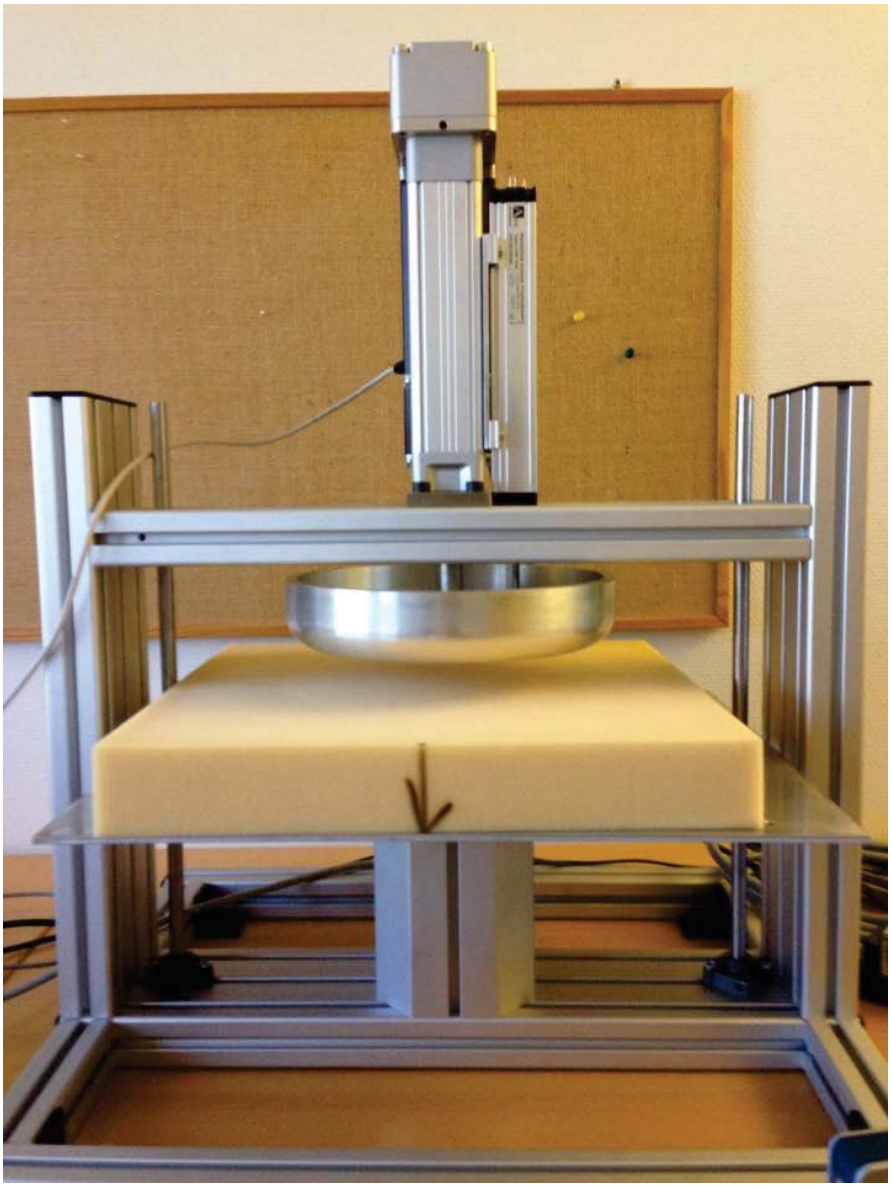
CHALMERS



Styrsystem för utmattningsrigg

Examensarbete inom högskoleingenjörsprogrammet Mekatronik

CHRISTOFFER ANDERSSON
FREDRIK GABRIELSSON

Institutionen för Signaler och system
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2015

Förord

Vi har utfört vårt examensarbete på Swerea IVF, vilket är slutmomentet på Mekatronikprogrammet (180 hp) på Chalmers tekniska högskola.
Den här rapporten omfattar 15 högskolepoäng och genomfördes under våren 2015. Vi vill tacka Alexander Mann på Swerea IVF som har hjälpt oss under resans gång.
Vi vill även passa på att tacka Manne Stenberg på Chalmers tekniska högskola som har guidat oss och lämnat feedback till den här rapporten.

Göteborg den 17 juni 2015.

Christoffer Andersson Fredrik Gabrielsson

Sammanfattning

I den här rapporten återfinns en beskrivning på hur ett styrsystem för en utmattningsrigg har konstruerats. Arbetet gjordes på begäran av Swerea IVF, då ett effektivare och mer lönsamt sätt att testa cellplast behövdes. I dagsläget har företaget fyra testriggar, tre av dessa styrs hydrauliskt och kräver mycket underhåll, vilket leder till onödiga driftstopp. Den fjärde testriggen har ett välfungerade styrsystem. Tanken med projektet är att efterlikna testriggen med det fungerade styrsystemet, men på ett mer kostnadseffektivt sätt.
Den här rapporten behandlar områdena hårdvara och mjukvara. Tanken är att få en klarare bild över hur hårdvaran kommunicerar med varandra, för att på så sätt kunna implementera ett styrsystem på bästa sätt. Den befintliga hårdvaran består av en DC-motor, ett ställdon, lastcell, mätutrustning samt ett CAN-USB. Mjukvaran som används är National Intstruments grafiska programspråk LabVIEW. Kommunikationen mellan motorn och datorn sker via CAN-USB, där motorn använder protokollet CANopen. Till sist beskrivs hur programkoden är uppbyggd och motivering av vilka index samt parametrar som skickas till motorn, för att denna ska kunna utföra det som efterfrågas.

Abstract

This report describes the development of how a control system for a test rig has been constructed. The basis of the work is Swerea IVF's wish to improve a more efficient and profitable way to test foam. In the current situation the company has four test rigs, three of which are controlled hydraulically and requires maintenance. The fourth test rig has a well-functioning control system. The idea of the project is to match the test rig with the well-functioning control system, but to a more affordable cost.
This report covers the areas of hardware and software. The idea is to get a clearer picture of how the hardware communicates with each other.
The existing hardware consists of a DC motor, an actuator, load cell, measuring equipment and a CAN-USB. The software used is National Intstruments LabVIEW graphical programming language. Communication between the engine and the computer takes place via the CAN-USB, where the engine uses the protocol CANopen. Finally the report describes how the application code is structured and an explanation of the index and the parameters passed to the engine is described.

Innehållsförteckning

1. Inledning.

Studenter på Chalmers tekniska högskola, som läser kursen Examensarbete har fått i uppgift att lösa ett problem som består av att konstruera ett styrsystem till en utmattningsrigg.

1.1 Bakgrund.

Swerea AB är en koncern som riktar in sig på forskning inom områdena material-, process-, produkt- och produktionsteknik. Koncernen bildades 2005 och en av ägarna är den svenska staten.
Ett av dotterbolagen, Swerea IVF, där examensarbetet utförs, behandlar industriell produktframtagning, process- och materialutveckling. Bakgrunden till detta examensarbete ligger i att Swerea IVF vill effektivsera och reducera kostnader för testning av cellplast.
I nuläget används testriggar som drivs hydrauliskt. Dessa kräver mycket underhåll och medför att driftstopp förekommer. För att undvika detta ska ett styrsystem konstrueras som sedermera kan implementeras på fler, mer effektiva testriggar. De nya testriggarna byggs ihop på företaget vilket reducerar kostnaden, dock behövs ett styrsystem för att testningen ska kunna ske.

1.2 Syfte.

Uppdraget går ut på att ta fram ett styrsystem som ska kunna utföra utmattningsprover på cellplast. Programkoden ska skrivas i LabVIEW, vilket är Swereas utvecklingsverktyg för testning samt mätning. Proverna säljs sedan till kund.

1.3 Avgränsningar.

Studenterna har tillgång till de material och apparater som har tilldelats av företaget. Styrsystemet ska även skrivas i programmet LabVIEW, då det är detta program som används på företaget. Cellplastprovet utsätts för utmattning enligt standard SS-EN ISO 3385 samt SS-EN ISO 2439. Detta medför att studenterna måste följa de anvisningar som finns i dessa dokument.
Projektet är tidsbestämt mellan den 23 mars till 5 juni 2015.

1.4 Precisering av frågeställning.

Det viktiga i projektet är att komma fram till vilket som är det mest lämpliga sättet att styra riggen på. För att lyckas med detta behövs kunskap om hur nuvarande, liknade styrsystem på marknaden fungerar. Hur ska delarna i utmattningsriggen kommunicera med varandra?

2. Teknisk bakgrund.

För att få en bättre översyn på projektet kommer nedan de olika komponenterna samt förkortningar som använts att beskrivas.

Utmattningsrigg.

Riggen består av ett linjärt ställdon som trycker på provbiten. En lastcell mäter kraften som det linjära ställdonet skapar. En datalogger styr indata samt mäter utdata i systemet.

Linjärt ställdon.

I detta projekt kommer ett elektriskt linjärt ställdon att användas. Ställdonets uppgift är att göra om den roterade rörelsen från motorn till en linjär motsvarighet. Detta kommer sedan generera en tryckande rörelse mot cellplasten.

Cellplast.

Är en form av plast som finns i både mjukt och hårt utförande. Den cellplast som används i detta projekt är av den mjukare formen och det är även den som är mest vanlig i möbler. (Wikipedia. Cellplast, 2015)

LabVIEW.

Laboratory Virtual Instrumentation Engineering Workbench är ett grafiskt programmeringsprogram som underlättar programmering vid hårdvara utveckling. I LabVIEW används språket ”G”. Programmet ska vara lätt och snabbt att lära sig och lämpar sig bäst vid hårdvara som ska utföra testning, mätning samt analys av data.
(National Instruments, 2015)

NI 9234.

Är en datainsamlingsmodul från företaget National Instruments, som används för noggranna mätningar av ljud, vibration och högfrekventa applikationer. Med NI 9234 tillkommer användningsbara LabVIEW funktioner som kan anpassa och läsa av och spara data. Modulen själv kräver ingen programmering. I detta projekt gör modulen om analoga mätningar till digitala. (National Instruments, 2015)

NI 9171.

Är ett 1-slots USB chassi från företaget National Instruments. I chassiet placeras ovanstående modul. NI 9171 används för att få in de digitala signalerna, från modulen, till datorn via USB. (National Instruments, 2015)

NI 8472.

Är en 1-port CAN-modul som arbetar i low-speed d.v.s. med en maximal hastighet med baudrate 125000 bit/s och minimal hastighet på 5000 bit/s. NI 8472 är av fabrikatet National

Instruments. Modulen används vid anslutning mellan datorer och CAN nätverk. Anslutningen till datorn görs via USB där modulen får sin matning.
(National Instruments, 2015)

ISO-standard.

Internationella standardiseringsorganisationen är en internationell standardiseringsorganisation som verkar för standard inom industri samt kommersiella områden. I projektet följs standarderna SS-EN ISO 2493 och SS-EN ISO 3385. Standarderna beskriver hur cellplastprovet utsätts för utmattning samt en flödesplan för hur riggen skall operera. (Wikipedia. ISO standard, 2015)

CAN.

Controller Area Network (CAN) är ett kommunikationssystem som används mellan styrsystem och datorer. CAN har utvecklats för att operera vid trånga och svåråtkomliga utrymmen, som i fordonsmotorer och dylikt, då användningen av kablar och komponenter kan minskas. Systemet är byggt på noder som är parallellkopplade till ledare. Mellan noderna sänds meddelanden som är serier med bit-signaler som vanligast innehåller 11 bitar. Bit-signalerna innehåller en identifierare, som visar vilken funktion som skall utföras, samt data om funktionen. Identifieraren visar på om en nod skall behandla meddelandet samt vilket meddelande som skall prioriteras då flera meddelanden skickas samtidigt. CAN kan operera i hög hastighet och hårdvaran kan omprogrammeras så att ett system är enkelt att utveckla och förbättras. (Wikipedia. Controller Area Network, 2015)

CANopen.

Protokollet CANopen är baserat på CAN kommunikation och används främst vid kommunikation av öppna nätverk. CANopen finns i stor utsträckning inom industrin vid styrning av t.ex. ställdon och regulatorer.
(Wikipedia. CANopen, 2015)

SDO.

Service data object ”SDO” är en typ av kommunikation inom CANopen. SDO används för att få direkt åtkomst av parametrar och ID från en CANopen enhet. Det går även att skicka meddelanden och kommandon via SDO.
(CAN in Automation, SDO, 2015)

PDO.

Process Data Object “PDO”, är den andra typen av kommunikation som används inom CANopen. Protokollet används främst då meddelanden ska skickas till fler nod id.
(CAN in Automation, PDO, 2015)

Nod id.

Vid kommunikation via CANopen används nod id för att visa vilken adress meddelandena skall skickas. Varje nod, eller mottagare har ett specifikt nod id. Det finns nod id mellan 1-127.

Baudrate.

Vid överföring av signaler används ofta enheten baudrate för att visa hur snabbt signalerna överförs. Enheten är signalpuls per sekund.

DAQ .

Data acquisition är en process som mäter verkliga analoga signaler och omvandlar dessa till digitala värden. Datainsamlingen konverteras och bearbetas av en dator för att få önskad enhet och anpassning till de digitala värdena. (Wikipedia. DAQ, 2015)

While loop.

Är ett vanligt förekommande kommando inom programmering. Loopen körs i oändlighet tills att ett bestämt villkor uppfylls. (Wikipedia. While loop, 2015)

True/false struktur.

Är ett programmeringskommando vars funktion är att returnera en slutsats till ett villkor. Om villkoret uppfylls returnerar strukturen true annars false. True och false representeras av 1 och 0. (Wikipedia. True/false struktur, 2015)

Array.

En array kan beskrivas som ett fält eller en matris med ett eller flera element. Elementen består av värden av samma datatyp. Elementen har index som visar i vilken ordning det befinner sig inom arrayen. Element med lågt index har högre representation än ett element med högt index. (Wikipedia. Array, 2015)

Hexadecimala värden.

Hexadecimala värden är värden med talbasen 16 som ofta används in programmering. Värdena är positionsbaserade på sexton siffror 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E och F där A motsvara värdet 10 decimalt, B motsvarar 11 O.S.V. Talen kan räknas om till binära där F, med värdet 15 decimalt, går att läsa som 1111. Här befinner sig det lägsta vädret, 1 decimalt, till höger och för varje steg till vänster fördubblas värdet. Värdena adderas sedan ihop enligt följande 8+4+2+1 = 15 vilket motsvarar F hexadecimalt. (Wikipedia. Hexidecimala värden, 2015)

3. Metod.

I arbetet mot den automatiserade utmattningsriggen kommer många deluppgifter att genomföras. Projektet kommer att utföras på ett strukturerat och metodiskt sätt för att eliminera delproblem, samt att förståelse för de olika stegen som utförs erhålls.
Första steget i processen blir att undersöka hur liknande system fungerar. Därefter kommer hårdvaran att granskas för att få en bättre bild över projektet. Informationsinhämtning om de mjukvaruprogram som ska användas ska göras för att på rätt sätt kunna använda de verktyg som krävs för att implementera ett väl fungerade styrsystem.

Eftersom projektet är tidsbegränsat kommer en tidsplan att användas för att lättare få en överblick hur projektet fortskrider. Tidsplanen är av typen Gantt-schema där projektet delas upp i delmoment. I tidsplanen kommer det att finnas spelrum för uppkommande problem och förseningar.

Avstämningsmöten med handledare på Chalmers och projektledare med personal på företaget kommer att hållas veckovis för att upprätthålla en god kommunikation samt att undvika misstolkningar och komplikationer.

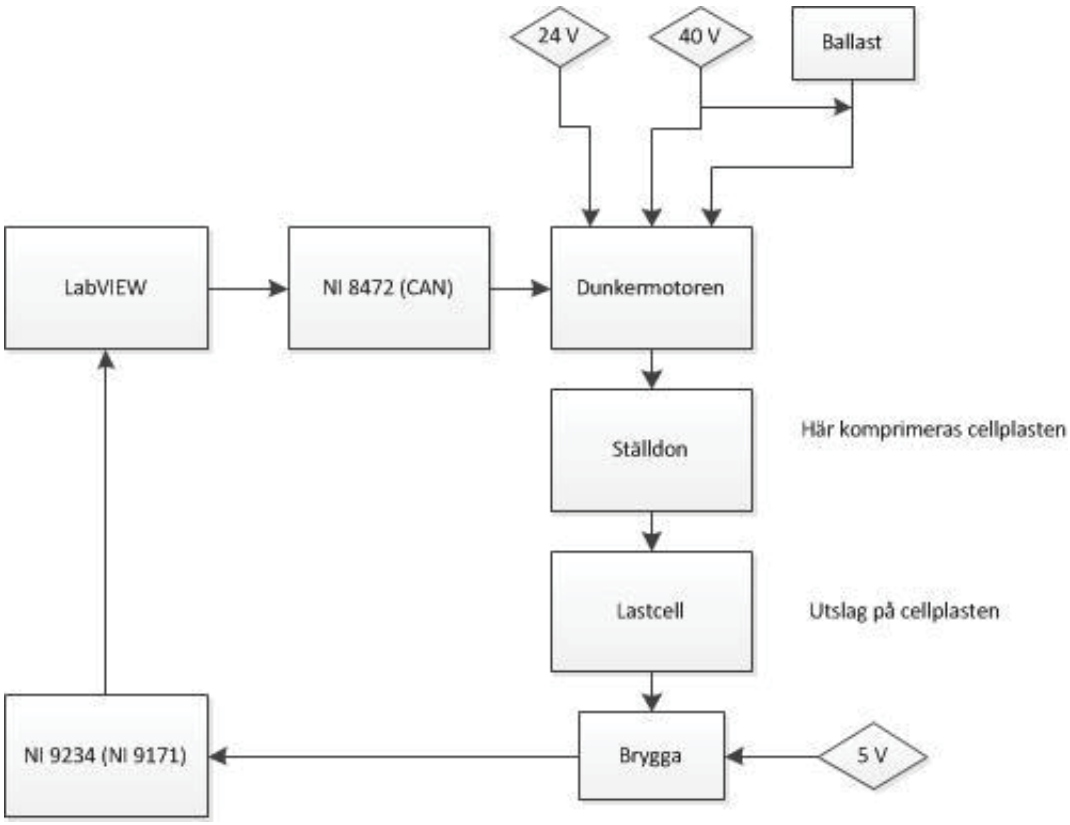
4. Kravspecifikation.

Följande punkter ska levereras/utföras under projektets gång:

- Det mest lämpliga sättet att styra riggen ska undersökas.
- Ett processchema av styrsystemet.
- Ett användargränssnitt med möjlighet att välja standard, användare och projektnummer.
- Ett styrsystem för att utföra utmattningsprover enligt standard. Under cyklingsdelen ska ställdonet röra sig sinusformat.
- Efter utfört utmattningsprov ska mjukvaran leverera en kundrapport med värden från testet. Rapporten ska även innehålla en graf från körningen, detta för att lättare kunna avgöra om något har gått fel.

5. Uppbyggnad av hårdvara.

För att kunna konstruera ett lämpligt styrsystem måste rätt komponenter kunna kommunicera med varandra. Nedan listas vilka komponenter som används och hur dessa kommunicerar.



Figur 2. Processchema för utmattningsrigg

5.1. Motor

Motorn är en DC-motor av fabrikatet Dunkermotoren BG75CI. Denna styr ställdonet som kommer att applicera kraft på cellplasten. Motorn matas med 40 V DC genom en fyra-pin kontakt. En pin är jordad medan två andra matas med spänning. En av dem matas via ett driftdon för inbromsning av donet. Motorns mjukvara behöver också matningsström för att kunna operera korrekt. Denna matning anpassas till 24 V genom ett nätaggregat. För att styra motorn är den kopplad till datorn via ett USB stick som skickar CAN meddelanden samt programkoder från LabVIEW.

5.2. Ställdon

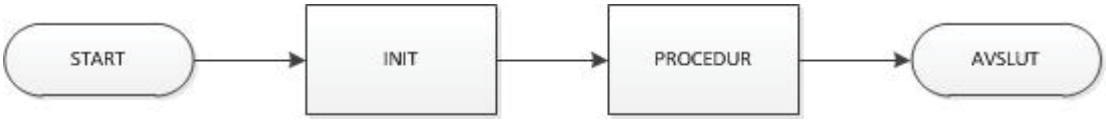
Ställdonet är monterat till motorn och styrs av denna. Ställdonet är programmerat till att röra sig i vertikalplanet i en sinusformad rörelse. Den resulterande kraften som utkommer från ställdonets rörelse appliceras på cellplasten.

5.3. Lastcell

Lastcellen mäter av ställdonets utslag på cellplasten. De analoga signalerna från lastcellen är kopplade till en brygga som matas med 5 V. Bryggan förstärker signalerna så att de gör utslag på ett mätbart sätt. Signalerna skickas vidare till en NI 9234 modul, som får ström från datorn genom ett USB stick, för att kunna tas in till datorn och läsas av i LabVIEW. Data som hämtas från lastcellen har från början enheten Volt, men görs om i LabVIEW till den önskade enheten Newton. Enhetsomvandlingen sker i funktionsblocket DAQ Assistant som senare kommer att förklaras i rapporten.

6. Processen i stegform.

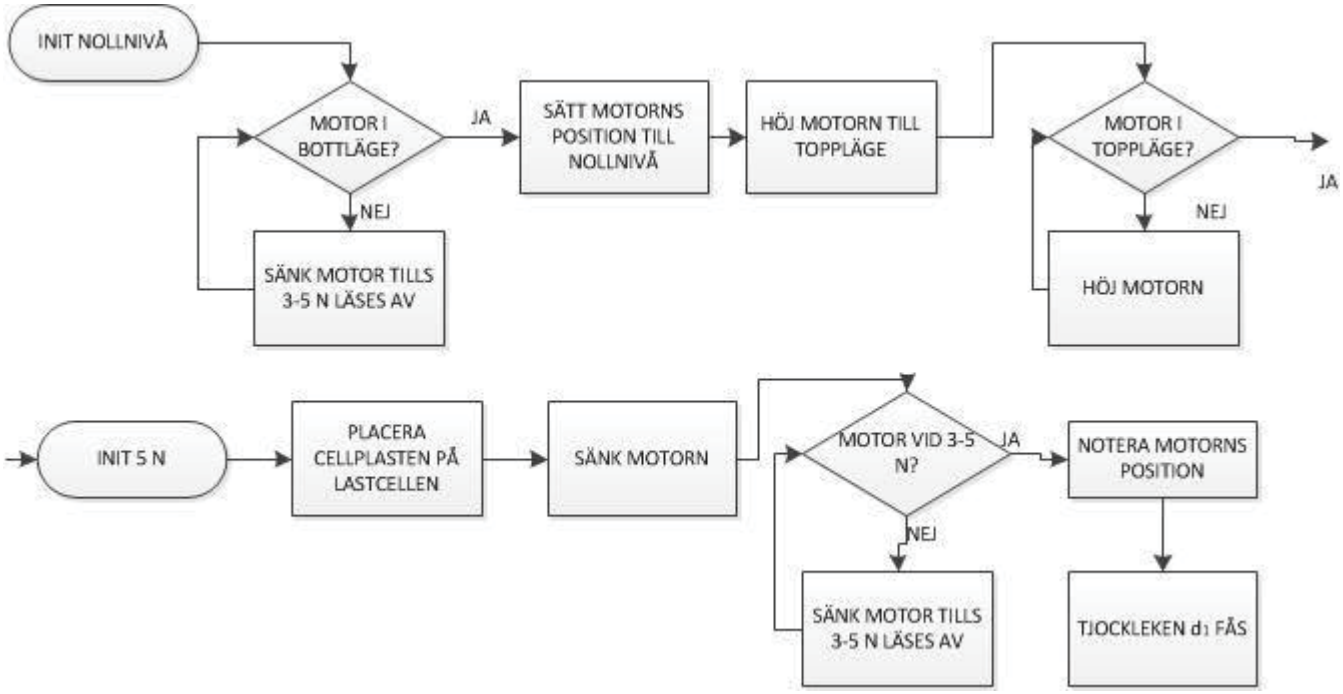
Processen kan delas upp i två delar. Den första delen består av initieringen som följs upp av proceduren.



Figur 3. Processen i flödesschema

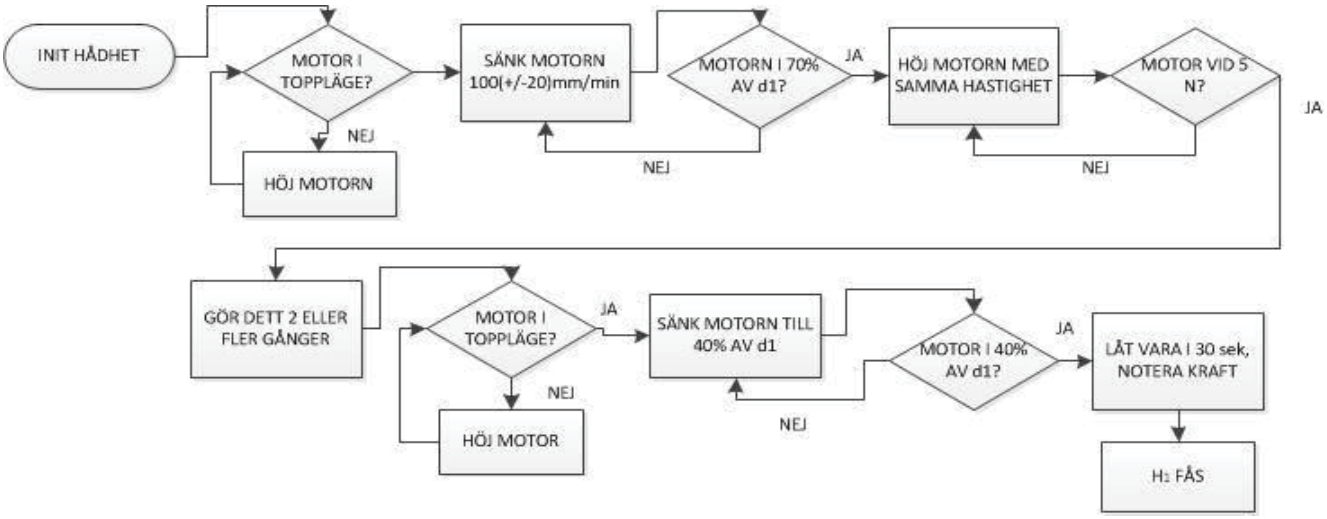
6.1 Initiering

I initieringen fås två parametrar, tjocklek och hårdhet. Dessa kommer användas vid beräkningar av det slutgiltiga testresultatet. Denna process följer en ISO-standard vilket medför att det inte finns mycket spelrum för hur initieringen kan utformas.



Figur 4. Flödesschema över initieringen av mätning på tjocklek d_1.

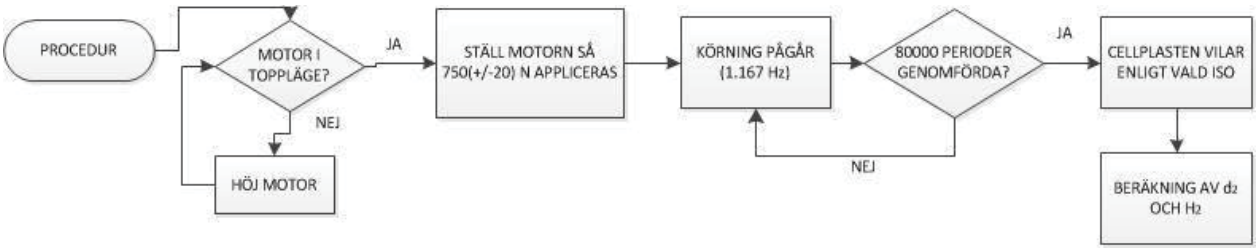
Steg ett i initieringen är att få fram den första parametern, tjockleken d_1. För att åstadkomma detta behövs en referenspunkt, nollnivån. Motorn ska applicera en kraft på 3-5 N för att nå detta läge, som är beläget vid bottenplattan. Där noteras motorns position och sätts till noll. Därefter placeras cellplasten på bottenplattan och motorn applicerar då samma kraft som behövdes för att nå nollnivån. Denna position som motorn då kommer befinna sig på, noteras och sparas som d_1.



Figur 5. Flödesschema över initieringen av mätning på hårdheten H_1.

Cellplatsen ska sedan komprimeras med en specifik hastighet tills dess att denna har komprimerats till 70 % (+/- 2.5 %) av tjockleken d_1. Ställdonet höjs sedan upp till positionen för d_1. För att få ett bra värde på tjockleken görs denna process om två till tre gånger.
För att räkna ut hårdheten, index H_1, komprimeras cellplatsen med 40 % (+/- 1 %) av tjockleken d_1. Ställdonet stannar i detta läge i ungefär 30 s, innan kraften, mätt i N noteras. Därefter åker ställdonet upp igen.

6.2. Procedur



Figur 6. Flödesschema över proceduren.

När väl initieringen är klar påbörjas testkörningen. Det första som sker är att man justerar slaglängden till att vara mellan positionen för 3-5 N samt 750 N (+/- 20 N) N. Körningen pågår i 80 000 perioder i en sinuskurva mellan 3-5 N och 750 N (+/- 20 N) N. Sinuskurvan justeras så att, Om cellplasten blir mjukare vid testkörningen, så skall den angivna kraften 750 N (+/- 20 N) alltid nås. Ställdonet skall arbeta med 70(+/- 5) slag/min.
Därefter får cellplasten vila, hur länge beror på ISO-standard, innan tjockleken d_2 mäts.
Hårdheten H_2 mäts nu genom att man tar 40 % (+/- 1 %) av den ursprungliga tjockleken d_1.

6.3 Uträkningar

Efter avslutad körning är det medelvärdena av tjockleken och hårdheten som är det intressanta. Dessa kommer automatiskt redovisas i en kundrapport med tillhörande graf från körningen. Nedan listas hur dessa räknas ut.

$$\Delta d = 100 * \frac{d_1 - d_2}{d_1}$$

$$\Delta H = H_1 - H_2$$

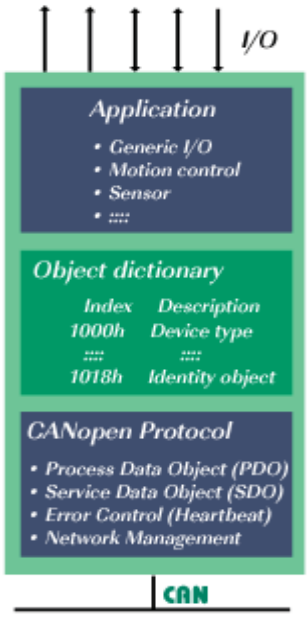
$$\text{Procentuell förlust i hårdhet} = 100 * \frac{H_1 - H_2}{H_1}$$

7. Utförande

Nedan återfinns beskrivning av skriven programkod och hur denna sänds, samt överförs till motorn.

Figur 7. Från LabVIEW till motorn

7.1. Kommunikation



Figur 8. Dataöverföring (CAN in Automation CANopen)

Mellan LabVIEW och motorn överförs data via ett CAN-USB (NI 8472). Detta möjliggör att meddelanden kan skickas ut på CAN-bussen för att sedan läsas av motorn. Det finns olika typer av CAN-kommunikation, men eftersom motorn kräver CANopen så används detta.
Uppbyggnaden av CANopen är definierat som en kommunikationsprotokollstack och enhetsprofil. Denna tillhandahåller adresseringsschema, kommunikationsprotokoll, objektlista samt ett applikationslager. .
Mellan två enheter sker kommunikationen från applikationslagret på den sändande enheten, ner till CAN-bussen vidare till applikationslagret på den andra enheten, som då är mottagare. Vid applikationslagret sker utbytet av kommunikations- och applikationsobjekt. Detta sker i form av index med tillhörande subindex, som tillhandahålls från objektlistan.

I objektlistan återfinns de objekt som adresseras med en 16-bitars index och en 8-bitars subindex. För att få åtkomst till de adresserade objekten finns det två möjligheter. Antingen används Service Data Objects, SDO eller den mer avancerade Process Data Objects, PDO.

För att kunna kommunicera mellan två enheter behöver dessa veta till vilken adress meddelandena ska skickas. När meddelandet skickas så har det specifika meddelandet en identifierare. Denna identifierare brukar kallas för COB-ID inom CANopen och består av 11-bitar. De fyra mest signifikanta bitarna är funktionskoden. Det den berättar är vad det är för typ av meddelande. De återstående sju bitarna är nod id.

För att lösa den här kommunikationstypen krävdes att motorn programmerades om från

parameterstyrning till att baseras på CANopen. Detta gjordes via en uppdateringsfil som konverterade motorn. Denna fil tillhandahölls från motortillverkaren.

7.2. Val av protokoll – SDO/PDO

SDO
CANopen protokollet SDO används då det finns en eller flera mottagare i nätverket. Då datorn skickar ett meddelande är det upp till enheterna om de ska ta emot meddelandet. Ett meddelande skickas till motorn som i sin tur bekräftar att meddelandet har mottagits genom att skicka tillbaka det till datorn. Vid varje nytt kommando som motorn får skickas det ett svar med en bekräftelse.

PDO
Protokollet PDO baseras på att ett meddelande skickas till motorns nod ID. Mottagaren skickar inte tillbaka det mottagna meddelandet utan datorn får själv efterfråga status. När meddelandet har skickats till motorns nod ID kan förändringar, såsom ut- och ingångar, göras av nodstyrenheten. Ett meddelande kan alltså ändras då det har mottagits av motorn. Ett meddelande skickas genom TPDO, Transmit PDO, och status läses av genom RPDO, Reecive PDO.

I projektet är det enbart en enhet som skall styras. Med detta finns det bara en mottagare i nätverket och alla meddelanden skickas till samma nod. Konfigueringen av objektlistan är mycket lättare vid användning av SDO, vilket i detta projekt är en väsentlig del.
Detta gör att användning av SDO är ett enkelt och smidigt tillvägagångssätt. Ett annat argument för användning av SDO är att en kontroll av att meddelandena har mottagits utförs automatiskt. Detta gör att felsökning av programkod samt simulering av ”hardware in the loop” blir betydligt enklare att genomföra. Huvudargumentet är dock att de index som ska användas i motorn är av typen, läs av, skriv, läs av, vilket är precis vad SDO tillhandahåller. I projektet används exempelvis ett index som öppnar nödbromsen på motorn. För att enkelt kontrollera att så är fallet, läses indexet av, om den inte är öppen, skrivs ett värde som gör att den öppnas och slutligen kontrolleras att det genomfördes.

7.3 DAQ

För att kunna mäta vilken kraft ställdonet applicerar på cellplasten används DAQ, Data Aqusition. I LabVIEW finns ett block, DAQ Assistant, där man kan ställa in hur man önskar att mäta data. I projektet används 1000 samplingar med frekvensen 10 000 Hz. Ett nytt värde mäts varje gång DAQ Assistant körs och i projektet ligger DAQ inuti en while loop, vilket medför att nya mätvärden fås kontinuerligt.
Detta verktyg kan med hjälp av en skalning räkna ut vilken kraft, mätt i Newton, som anläggs på bottenplattan. För att få fram exakt vad 1 N motsvarar gjordes beräkningar, vilket i slutändan resulterade i en ekvation av typen, $y = kx + m$.
I programkoden används DAQ Assistant när positionen för 5 N respektive 750 N ska mätas.

7.4. Index

Motorns alla funktioner och styrningssätt motsvaras av olika index samt tillhörande subindex. Till dessa skickas meddelanden via SDO, som beskrevs i föregående kapitel. Då kommunikation görs via CAN går det enbart att skicka ett meddelande åt gången till respektive index. När ett nytt kommando skickas skriver detta över föregående kommando som skickats till det valda indexen. För att motorn skall realisera önskat utförande skickas det med ett meddelande i form av ett värde i en array till valt index. Värdena som skickas till motorn består av antingen hexadecimala eller decimala värden med representationen int16. Anledningen till att representationen valdes till en signed integer i stället för unsigned, var att motorn krävde negativa värden till några av indexen. De index, subindex och värden som kommer att styra motorn i projektet har hämtats ur en parameterlista från motortillverkaren.

Då styrningen av motorn önskas vara positionsbaserat används index därefter. Nedan visas och beskrivs de index som används i projektet.

3150 00 - Open brake. Hit skickas värdet 1 för att öppnas motorns nödstopp. Nödstoppet aktiveras varje gång strömmen bryts och måste återställas för att köra motorn.

3000 01 - Clear error. Hit skickas värdet 1 för att återställa eventuella error.

3000 10 - Rpm. Ställer in hastigheten på motorn. Hit skickas önskat värde.

3000 11 - Count. Sätter vilken position motorn skall köra till. Hit skickas önskat värde.

3000 01 - Start absolute positioning. Aktiverar motorn till att köra till positionen som numera återfinns i Count. Hit skickas det decimala värdet 118 för att utföra sekvensen.

3762 00 - Actual Position. Här går det att både läsa av och skriva vilken aktuell position motorn befinner sig i.

3000 01 - Quick stop. Används för att stoppa motorn. Quick stop aktiveras när värdet 2 skickas till index. När detta inträffar kan inga kommandon påbörjas förrän index till Continue är aktiverat.

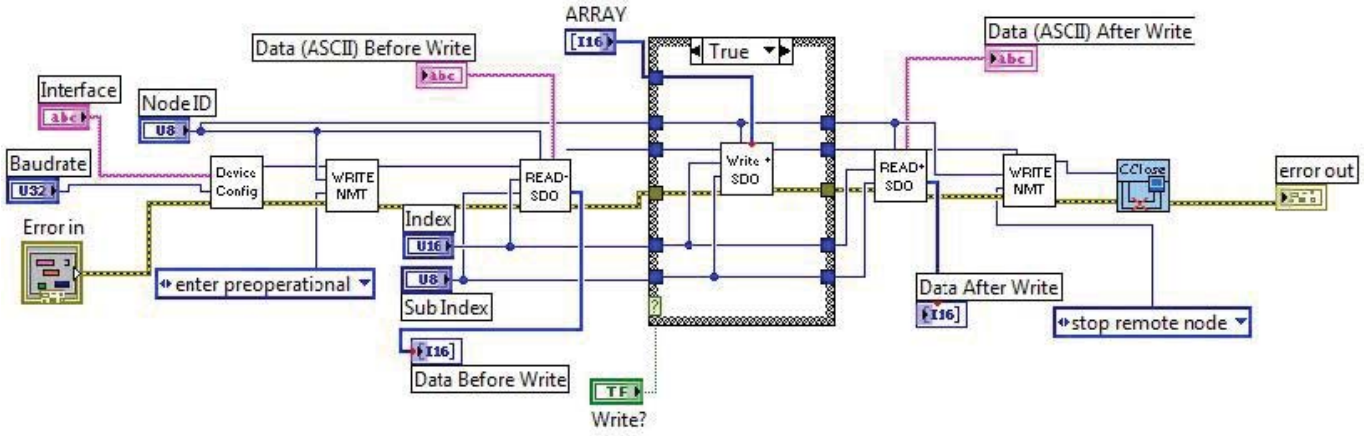
3000 01 - Continue. Detta index används efter Quick stop har aktiveras för att kunna utföra nya kommandon. Hit skickas värdet 4.

7.5. Programmering

I det här avsnittet kommer upplägget av programkod att beskrivas.

7.5.1 SDO – Read, write, read.

I nedanstående bild visas hur kommunikationen mellan datorn och motorn är uppbyggd med programkod. Med hjälp av den här programkoden, kan instruktioner med tillhörande värde skickas för att sedan läsas av och utföras av motorn.



Figur 9. Överblick SDO – read, write, read i LabVIEW

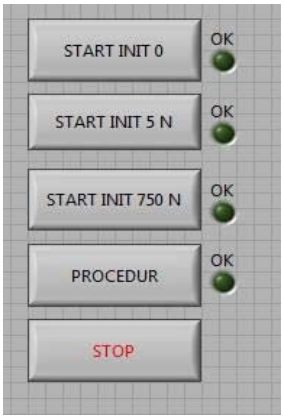
Det första som sker för att upprätthålla kommunikationen är att konfigurera enheten. Detta medför att datorn vet till vilken enhet den ska skriva till. För att åstadkomma detta väljs ett interface samt vilken baudrate enheten ska operera i. Den här informationen återfinns hos motortillverkaren.
När initieringen av enheten är klar sker en skrivning till NMT (Network Management). De signaler som det blocket tar emot är enhetens konfigurering, nod id och vilket av de fyra lägena som NMT tar emot. Dessa fyra lägen är initiering, före drift, drift samt stopp. Eftersom initieringen är klar, är nästa steg för NMT, före drift (preoperational), vilket säger till CANopen nätverket att enheten är redo att användas.

Figur 10. Avläsning samt skrivning

Nästa steg är en avläsning. Först väljs vilket index med tillhörande subindex som ska läsas av och sedermera skrivas till. Anledningen till att programmet gör en avläsning innan skrivningen är för att kontrollera att rätt värde ligger på bussen, då det i vissa fall sker överskrivning. I figuren ovan kan man se vilket värde som finns på den adress där index ”3000” och subindex ”10” befinner sig. I det här fallet ligger värdet 50 där.
Vidare i programslingan sker skrivningen. Detta block ligger inuti en true/false struktur, vilket innebär att om knappen ”Write?” är aktiverad skickas en etta och skrivningen utförs.
För att skicka värden till det index som valts används en array. I ovanstående bild kommer värdet 10 att skickas till index ”3000” när knappen ”Write?” aktiveras. Detta värde kommer sedan att dyka upp på ”Data after write” då en avläsning sker efter det att skrivningen utförts. Detta för att säkerhetsställa att värdet verkligen skrevs till det valda indexet.

För att avsluta programslingan behöver NMT veta att allt är gjort på CANopen nätverket. Läget stopp väljs och signalen skickas till ett block som stänger ner nätverksobjektet som i början konfigurerades. Avslutningsvis går en error-signal genom programmet.

7.5.2 Main



Figuren till vänster illustrerar uppbyggnaden av programmet i form av ett användargränssnitt av programmet Main. I Main finns det ett antal delprogram där det skickas meddelanden till motorn. Delprogrammen är baserade på SDO – read, write, read och de index som beskrivs i tidigare kapitel. För att programmet skall veta vilken ordning av utförandet dras en error-signal mellan delprogrammen. Genom att aktivera de olika kontrollerna, som visas i figuren, skickas det meddelanden till motorn som utför dessa programslingor. Då delprogrammen har utförts tänds en grön indikator och nästa delprogram kan köras. För att avsluta körningen aktiveras ”STOP”.

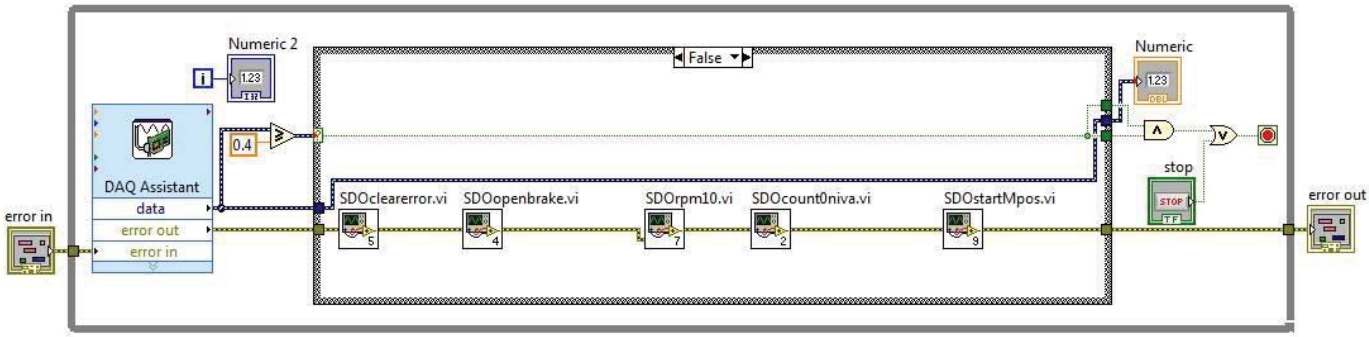
Figur 11. Main

START INIT 0

Det första som skall göras är att sätta nollnivån. Den är vald att befinna sig i ställdonets bottenläge. Detta då motorn annars hade räknat negativa positioner när ställdonet är på väg nedåt. I och med att nollnivån sätts nere vid bottenplattan undviks detta och endast positiva positioner kommer därefter att finnas. Genom att aktivera kontrollen ”START INIT 0” skickas värdet ett och programmet går in i en true/false struktur med följande delprogram.

Figur 12. Initieringen av nollnivån

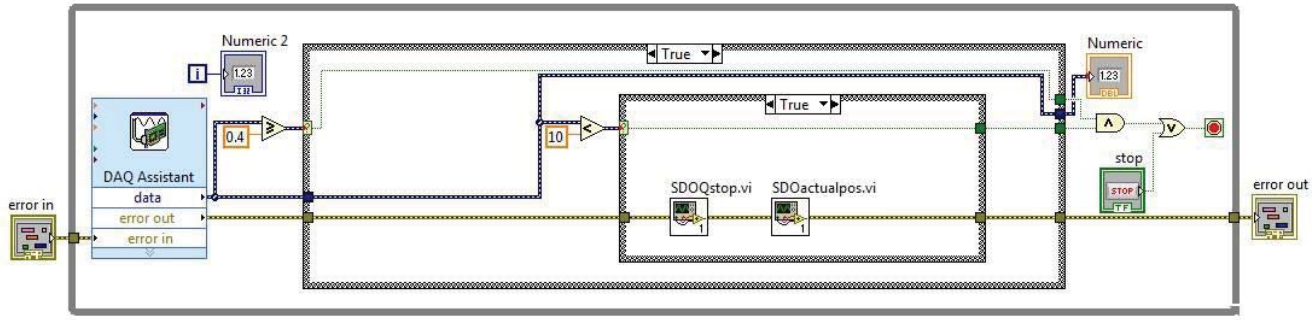
I subMainInit0 går programmet in i en while loop. While loopen körs tills att antingen ”STOP” aktiveras eller att nollnivån är satt korrekt. Inne i loopen jämförs värden, som skickas från DAQ Assistant, med 0.4 N. Om värdet är mindre än 0.4 N så skickas det noll till en true/false struktur. I strukturen utförs då delprogram som nollställer error samt nödstopp, sätter motorns hastighet, ställer in vilken position donet skall köra till, samt startar körning av motorn. Hastigheten sätts till det låga värdet 10 rpm för att få en precis avläsning från DAQ Assistant. Positionsvärdet ställs in till ett negativt tal för att med säkerhet få donet att köras nedåt mot bottenplattan. I och med att programmet körs i en while loop får DAQ Assistant in nya värden upprepande och dessa jämförs mot 0.4 N.



Figur 13. subMainInit0

Då värdet från DAQ Assistant når över 0.4 N går programmet in i thrue/false strukturen som true. Där startas en ny jämförelse mot värdet från DAQ Assistant. Denna gång är värdet som skall jämföras 10 N. Intervallet 0.4 N - 10 N är satt då det önskas att nollnivån sätts då donet precis vidrör bottenplattan med minimal kraft. Den nya jämförelsen är i sin tur kopplad till en ny thrue/false struktur. Om värdet från DAQ Assistant överstiger 10 N skickas värdet noll till thrue/false strukturen. Inne i strukturen sätts motorn då till att sakta köras uppåt. Programmet börjar därefter om med att jämföra värdet från DAQ Assistant med 0.4 N. Donet kan behöva jobba upp och ner några sekunder för att hamna inom intervallet. Då detta realiseras stoppas

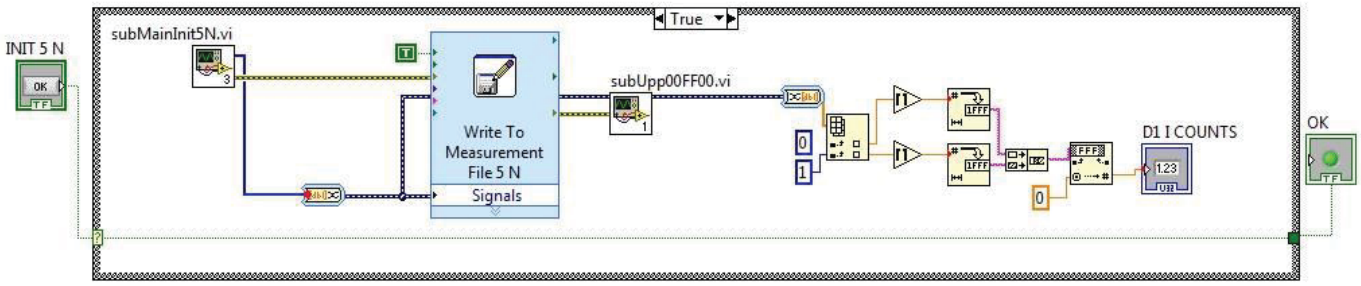
motorn och donets position sätts till noll genom delprogrammet SDOactualpos.
Därefter är programslingan slut och while loopen stängs då ner automatiskt eftersom en etta skickas genom båda strukturerna, med slutdestination stop. Därpå går programmet vidare till nästa delprogram genom errorsignalen. Detta delprogram kör upp donet till positionen 00FF, som befinner sig i motorns toppläge utifrån nollnivån. Delprogram innehåller också funktionen continue. Den måste genomföras för att nya meddelanden skall kunna skickas till motorn, då motorn har fått ett Quickstop meddelande i föregående delprogram. Nästa delprogram ur huvudprogrammet kan därefter startas.



Figur 14. subMainInit0

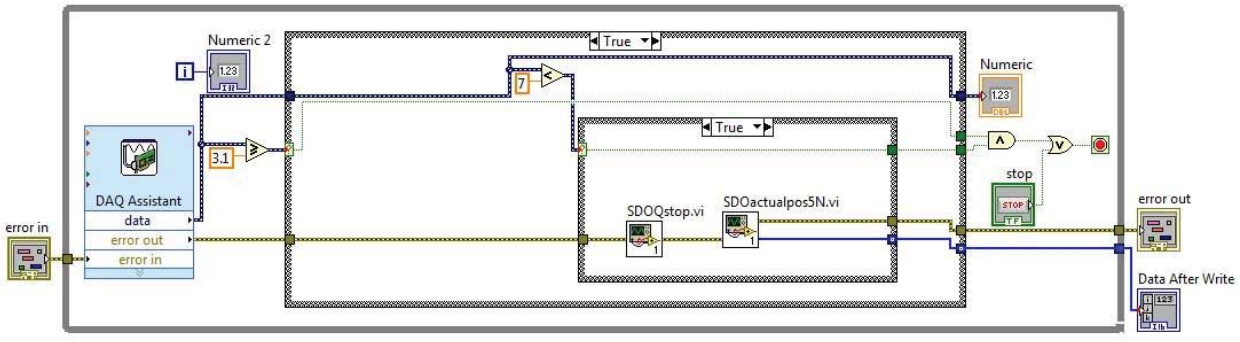
START INIT 5 N

Då cellplasten är applicerad på bottenplattan kan delprogrammet ”START INIT 5 N” startas. Detta delprogram är uppbyggt på ett liknade sätt som INIT 0. Det är till för att dels mäta tjockleken d1 samt även få en position som proceduren sedermera kommer ha som max position för sinuskurvan. Först går programmet in i en thrue/false struktur där kontrollen ”START INIT 5 N” aktiveras. Där går programmet vidare in i delprogrammet subMainInit5N.



Figur 15. Initiering av 5 N

Här hämtas ett värde från DAQ Assistant och jämförs om det är större, eller lika med värdet 3 N. I kravspecifikationen har beställarna ett intervall på 5 (+/- 2) N. I början av körningen appliceras ingen kraft på cellplasten, vilket medför att villkoret med 3 N inte uppfylls. Programmet är utformat precis som subMainInit0 och donet börjar sakta köras nedåt mot cellplasten. När donet når cellplasten och kraften överstiger 3 N uppfylls det första kravet och värdet ett skickas till thrue/false strukturen. Där väntar en ny jämförelse för att undersöka om kraften är mindre än 7 N. Om detta inte inträffar höjs donet och programmet börjar om med jämförelserna. Innan kraften från donet hamnar inom rätt intervall kan det behöva ställas in genom att köras upp och ned i ett fåtal sekunder. När donet är inom rätt intervall aktiveras delprogrammet SDOstop där motorn stannar helt. Därefter läses donets position av och programmet går vidare till nästa delprogram.



Figur 16. subMainInit5N

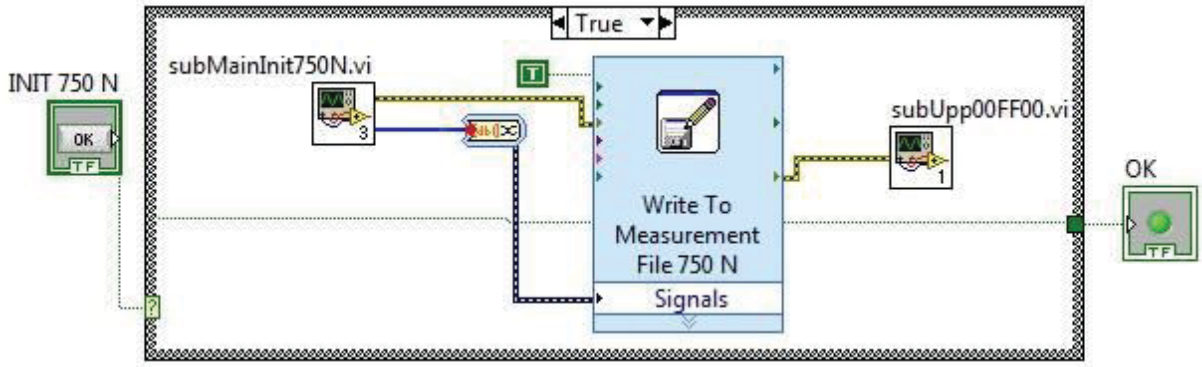
Positionsvärdet som läses av sparas i en textfil för att i senare program användas som en variabel. Värdet skrivs också ut som cellplastens tjocklek d1. Då värdet läses som decimala

tal i en array måste det anpassas för att skrivas ut som ett numeriskt värde. Detta görs genom att göra om talen till hexadecimala värden. Efter det delas elementen, i arrayen, upp för att sedermera adderas tillsammans och göras om till ett decimalt tal.

Donet skickas därefter upp till dess toppläge genom delprogrammet subUpp00ff00.

START INIT 750 N

Den andra positionen, som procedurens sinuskurva skall nå, är det värde som kommer ur delprogrammet INIT 750 N. Positionen som fås ur detta delprogram är positionen, då donet applicerar 750 N mot cellplasten. Delprogrammet, som figuren nedan illustrerar, är näst intill identiskt med delprogrammet INIT 5 N.



Figur 17. Initiering av 750 N

Skillnaderna är att intervallet för INIT 750 N är satt till 730 N- 770 N, samt att positionen inte skrivs ut, utan enbart sparas i en textfil.

PROCEDUR

Ett av de viktigaste kraven från Swerea IVF var att ställdonet, vid testkörningen skulle röra sig sinusformat med en bestämd frekvens. För att åstadkomma detta, behövs värdena från initieringen av 5 N och 750 N. Det första värdet, som hämtas vid initieringen på 5 N, används som maxposition i den tilltänkta sinuskurvan och positionen från 750 N används som minimumposition.
Dessa båda värden hämtas från den textfil dit de tidigare placerats. Värden ligger på en array. För att få ställdonet att röra sig sinusformat behöver en amplitud och en offset sättas. Offseten används då nollnivån, som ligger belägen vid bottenplattan, gör att sinuskurvan inte kan röra sig under nollnivån.

Amplituden beräknas enligt att positionen från 750 N subtraheras från positionen från 5 N. Exempel: [179, 45, 0, 0] – [102, 25, 0, 0] = [77, 20, 0, 0]. Den arrayen divideras sedan med två, vilket medför att utseendet blir [38.5, 10, 0, 0]. Den här arrayen delas sedan upp till två element [38,5] och [10] för att sedan avrundas uppåt. Anledningen till att en kontrollerad avrundning sker är främst en säkerhetsåtgärd, då det i vissa fall avrundades nedåt, vilket medför problem.
Därefter konverteras de två elementen till en hexadecimal sträng som sedermera görs om till ett numeriskt värde. Allt detta jobb är till följd av att LabVIEW inte tillåter att innehållet på en array, direkt kan omvandlas till ett numeriskt värde. Det numeriska värdet motsvarar amplituden.
Hela processen för offseten är likadan, bortsett från att positionsvärdena från 5 N och 750 N adderas för att sedan divideras med två.

Amplituden och offseten implementeras sedan i ett block ”Simulate signal”, som simulerar en sinuskurva utefter dessa värden. I det blocket ställs även frekvens, fasförskjutning, antal samplingar samt samplingsintervall in. Frekvensen ställs till 1.16 Hz då detta är ett krav från beställaren. Signalen som kommer från blocket skall sedan användas som positionsvariabler för motorn. Först måste signalen anpassas och göras om till en array med hexadecimala tal. Detta görs genom att göra om signalen till en hexadecimal sträng som sedan delas upp till element med två tal i varje. Dessa element sätts i en array. Då meddelanden som skickas till motorn måste bestå av en array med fyra element tillsätts värdet noll till de minst signifikanta bitarna tills att fyra element finns. Denna array skickas sedan till SDOsinuspos som sätter vilken position motorn skall köra till. Efter det startas motorns körning med SDOstartMpos. Denna del av proceduren finns i en while loop. Det medför att nya positioner, i form av en sinuskurva, skickas till motorn kontinuerligt. While loopen körs tills att antingen stoppknappen aktiveras manuellt eller att motorn har kört 80000 perioder av sinuskurvan. (Se bilaga för programkod)

8. Resultat

Målsättningen med examensarbetet var att ta fram ett fungerade styrsystem, som enligt ISO-standard kan utföra utmattningstester på cellplast.

För att åstadkomma detta på ett lämpligt sätt har ny hårdvara köpts in, då styrsystemet skulle vara positionsbaserat istället för att vara hastighetsbaserat. Detta underlättade programmeringen då man med hjälp av SDO på ett enkelt sätt kan kontrollera vid vilken position ställdonet befinner sig i.
Uppbyggnaden av programkoden är enkel och upprepar sig, vilket medför att vidare utveckling av programmet kan fortgå.

Vidare när det gäller kraven från beställaren har ett processchema konstruerats samt att ställdonet under testningen rör sig sinusformat. De punkter som inte blev uppfyllda var ett användargränssnitt med möjlighet att välja standard, användare och projektnummer samt att efter utfört utmattningsprov leverera en kundrapport.
Dock saknas en programslinga, nämligen den för att mäta hårdheten och hur denna ändras efter körningen. Denna behövs för att kunna påbörja testningen och sälja testresultatet till kund.

Test och verifiering
Under projekts gång har tester utförts kontinuerligt då ett nytt delprogram konstruerats. Detta medförde att direkta korrigeringar kunde göras för att på så sätt undvika större problem. Initieringen av programmet består av tre delar, nollnivå, hämta in positionen för 5 N samt 750 N. Dessa delprogram testades först var för sig, vilket resulterade i bra resultat.

Sluttestet bestod av att köra hela programmet. Först ut var att sätta nollnivån vid bottenplattan. När den var satt, åkte ställdonet till toppläget i väntan på nya direktiv. Samtidigt lades cellplasten på bottenplattan och kontrollen ”START INIT 0” kan då aktiveras. När kontrollen aktiveras åker ställdonet med en bestämd hastighet ned till där kraften 5 N appliceras av ställdonet, sedan återgår den till toppläget i väntan på nya direktiv.
Liknade process utförs då kontrollen ”START INIT 750 N” aktiveras, med skillnaden att ställdonet åker ned till där kraften 750 N appliceras för att sedan återgå till toppläge.

Till sist när testningen skall börja aktiveras kontrollen ”PROCEDUR”. Då startar programslingan med 80 000 cykler. Resultatet blev bra, då ställdonet åker mellan de två positionerna som anger var krafterna 5 N respektive 750 N återfinns, i en sinuskurva.

Verifieringen av programmet fungerar som önskat, vilket innebär att ställdonet rör sig enligt ISO-standard och enligt de krav som kommer från beställaren.

9. Diskussion

Arbetet med projektet har inte fortskridit smärtfritt utan oberäknade svårigheter och problem har uppstått. Då projektbeskrivningen från Swerea mestadels innehöll programmering samt val av ett optimalt sätt att styra motorn fanns det inte någon konfigurering samt inställning av hårdvara med i planeringen. Dessa oförutsägbara tillägg tog relativt lång tid att lösa då undersökning och information av hårdvaran var tvungen att hämtas och bearbetas.

En datalogger för att få data från lastcellen skulle implementeras i hårdvarustrukturen och denna skulle även konverteras till att kunna användas på rätt sätt.

En annan del i arbetet som tog längre tid än förväntat var att konvertera om motorn från parameterstyrning till styrning av CANopen. Genom att ta kontakt med den svenska distributören av motorn blev vi vidarekopplade till tillverkaren, Dunkermotoren i Tyskland. Genom mailkontakt och fjärrstyrning av datorn blev problemet löst efter cirka två veckor.

Det stora problemet under arbetes gång var att kommunikation mellan LabVIEW och motorn saknades. Detta medförde att vi inte kunde börja programmera styrsystemet. Vi tog då kontakt med utvecklarna av LabVIEW, National Instruments, för att höra vad de ansåg vara det bästa sättet att lösa problemet, samt egna efterforskning gjordes. Resultatet av detta blev att en ytterligare komponent, NI 8472, behövdes för att implementera en extern enhet för kommunikation via National Instruments LabVIEW. Projektledaren beslutade, efter att fått godkänt från sin ekonomiavdelning, att beställa komponenten. Detta gjordes sex veckor in i projektet.

Summan av ovanstående problem gjorde att utvecklingen av mjukvaran blev uppskjuten tidsmässigt.

När väl programmeringen påbörjades, kunde arbetet fortlöpa i ett bra tempo med ett strukturerat tillvägagångssätt, där det lades fokus på de stora och komplicerade delprogrammen. Även om inte alla krav uppfylldes utformades programmet så att de kvarstående delprogrammen är enkla att implementera i huvudprogrammet.

Ett av de problem som upptäcktes när programmeringen kommit igång var att motorn räknade negativa positioner när ställdonet skulle åka nedåt, mot cellplasten. Eftersom värdena som skrivs till ett specifikt index går via en array behövdes nollnivån befinna sig vid bottenplattan. Detta medförde att efter att nollnivån satts, implementeras endast positiva värden till arrayen. För att få ner ställdonet mot bottenplattan innan nollnivån befann sig vid bottenplattan behövdes ett negativt tal skrivas till arrayen. Detta tal skrevs hexadecimalt så att motorn förstod att den skulle operera nedåt.

Om mer tid hade tillhandahållits hade vi gått tillväga på följande sätt för att genomföra de krav som inte blev uppfyllda.
Ett krav var att donet skulle anlägga en kraft på 750 (+/- 20) N vid körningen och att detta skulle anpassas då cellplasten blev mjukare. Detta hade kunnat lösas genom att kontrollera den sinusformade körningen med data från lastcellen via DAQ Assistant. Genom att höja eller sänka offseten på sinuskurvan, under körningen, kan man anpassa kurvan så att donet når det önskade värdet.

En annan funktion som inte genomfördes var att mäta cellplastens hårdhet och hur den ändras efter körning. Detta hade lösts på ett liknande sätt som då tjockleken mäts. Genom att följa ISO standard, för hur programmet önskas vara uppbyggt, kan en programslinga för mätning av hårdhet enkelt utvecklas. För att få skillnaden i hårdhet implementeras slingan både före samt efter körningen. Alla index, subindex och värden för detta finns dokumenterade.

Till slut bör det tilläggas att programmet är välutvecklat och det finns goda möjligheter att gå vidare för att fullfölja projektet. Detta då de tunga utvecklingsbitarna är gjorda och endast ett delprogram och små finjusteringar kvarstår. Vi rekommenderar företaget att fortsätta utveckla detta program då det går att styra flera testriggar med programmet. En överlämning av programkoden har gjorts där vi pedagogiskt beskriver vad som har gjorts och hur vidareutveckling bör fortlöpa.

10. Referenser

CAN in Automation.
http://www.can-cia.org/fileadmin/cia/pdfs/candictionary_v7.pdf
(2015-05-31)

CAN in Automation. CAN protocol
http://www.can-cia.org/index.php?id=systemdesign-can-protocol
(2013-05-31)

CAN in Automation. Controller Area Network
http://www.can-cia.org/index.php?id=systemdesign-can
(2013-05-31)

CAN in Automation. NMT
http://www.can-cia.org/index.php?id=155
(2015-05-31)

Wikipedia. CANopen.
http://en.wikipedia.org/wiki/CANopen
(2015-05-15)

Wikipedia. ISO standard
http://sv.wikipedia.org/wiki/Internationella_standardiseringsorganisationen
(2015-04-20)

Wikipedia. CAN
http://en.wikipedia.org/wiki/CAN_bus
(2015-04-20)

Wikipedia. Cellplast
http://sv.wikipedia.org/wiki/Cellplast
(2015-04-20)

Wikipedia. While loop
http://sv.wikipedia.org/wiki/O%C3%A4ndlig_loop
(2015-06-05)

Wikipedia. DAQ
http://en.wikipedia.org/wiki/Data_acquisition
(2015-06-05)

Wikipedia. True/false struktur
http://en.wikipedia.org/wiki/True_and_false_(commands)
(2015-04-20)

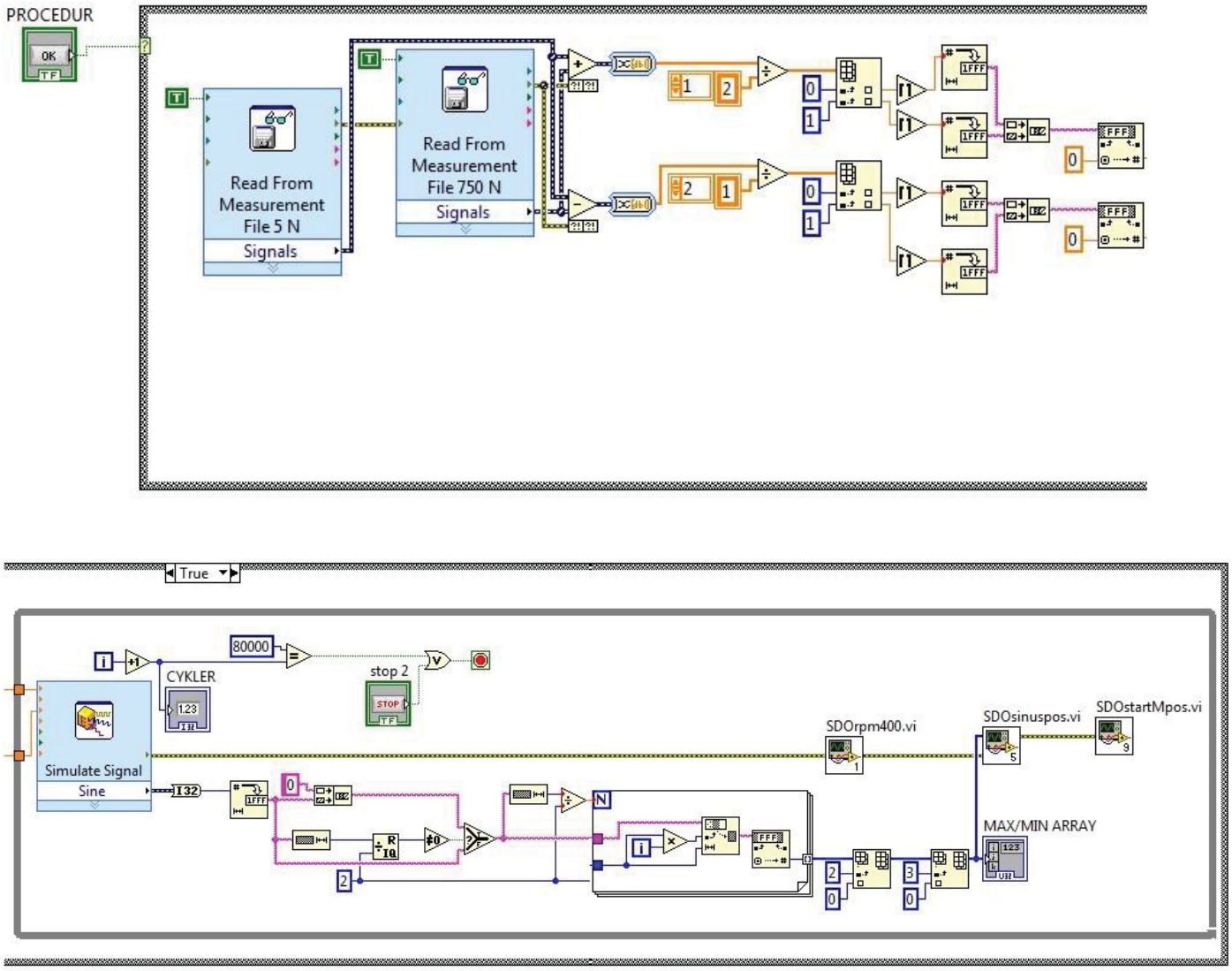
Wikipedia. Array
http://sv.wikipedia.org/wiki/F%C3%A4lt_(datastruktur)
(2015-04-20)

Wikipedia. Hexidecimala värden
http://sv.wikipedia.org/wiki/Hexadecimala_talsystemet
(2015-04-20)

Swerea. Manualer samt ISO-standard.
www.swerea.se
(2015)

National Instruments. All produktinformation.
http://sweden.ni.com/
(2015)

11. Bilagor



Figur. 18. Programkod för proceduren, uppdelad i två.