

UNIVERSITY OF APPLIED SCIENCES TRIER

BACHELOR PROJEKTARBEIT

# Entwicklung und Implementierung eines Can-Interfaces zur präzisen Lageregelung eines Portalkrans

BACHELOR OF ENGINEERING

ELEKTROTECHNIK  
INFORMATIONSTECHNOLOGIE UND ELEKTRONIK

verfasst von:  
Torsten ZIMMERMANN. [966352]

Teamprojekt mit:  
Steave DAHM. [969471]

betreut von:  
Prof. Dr. Ing. Matthias SCHERER  
Andreas REIS, M. Sc.

Abgabedatum:  
13. Oktober 2020

---

# Eidesstattliche Erklärung

Ich versichere, die Bachelor-Projektarbeit selbstständig und lediglich unter Benutzung der angegebenen Quellen und Hilfsmittel verfasst zu haben.

Ich erkläre weiterhin, dass die vorliegende Arbeit noch nicht im Rahmen eines anderen Prüfungsverfahrens eingereicht wurde.

---

Ort, Datum

---

# Kurzfassung

Im Rahmen einer Bachelor Projektarbeit wird für den Portalkran Versuch des Regelungstechnik Labors eine Ansteuerung über ein Can Interface Entwickelt und Implementiert. Dabei übernimmt eine Regeleinheit namens Unicontrol, welche auf einer Studentischen Master Arbeit aufbaut, die komplette Regelung des Versuchs. Zudem besitzt die Laufkatze eine Zusatzplatine mit mehreren Sensoren und Aktoren, die in meinem Teil der Arbeit noch programmiertechnisch implementiert werden mussten. Das Can Interface hat die Aufgabe Unicontrol, das Motorsteuergerät und die Platine, welche ebenfalls auf einer Studentischen Arbeit besteht mit einander kommunizieren zu machen. Hierbei musste darauf geachtet werden, dass das Motorsteuergerät mit Can-Open kommuniziert. Unicontrol und die Zusatz Platine kommunizieren über Can-Automotive. Dabei wurden auch verschiedene Sicherheitsmechanismen implementiert, sodass niemand zu Schaden kommen kann.

Die Ansteuerung soll über einen externen Rechner von Zuhause aus für Studierende möglich gemacht werden. Dies hat den Vorteil, dass Studierende in Zukunft das Regelungstechnik-Labor absolvieren können auch über das Wochenende hinweg. Der komplette Aufbau ist im Anhang nochmal durch ein Bild kenntlich gemacht worden. Zum besseren Verständnis, wird auch nochmal kurz auf den Part meines Teampartners eingegangen, der sich um den kompletten mechanischen Aufbau des Portalkrans gekümmert hat.

---

# Abstract

As part of a bachelor projekt for the control engineering laboratory, a control via a CAN interface is being developed and implemented for the gantry crane attempt. A control unit called Unicontrol, which is based on a student master's thesis, takes over the complete control of the experiment. In addition, the trolley has an additional circuit board with several sensors and actuators, which in my part of the work still had to be implemented in programming. The Can Interface has the task of making Unicontrol, the engine control unit and the circuit board, which is also based on a student work, capable of communicating with each other. It had to be ensured that the engine control unit communicates with Can-Open. Unicontrol and the additional circuit board communicate via Can-Automotive. Various security mechanisms have also been implemented so that nobody can be harmed.

The control should be made possible for students from home via an external computer. This has the advantage that in future students will be able to complete the control engineering laboratory over the weekend. The complete structure is shown again in the appendix by a picture.

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>6</b>
1.1	Aufgabenstellung . . . . .	6
1.2	Herangehensweise . . . . .	6
1.2.1	Betriebssystem . . . . .	6
1.2.2	Entwicklungsumgebung . . . . .	7
1.2.3	Sicherheit . . . . .	7
1.2.4	Simulink . . . . .	7
1.2.5	Asap Editor . . . . .	7
1.2.6	Control Desk . . . . .	8
1.2.7	XCP . . . . .	8
<b>2</b>	<b>Initialisierung der Peripherie Module</b>	<b>9</b>
2.0.1	QEI-Module . . . . .	9
<b>3</b>	<b>Fazit</b>	<b>10</b>
3.1	Aktueller Stand . . . . .	10
3.2	Ausblick . . . . .	10
	<b>Abbildungsverzeichnis</b>	<b>11</b>
	<b>Tabellenverzeichnis</b>	<b>12</b>
	<b>Literaturverzeichnis</b>	<b>13</b>

# 1 Einleitung

## 1.1 Aufgabenstellung

Die zentrale Aufgabe dieser Projektarbeit war die Vernetzung der Regeleinheit Unicontrol zusammen mit dem Motorsteuergerät und der eigens entwickelten Platine.

Die Zusatz Platine welche auf einer studentischen Arbeit aufbaut besitzt Sensoren und Aktoren. Vorhandene Sensoren sind der Linearencoder, der Winkencoder und der Hall Sensor. Vorhandene Aktoren sind der Hubmagnet und der Servomotor. Diese Peripherie Module der Zusatz Platine sollten zudem initialisiert und getestet werden. Dabei war besonders darauf zu achten, dass der Linearencoder über die komplette Distanz keine Striche des Codierfilms überspringt. Kommuniziert wurde wie bei Unicontrol über Can-Automotive. Can hat unter anderem den Vorteil, dass auf Ereignisse sofort reagiert werden kann, sofern die Message ID hoch priorisiert ist. Die Informationen über den Winkencoder und den Linearencoder werden sobald die Laufkatze initialisiert ist, alle 10 ms übertragen. Damit kann man das System als echtzeitfähig bezeichnet werden.

Die Regeleinheit Unicontrol sollte über die Umgebung Controldesk welche auch im Automotive Bereich genutzt wird, steuerbar sein. Weiter unten ist diese Umgebung nochmal gezeigt und erklärt. Zudem wird der Regler in Software implementiert. Dies ist möglich indem in Simulink der Regler aufgebaut wird und in C-Code kompiliert wird und dem Unicontrol Code hinzugefügt wird.

Das Motorsteuergerät verwendet zur Kommunikation eine Can-Open Verbindung. Dies machte es nicht möglich wie zuvor die Driverlib zu nutzen. Deshalb mussten hier 2 Funktionen integriert werden um die Can Nachrichten im gewünschten Format zu verpacken und zu entpacken. Es wurde gefordert, den Motor im Current Mode zu betreiben.

## 1.2 Herangehensweise

### 1.2.1 Betriebssystem

Jeweils Unicontrol sowie die Platine nutzen das Betriebssystem TI-RTOS[1]. Eine Besonderheit ist dabei, dass es sich um ein Echtzeitbetriebssystem handelt. Das bedeutet, dass Code in einem bestimmten Zeitfenster auf jeden Fall ausgeführt wird. Speziell als Beispiel dieses Projektes ist zu nennen, dass alle 10 ms die Winkel und Winkencoder Werte angefragt werden und auch alle 10 ms zur Auswertung bereit stehen müssen um den weiteren Betrieb möglich zu machen.

### 1.2.2 Entwicklungsumgebung

Als Entwicklungsumgebung kommt Code Composer Studio zum Einsatz, welche speziell für die Entwicklung von Applikationen von TI Mikrocontroller entwickelt wurde. Unicorn verwendet den Tiva? TM4C1294NCPDT Mikrocontroller. Die Zusatz Platine verwendet den Tiva? TM4C123GH6PM Mikrocontroller.

### 1.2.3 Sicherheit

Erst einmal ist die Einstellbare Position in ControlDesk für den Nutzer auf die vorhandene Strecke begrenzt. Dies macht es sofern alles ordnungsgemäß funktioniert unmöglich für den Nutzer über die Position hinaus zu fahren. Sollte es jedoch mal zum Problem kommen zum Beispiel durch mechanische Abnutzung des Kodierfilms, besitzt die Laufkatze Magnete an beiden Enden als Endschalter. Diese sollen, sobald der Hall Sensor diese im initialisiertem Betrieb erkennt, die Platine komplett abschalten. Vorgebeugt werden soll dieser Fehler jedoch schon direkt von der Initialisierung. Diese fährt erst einmal die komplette Strecke ab und prüft die Funktion der beiden Magneten. Sind die Magneten erst einmal erkannt, wird noch geprüft ob keine Striche auf der Strecke verloren gegangen sind. Das Steuergerät schaltet automatisch ab, sobald das Can Kabel gekappt wird. Dies ist auch sehr sinnvoll, da nun keine Informationen mehr ankommen, das der Motor aufhören soll zu fahren. Diese Funktion wurde programmier technisch auch bei der Platine realisiert.

Solange diese Initialisierung nicht abgeschlossen ist, verfügt der Benutzer nicht über alle Funktionen von ControlDesk. Durch diese Initialisierung wird sicher gestellt, das die Magnete ordnungsgemäß befestigt sind und später als Endschalter funktionieren. Zudem wird durch die Initialisierung bestätigt, das keine Streifen auf der Strecke Verloren gehen.

### 1.2.4 Simulink

Es ist geplant, den kompletten Regler in Simulink aufzubauen. Simulink erzeugt aus dem kompletten Modell C-Code und fügt diesen automatisch in Code Composer Studio hinzu. Hierbei war darauf zu achten, das man die Variablen, die später über ControlDesk verstellbar sind, extra eingepflegt werden.

### 1.2.5 Asap Editor

In Asap Editor ist es möglich eine .A2L Datei zu erstellen. Die .A2L Datei wird später von dem Programm Control Desk benötigt. In dieser Datei wird unter anderem eine Zuordnung zwischen dem symbolischen Variablennamen und zugehörigen Adressbereich getroffen. Dies soll es später möglich machen, die Variablen Werte während der Laufzeit zu verändern.

### **1.2.6 Control Desk**

ControlDesk wird hier zusammen mit einer GUI dem Benutzer zur Verfügung gestellt. Von hier ist es möglich, Variablen Werte während der Laufzeit des Programm zu verändern. Variablen Werte konnten hier in der GUI über Verschiedene Komponenten verstellt werden. Möglich macht das verändern der Werte aber in wirklichkeit die XCP Verbindung. ControlDesk dient hier nur als GUI.

### **1.2.7 XCP**

Eine XCP Verbindung wird benötigt um während der Laufzeit die verschiedenen Parameter zu Verändern oder auszulesen. Die XCP Verbindung läuft über den Can 1 Anschluss des Unicontrols. Bei XCP handelt es sich um ein universelles Mess- und Kalibrierprotokoll, welches komplett unabhängig vom verwendeten Netzwerktyp verwendet werden kann. Möglich ist zum Beispiel wie hier XCP on CAN oder aber auch XCP on USB usw.



## 2 Initialisierung der Peripherie Module

### 2.0.1 QEI-Module

Der Linearencoder erzeugt ein Quatratur Signal sobald es über den Codierfilm der Laufkatze fährt. Dieses Quatratur Signal wird an die entsprechenden Pins des Mikrocontrollers geführt, die ein Module beinhalten zum auswerten eines solchen Quatratur Signals. Ausgewertet wurde das Signal mit Funktionen, die einem zur Verfügung standen durch eine Driverlib.

## 3 Fazit

### 3.1 Aktueller Stand

In dem Projekt konnten alle gestellten Anforderungen erfolgreich umgesetzt werden. Es ist möglich die komplette Laufkatze an eine klar gewünschte Position zu Regeln. Möglich macht das der Regler der in Matlab implementiert wurde. Zudem konnten alle Sensoren und Aktoren erfolgreich in Betrieb genommen werden. Interessant zu sehen war hier, das die QEI keine Striche über die komplette Strecke verliert. Die Sicherheitsmechanismen wurden zudem auch erfolgreich Implementiert und getestet, sodass niemand zu schaden kommen kann. Zum besseren Verständnis wurde der C-Code auch nochmal sorgfältig mit Kommentaren versehen um später Änderungen schnell möglich zu machen.

### 3.2 Ausblick

# Abbildungsverzeichnis

# Tabellenverzeichnis

## Literaturverzeichnis