|  |
| --- |
| Fachhochschule Frankfurt am Main |
| Batchlorarbeit: Entwicklung einer Seegangskompensation für Krananlagen |
|  |
|  |
| **Benjamin Lesky** |
| **15.07.2013** |

Inhalt

[Danksagungen 2](#_Toc366147340)

[1 Problehmbeschreibung 3](#_Toc366147341)

[1.1 Möglichkeiten zur Seegangs Kompensation 3](#_Toc366147342)

[1.2 Vereinfachungen und Notwendige Einschränkungen 3](#_Toc366147343)

[2 Hardwareentwicklung 4](#_Toc366147344)

[2.1 Modell 4](#_Toc366147345)

[2.2 Sensoren 4](#_Toc366147346)

[2.3 Micro-Controller 4](#_Toc366147347)

[2.4 Motor und Motortreiber 4](#_Toc366147348)

[2.5 Verschaltung der Komponenten 4](#_Toc366147349)

[3 Systehmmodellierung 6](#_Toc366147350)

[3.1 Beschreibung der Regelstrecke 6](#_Toc366147351)

[3.2 Simulation mittels BORIS 6](#_Toc366147352)

[3.3 Liniarisierung 6](#_Toc366147353)

[3.4 Kompensation 6](#_Toc366147354)

[3.5 Entwicklung der Regleralgorythmus 6](#_Toc366147355)

[4 Softwareentwicklung 8](#_Toc366147356)

[4.1 Plahnung 8](#_Toc366147357)

[4.2 Programmierung 8](#_Toc366147358)

[5 Test 8](#_Toc366147359)

[5.1 Testroutine mittels Präprozessoren 8](#_Toc366147360)

[6 Anhänge 9](#_Toc366147361)

[6.1 Programmcoed 9](#_Toc366147362)

[6.2 Schaltplan 12](#_Toc366147363)

[6.3 UML diagramme 12](#_Toc366147364)

[7 Literaturverzeichnis 14](#_Toc366147365)

# Danksagungen

Zuerst möchte ich mich bei Herrn Proffessor Stief für die Herforragende betreuung während meiner Batchlorarbeit bedanken.

Sowie bei Herrn Mechallik welcher mir bei Problemen mit dem Microcontroller mit rat und tat zur seite stand.

Familie und Freunden für iehr Verstänntnis und iehre unterstützung warend der Batchlorarbeit und während des Studiums.

# 1 Problehmbeschreibung

Bei der Installation von Offshorewindanlagen ist es wichtig, dass eine an einem Kran hängende Last konstant in der gleichen Höhe über dem Meeresgrund gehalten werden kann. Die Position der Last ändert sich jedoch durch den Seegang konstant. Hierdurch wird eine Kompensation des Seegangs nötig.

Ziel der Batchlorarbeit ist es, die durch Seegang verursachte Höhenänderung zu Kompensieren. Der Abstand zum Meeresgrund wird mit einem Sensor erfasst. Die Hauptstöhrgröße in Form der Höhenänderung wird mit einem Beschleunigungssensor erfasst. Der Sollwert wird vom Kranfahrer eingestellt.

Es gibt mehrere Möglichkeiten Seegang zu kompensieren:

- Rumpfform

- Stabilisatoren

- Position des Krans auf dem Schiff möglichst weit mittschiffs

Beides keine 100% Kompensation daher Reglungstechnik mit verschiedenen möglichen ansetzen(normale Reglung nicht ausreichend):

-Vorhersage

-Störgrößenaufschaltung

Für diese Batchelorarbeit wird davon ausgegangen, dass sich das Montageschiff konstant auf einer Position befindet, das heist es macht keine Fahrt über Grund und es ist bei ruhiger See nicht mit Krängung (Bewegung um die Querachse) des Schiffes zu rechnen.

Durch Eigenantrib oder Schlepper wird das Schiff am Abtreiben und am Schwojen gehindert.

Des weiteren reduzieren folgende Schiffsbaulichen Maßnahmen den Seegang:

-Stabilisatoren verhindern Rollen (Bewegung um die Längsachse)

- Eine geeignete Rumpfform reduziert die Auswirkungen des Seeganges

Daraus folgt das man im Rahmen dieser Batchelorarbeit davon ausgehen kann das es sich bei Seegang um ein eindimensionales Problem handelt.

# 2 Hardwareentwicklung

## 2.1 Modell

Das Modell wurde mithilfe eines Baukastens der Firma Fichertechnik gebaut. Motor und Controller sind in das Modell integriert.

## 2.2 Sensoren

- Messbereich

- Eingangssignahl

Der Infrarot-Entfernungssensor hat einen Messbereich von 10-80cm. Das Analoge Ausgangssignal variiert zwischen 0,4 V und 2,3V.

## 2.3 Micro-Controller

Zur Regelung des Systems wird ein PSoC1 Evaluation Kit mit einem Prozessor vom Typ CY8C27446-24PXI verwendet. Auf dem mitgelieferten LCH-Display wird die Entfernung der Last zum Entfernungssensor angezeigt. Das System wird durch das Bord mit mit einer Versorgungsspannung von 5 Volt versorgt.

Zur Ansteuerung des Motors wird ein Treiberbaustein vom Typ L293 D verwendet dieser enthält 2 H-Brücken von denen eine zur Motorsteuerung verwendet wird. Durch eine Pulsweitenmudulation am Enable-Eingang wird die Drehzahl gesteuert.

Der gleichstrohmmotor der Firma Fichertechnik benötigt eine Versorgungsspannung von 9 Volt daher muss zusätzlich zu der 5 Volt Spannungsversorgung das PSoCs eine 9V Spannungsquelle am Motottreiber angeschlossen werden.

Infrarot und Beschleunigungssensor Geben beide eine Analoge Spannung aus. (Mehr Schreiben)

## 2.5 Verschaltung der Komponenten

-Pinbelegung

-LCD-Port 2

-Eingänge Port 0

-Ausgängr Pirt 1

|  |  |
| --- | --- |
| Pin | Belegung |
| 0.1 | Beschleinigungssensor |
| 0.4 | Enfernungssensor |
| 0.5 | Sollwert |
| 1.3 | Selbsttest des Beschleunigungssensors |
| 1.4 | Pulsweitenmudulation |
| 1.5 | Drehrichtungsumkehr |
| 2.0-2.7 | LCD-Display |

- Entwicklung/Pinbelegung

-Schaltplahn

# 3 Systehmmodellierung

## 3.1 Beschreibung der Regelstrecke

Die Regelstrecke besteht im wehsentlichen aus einer Seilwinde, welche sich wie ein I-Glied verhällt.

-Störgröße

- Berechnung der Streckenparameter

- Blockschaltbild

## 3.2 Simulation mittels BORIS

Mithillfe des Programms BORIS wurde das Systehm simuliert und die Regelung auf funktion geprühft.

- Verifikation des Korekten Verhaltens

-Bestimmung der Ersten Reglerparameter

## 3.3 Liniarisierung

Da bei der Entfernubgsmessung kein linearer Zusammenhang zwischen gemessener Entfernung und ausgegebener Spannung besteht muss dies im Programm berücksichtigt werden. Der Relevante Teil der Kurve entspricht annähernd einer E-Funktion mit der Funktionsgleichung:

Mit Hilfe des Programms skidavis können nun die Parameter a und b bestimmt werden. Durch einsetzen in die Funktionsgleichung erhält man eine Funktion, die ?????????????

Dieser Zusammenhang kann direkt im Algorithmus berücksichtigt werden

## 3.4 Kompensation

- umkerfunktion strecke

- Sinus von Sinus abhängig

-durch totzeit keine follständige kompensation

## 3.5 Entwicklung der Regleralgorythmus

Aus dem Blockschaldbild läst sich folgende Gleichung bestimmen:

Bzw

Diese muss zum Einsatz im Programm noch modifiziert werden.(Schlechte vormolierung) Da die Regeldifferenz noch nicht bekannt ist muss sie erst noch im Microcontroller berechnet werden.

Da der Microkontroller mit zeitdiskreten Signalen Arbeitet muss die Gleichung entsprechend angepasst werden und das Integral kann durch eine Summe ersetzt werden.

Da die Abstandsmessung nicht linear erfolgt muss dies in der Gleichung berücksichtigt werden. .(Schlechte vormolierung)

Im Programm wird Die Gleichung dann in zwei Schritten berechnet. Zuerst wird die Summe bestimmt.

Anschließend wird die Ausgangs Größe bestimmt.

# 4 Softwareentwicklung

## 4.1 Plahnung

Vor beginn der Programmierung wurde ein Ein Ablaufdiagramm nach UML Version 1.4 erstellt um ????????????

Zusetzlich zu den üblichen Konventionen wird folgendes Festgelegt:

-Komentare in Deutch

-Ungaricge Notation nach Charles Simonyi (Apps Hungarian)

Zusätzlich definierte Präfixe:

|  |  |
| --- | --- |
| Pd | Prozessbezogene Daten |
| Ko | Konstanten |
| Hi | Interne Hilfsgrößen |
| In | Datentyp Integer |

Zum versionsmanegemant wird die Internettplattform GitHub benutzt. Dadurch ist jede Programmänderung nachvollziehbar und es kann jederzeit an eine früheren stand zurückgekehrt werden. Desweiteren exestiert immer eine Sicherheitskopie (Vormulierung).Das Projekt ist zu finden unter:

<https://github.com/lesky/Batchlorarbeit--Seegangskompensation-bei-Krananlagen>

- Übergabe mit globalen variablen schlechter Stiel

## 4.2 Programmierung

- Test mit Präprozessoren

-Viele PSOC Funktionen

# 5 Test

## 5.1 Testroutine mittels Präprozessoren

-Daten Von eingängen werden direkt am Bildschierm ausgegeben

# 6 Anhänge

## 6.1 Programmcoed

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* C main-Funktion \*

\* Programm: Seegangskompensation bei Krahnanlagen \*

\* Controler: CY8C27446-24PXI \*

\* \*

\*I/O-Konfiguration: \*

\* LCD -> Port 2 \*

\* Selbsttest -> 1.1 \*

\* IN1 -> 1.2 \*

\* IN2 -> 1.3 \*

\* PWM 1 -> 1.4 \*

\* Beschleunigung-> 0.1 \*

\* Entfernung -> 0.4 \*

\* Sollwert -> 0.5 \*

\* \*

\*Zusätzlich definierte Prüfixe zur Variablenbenennung \*

\*Mittels Ungaricher Notation nach Charles Simonyi: \*

\*pd -> Prozessbezogene Daten \*

\*ko -> Konstanten \*

\*hi -> Interne Hilfsgrößen \*

\*in -> Datentyp Integer \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <stdio.h>

#include <m8c.h>

#include "PSoCAPI.h"

// Präprozessor:Für testzwecke

// #defein Test nicht auskomentieren

// #define TEST

// Funktionsprototypen:

void LCDansteuern(char);

void Dateneinlesen(void);

void Ausgangansteuern(char);

// Präprozessor: kompiliere Funktion nur wenn Test

#ifdef TEST

void test(char);

#endif

// globale Structur zur Übergabe der Prozessdaten:

struct

{

char pdchBechleunigung, pdchEntfernung;

char pdchSollwert;

char pdchPulsweite;

} prozess;

// Reguläre mani Funktion

void main(void)

{

// Difinition der Konstanten

char kochPeriodendauer = 50;

char kochKP;

char kochKS;

// Variablendeklration

char hichAusgangswert;

char hichBeschleunigungssumme;

// Initialisierung des Controlers

//globale Interrupts Freigeben

M8C\_EnableGInt;

// Initialisieren des LCD-Displays

LCD\_1\_Start();

// Initialisieren des PWM-Moduls

PWM8\_1\_WritePeriod(kochPeriodendauer);

PWM8\_1\_Start();

// Initialisieren der Verstärker

PGA\_1\_SetGain(PGA\_1\_G0\_12);

PGA\_2\_SetGain(PGA\_2\_G0\_12);

PGA\_3\_SetGain(PGA\_3\_G0\_12);

PGA\_1\_Start(PGA\_1\_LOWPOWER);

PGA\_2\_Start(PGA\_2\_LOWPOWER);

PGA\_3\_Start(PGA\_3\_LOWPOWER);

// Initialisieren des Dualen AD-Wandlers

// für Entfernung und Beschleunigung

DUALADC8\_Start(DUALADC8\_HIGHPOWER);

DUALADC8\_SetCalcTime(100);

DUALADC8\_GetSamples();

// Initialisieren des AD-Wandlers

// für den Sollwert

ADCINC\_Start(ADCINC\_HIGHPOWER);

ADCINC\_GetSamples(0);

//Initialisieren der Digitalen Ausgänge

IN1\_Start;

IN2\_Start;

SELBSTTEST\_Start;

// Präprozessor: kompiliere whileschleife wenn kein test

#ifndef TEST

// Endlosschleife

while(1)

{

// Daten Einlesen

Dateneinlesen();

// Parameter Berechnen

hichBeschleunigungssumme = hichBeschleunigungssumme + prozess.pdchBechleunigung;

hichAusgangswert = ( prozess.pdchSollwert - prozess.pdchBechleunigung ) \* kochKP

- 1 / kochKS \* hichBeschleunigungssumme;

//TODO: Korekturfaktor Einfügen

prozess.pdchPulsweite = hichAusgangswert;

Ausgangansteuern(hichAusgangswert);

LCDansteuern(prozess.pdchEntfernung);

};

// Präprozessor: kompiliere whileschleife wenn test;

#else

// whileschleife zu testzwecken

// Konstante zur verweildauer in der schleife

while (1)

{

// Daten Einlesen

void Dateneinlesen(void);

// Daten Nacheinander auf LCD Ausgeeben

test(prozess.pdchBechleunigung);

test(prozess.pdchEntfernung);

test(prozess.pdchSollwert);

};

// Präprozessor: Ende der Verzweifung

#endif

}

// Funktionen:

void LCDansteuern(char hichdata)

{

char rgchErstzeile[12];

// LCD Ansteuern

csprintf(rgchErstzeile,"Abstand:%c",hichdata);

LCD\_1\_Position(1,0);

LCD\_1\_PrString(rgchErstzeile);

}

void Dateneinlesen(void)

{

// Wenn Sollwertdaten bereit sind

if(ADCINC\_fIsDataAvailable() != 0)

// Einlesen des Sollwertes

// data ready flag zurüvksetzen

prozess.pdchSollwert = ADCINC\_cClearFlagGetData();

// Auf Entfernung und Position Warten

while(DUALADC8\_fIsDataAvailable == 0);

// Einlesen der Beschleunigung

prozess.pdchBechleunigung = DUALADC8\_cGetData1();

// Einlesen der Entfernung

// data ready flag zurüvksetzen

prozess.pdchEntfernung = DUALADC8\_cGetData2ClearFlag();

}

void Ausgangansteuern(char hichAusgangswert)

{

// linksdrehend

if (hichAusgangswert > 0){

IN1\_On;

IN2\_Off;

PWM8\_1\_WritePulseWidth(prozess.pdchPulsweite);

}

// rechtsdrehend

else if (hichAusgangswert < 0){

IN1\_On;

IN2\_Off;

PWM8\_1\_WritePulseWidth(-prozess.pdchPulsweite);

}

// Bremsen durch Kurzschluss sinvoll???

else{

IN1\_On;

IN2\_On;

PWM8\_1\_WritePulseWidth(0);

}

}

// Präprozessor: kompiliere Funktion nur wenn Test

#ifdef TEST

// Ausgabe der Testdaten auf LCD-Display

void test(char hichdata)

{

// gibt 99999 mal LCD Aus

int iin;

for (iin = 0; iin <= 99999; iin ++)

{

LCDansteuern(hichdata);

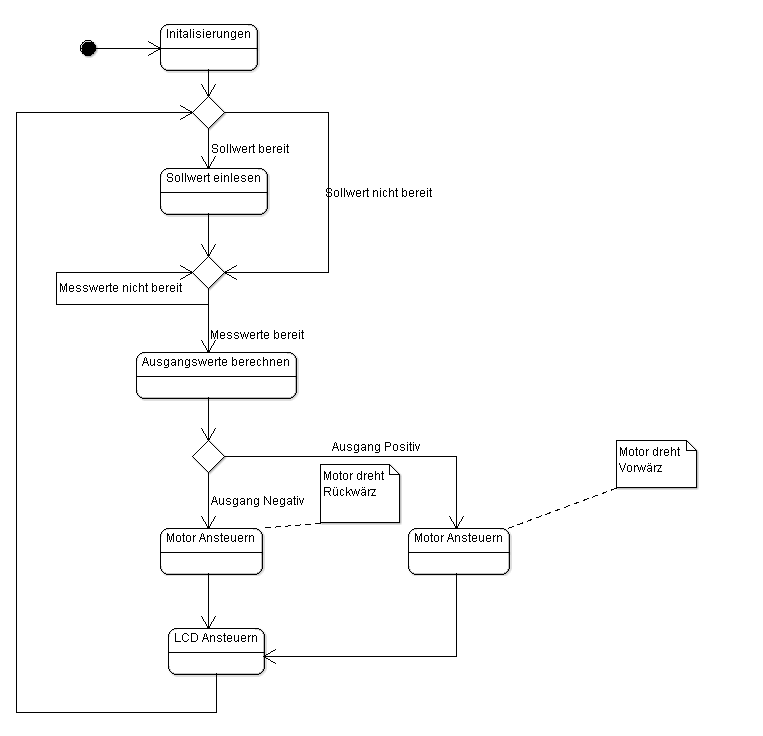
}

}

#endif

## 6.2 Schaltplan

## 6.3 UML diagramme



# 7 Literaturverzeichnis

(kein Datum). Abgerufen am 22. 8 2013 von rn-wissen: http://www.rn-wissen.de/index.php/Getriebemotoren\_Ansteuerung

Bosch Rexroth AG. (3 2012). Sicher durch Wellen Sekundärgeregelte Antriebe Kompensieren den Seegang. *drive&control*, S. 2-3.

Chacon, S. (2009). *Pro Git (Expert's Voice in Software Development).* http://git-scm.com/book.

Lehmann, H. L. (2008). *Taschenbuch der Elektrotechnik und Elektronik.* Münchem.

Merz, L. J. (1990). *Grundkurs der Reglungstechnik.* Oldenburg.

Papula, L. (2009). *Mathematiche Formelsammlung für Ingeniere und Naturwissenschaftler.*

Rabe, G. L. (kein Datum). *github*. Abgerufen am 28. 08 2013 von https://github.com/lesky/Weather-Station/