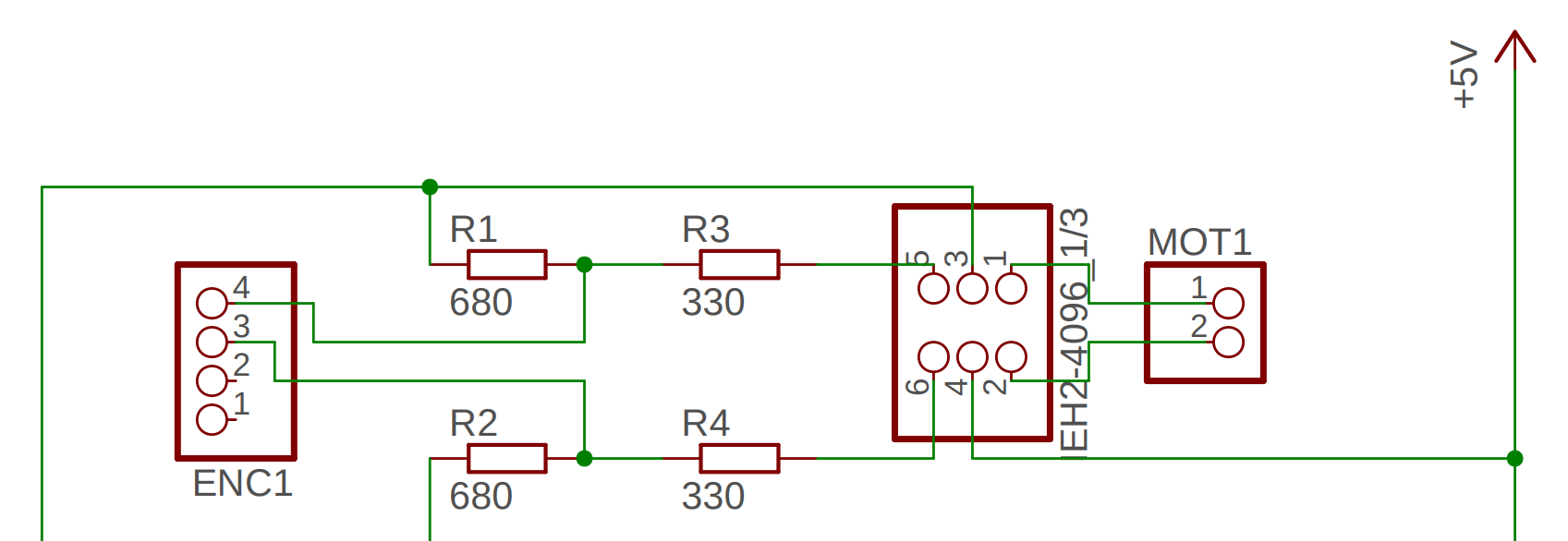


Abbildung : Spannungsteiler der Encoder

## Berechnung HAL

### Encoder IEH2-4096

Die Encoder der Achsenmotoren liefern 4096 Linien pro Umdrehung. Durch eine vier Quadranten Auswertung ergibt dies 16.384 Flanken pro Umdrehung. Da auf den Achsenmotoren ein Getriebe mit Übersetzung von 76:1 angebracht ist, entspricht 76 Umdrehungen des Motors einer Umdrehung an der Getriebeachse. Somit sendet der Encoder für jede Umdrehung der Getriebeachse

Flanken. Da sich diese Achse jedoch um höchstens 120° drehen soll wird die «scale» im HAL Konfigurationsfile auf diesen Wert angepasst.

Die Berechnung der maximal möglichen Positionen wurde in zwei Bereiche unterteilt, wobei nur einer der beiden berechnet wurde. Die Konstruktionszeichnungen der Bauteile sind unter <https://github.com/eeduro/delta-mechanics> zu finden.

Da die Gelenkverbinder (in Abbildung 1 als hellgraue Kreise gezeichnet) nicht in der Bauteilliste des Delta Roboters vorhanden waren und somit keine genauen Masse gegeben sind, wurde die Berechnung des unteren Anschlages nicht durchgeführt. Stattdessen wurde der Winkel α am Delta Roboter auf etwas mehr als 90° geschätzt.

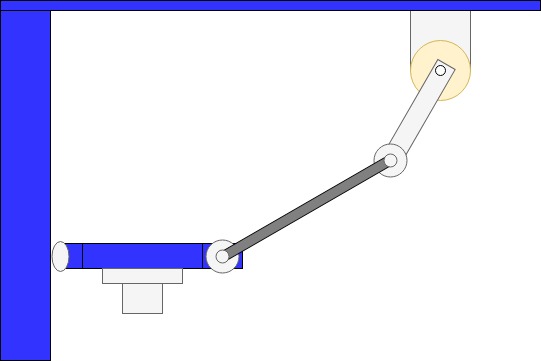
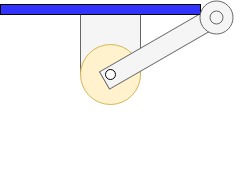


Abbildung : Delta Axis Motor - unterer Anschlag

α

17.5

15



6

β

Abbildung : Delta Axis Motor - oberer Anschlag

y

Für den oberen Anschlag wurde die Berechnung durchgeführt. Für diesen gilt:

Durch einsetzen und umformen ergibt sich

Die Subtraktion einer Kosinus Funktion von einer Sinus Funktion resultiert in einer Sinus Funktion mit Phasenverschiebung. Somit soll anhand der obigen Funktion

resultieren. Aus dieser kann durch das Additionstheorem

aufgestellt und die Koeffizienten vor sin() und cos() miteinander verglichen werden.

Durch dividieren der beiden Gleichungen ergibt sich die Phasenverschiebung der neuen Sinus Funktion

Durch quadrieren und addieren der beiden Gleichungen ergibt sich die Amplitude

Somit ist

Durch umformen erhält man nun

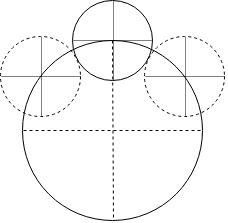
Somit beträgt der maximale Drehwinkel der Getriebeachse ~95°+25° = 120°.

### Encoder HEM3-256

Äquivalent zur Berechnung des ieh2 Encoders. Der hem3-256 liefert 256 Linien pro Umdrehung. Durch FQD ergibt das 1024 Flanken pro Umdrehung des Motors. Das Getriebe hat in diesem Fall jedoch ein Übersetzungsverhältnis von 120:1, somit ergibt dies 122.880 Flanken für eine Umdrehung der Getriebeachse. Der maximale Drehbereich wird jedoch auf 320° beschränkt, welches ebenfalls im HAL Konfigurationsfile mitberücksichtigt wird.

Der genaue maximale Drehwinkel wird wie folgt berechnet:

Aus den Konstruktionszeichnungen des Delta Roboters ist ersichtlich, dass sowohl auf dem «Rotating tool carrier[[3]](#footnote-3)» als auch auf dem «TCP link[[4]](#footnote-4)» ein Stift eingebaut wird, der eine komplette Umdrehung der Teile verhindert.



2\*r

Abbildung : "Rotating tool carrier" Beschränkung

D

γ

Die Stifte haben einen Durchmesser von 1.5 mm und liegen auf einem Kreis mit Durchmesser 50 mm. Aus den Mittelpunkten dieser Kreise bildet sich ein gleichschenkliges Dreieck. Anhand von Trigonometrie kann der Winkel α bestimmt werden.

Da dieser Anschlag von beiden Seiten gezählt werden muss ergibt sich ein gesamter Drehbereich von

# Auslastung

Die Auslastung der EEROS Applikation wurde mit Hilfe von «htop[[5]](#footnote-5)» gemessen.

## RTTest

Für eine minimale Grundlegende EEROS Anwendung wurde aus dem Tutorial der «Real Time Test[[6]](#footnote-6)» verwendet, und mit dieser die erste Messung aufgenommen. Diese Anwendung erzeugt einen periodischen Task, welcher jede Millisekunde aufgerufen wird. Dieser Task wird dem «Executor» hinzugefügt und wird dort ausgeführt. Der «Executor» selbst ist ebenfalls ein Task, welcher aber mit einer höheren Priorität ausgeführt wird. 

Abbildung : htop – rtTest

In Abbildung 6 unter PID 2177 ist die Anwendung «rtTest» zu sehen. Diese beinhaltet einen «Executor», welcher einen neuen Task mit höherer Priorität erzeugt, zu sehen unter PID 2178. Die Priorität des «Executor» ist -50 welche deutlich höher ist, als die der «rtTest»-Anwendung mit 20.

## Simple Motor Controller

Als nächstes Beispiel wurde ein etwas aufwändigeres Beispiel genommen. Ebenfalls im Tutorial zu finden unter «SimpleMotorController[[7]](#footnote-7)».

Diese Anwendung verwendet neben dem «Executor» noch: ein Control System; Safety Properties; eine Sequenz und ein HAL-Konfigurationsfile. Die Messung mit «htop» sieht folgendermassen aus:

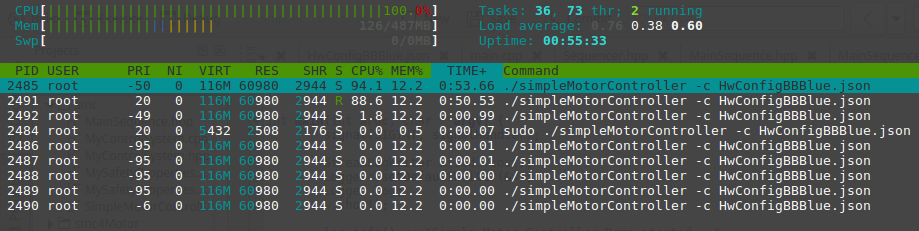


Abbildung : htop - Simple Motor Controller

Wie auch im einfachen EEROS Beispiel ist hier die Anwendung mit PID 2484 zu sehen, und der «Executor» mit PID 2485. Neu hinzugekommen sind Tasks für:

* Die TimeDomain, PID 2492, diese beinhaltet sämtliche Blocks des Control Systems
* Den Sequencer UI, PID 2490,
* Die Sequenz, PID 2491
* Vier Tasks mit hoher Priorität (-95). Diese werden von der «robotics cape library[[8]](#footnote-8)», welche EEROS für das Beagle Bone Blue verwendet, erzeugt. Hinter diesen Task stecken die In- und Outputs zur angeschlossenen Hardware.

## Delta Roboter

Die Anwendung des Delta Roboters beinhaltet noch zusätzliche Sequenzen, für welche einen Task angelegt wird.

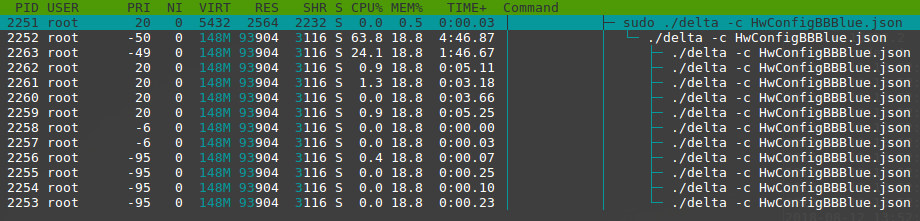


Abbildung : htop - Delta Roboter

So sind wie im vorherigen Beispiel folgende Tasks wieder zu finden:

|  |  |
| --- | --- |
| **PID** | **Beschreibung Task** |
| 2251 | Delta Anwendung |
| 2252 | Executor |
| 2253 | Task von robotics cape |
| 2254 | Task von robotics cape |
| 2255 | Task von robotics cape |
| 2256 | Task von robotics cape |
| 2257 | Sequencer UI |
| 2259 | Main Sequence |
| 2260 | Sort Sequence |
| 2261 | Shuffle Sequence |
| 2262 | Mouse Sequence |
| 2263 | Time Domain |

Tabelle : Tasks des Delta Roboters

# Abbildungsverzeichnis

[Abbildung 1: Schema Anschluss Elektromagnet 6](file:///C:\Users\simon\Documents\MSE\VT-Delta\git\Bericht-VTDelta.docx#_Toc521866135)

[Abbildung 2: Spannungsteiler der Encoder 6](file:///C:\Users\simon\Documents\MSE\VT-Delta\git\Bericht-VTDelta.docx#_Toc521866136)

[Abbildung 3: Delta Axis Motor - unterer Anschlag 7](file:///C:\Users\simon\Documents\MSE\VT-Delta\git\Bericht-VTDelta.docx#_Toc521866137)

[Abbildung 4: Delta Axis Motor - oberer Anschlag 7](file:///C:\Users\simon\Documents\MSE\VT-Delta\git\Bericht-VTDelta.docx#_Toc521866138)

[Abbildung 5: "Rotating tool carrier" Beschränkung 8](file:///C:\Users\simon\Documents\MSE\VT-Delta\git\Bericht-VTDelta.docx#_Toc521866139)

[Abbildung 6: htop – rtTest 9](#_Toc521866140)

[Abbildung 7: htop - Simple Motor Controller 9](#_Toc521866141)

[Abbildung 8: htop - Delta Roboter 10](#_Toc521866142)

# Eigenständigkeitserklärung

Hiermit bestätige ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen sind, wurden unter Angabe der Quelle kenntlich gemacht.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Simon Fink Datum

1. https://beagleboard.org/blue [↑](#footnote-ref-1)
2. http://www.onsemi.com/pub/Collateral/BC337-D.PDF [↑](#footnote-ref-2)
3. https://github.com/eeduro/delta-mechanics/blob/master/EEDURO-D-011-00.pdf [↑](#footnote-ref-3)
4. https://github.com/eeduro/delta-mechanics/blob/master/EEDURO-D-009-00.pdf [↑](#footnote-ref-4)
5. https://wiki.ubuntuusers.de/htop/ [↑](#footnote-ref-5)
6. http://wiki.eeros.org/getting\_started/tutorials/rttest [↑](#footnote-ref-6)
7. http://wiki.eeros.org/getting\_started/tutorials/oneaxis [↑](#footnote-ref-7)
8. https://github.com/StrawsonDesign/librobotcontrol/tree/master/library [↑](#footnote-ref-8)