Styresystem for kybernetisk håndledd

Geir Turtum

1. desember 2013

Oppgavetekst

Instituttet har sammen med University of New Brunswick, Canada, stått sentralt i utviklingen av et motorisert protesehåndledd med unike kinematiske egenskaper. Det foreligger et maskinvaredesing og en prototyp av styresystemet, og dette systemet skal nå ferdigstilles og programmeres slik at protesen kan brukes i forsknigssammenheng. Sentrale systemegenskaper inkluderer kommunikasjon over en CAN-buss ved hjelp av protokollen PDCP, noe signalbehandling og styring av en børsteløs DC-motor.

1. Gi en kort oversikt over prosjektets bakgrunn og nåværende status, der du legger vekt på systemets tilstand relativt dets spesifikasjoner og tiltenkte anvendelse.

2. Foreliggende HW-design er basert på en enkelt mikrokontroller som skal stå for både motorkommutering, regulering, signalbehandling og kommunikasjonsprotokoll(er). Foreta en vurdering av om dette designet er egnet, med spesiell vekt på avbruddshåndtering og sanntidsaspekter. Gjør om nødvendig praktiske målinger for å underbygge din konklusjon.

3. Foreslå på bakgrunn av punkt 2 en maskin- og programvarearkitektur for systemet. Eksisterende løsninger bør gjenbrukes i den grad det er mulig og hensiktsmessig.

4. Implementer og test systemet så langt tiden tillater det.

Sammendrag

Denne prosjektoppgaven tar for seg design og utvikling av styresystemet til håndleddsprotesen NRWD(NTNU Rotary Wrist Device). NRWD ble i sin tid skapt på bakgrunn av doktorgradsavhandingen[2] til Øyvind Stavdahl.

Det har blitt skrevet flere prosjekt- og masteroppgaver[6, 3] om NRWD'en hvor alle har hatt som mål å realisere en prototype av protesen, men på grunn av ulike årsaker ikke har fullført. Denne oppgaven baserer seg i hovedsak på arbeidet til Andreas Kråkenes [3]. Han har produsert maskinvare lagt ut på flerlagskort, samt skrevet programvare for testing av maskinvaremodulene, noe som la et meget godt grunnlag for videre arbeide.

I denne prosjektoppgaven har styresystemet blitt ferdigstilt og det har blitt gjort anbefalinger til endringer i maskinvare der man har sett muligheter for forbedringer.

Kommunikasjonen med protesen går over standard analoge og digitale elektroder og med den åpne protokollen PDCP(Prosthetic Device Communication Protocol). Implementeringen av protokollen er gjort av tidligere NTNU studenter[4, 5] mens undertegnede har stått for integreringen av protokollen med protesens styresystem.

Forord

Jeg vil takke katten min Tom.

Nomenklatur

- NRWD - NTNU Rotary Wrist Device
- PDCP - Prosthetic Device Communication Protocol
- CAN - Controller Area Network
- PWM - Pulse Width Modulation
- UNB - University of New Brunswick

Innhold

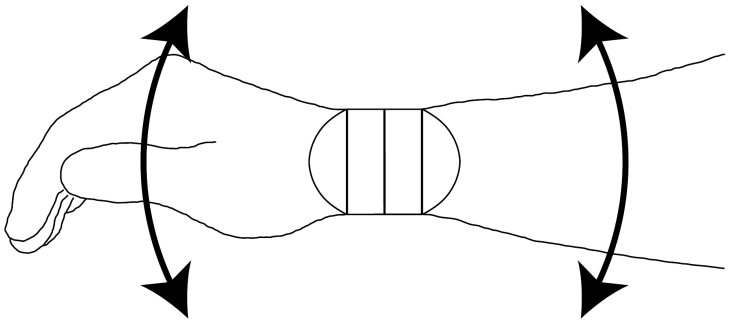
Kapittel 1

Introduksjon

1.1 Bakgrunn for oppgaven

Ideen bak protesen kommer fra Øyvind Stavdahl sin doktorgradsavhandling [2] fra 2002. I den viste eksprimenter med friske håndledd at en vinkel mellom lengdeaksen til underarm og rotasjonsaksen til håndleddet gjorde flere bevegelser enklere for brukeren å utføre.

For å teste disse resultatene i praksis trengte man en 1-akse håndleddsprotese med design som muliggjør valgfri rotasjonsaske, se figur 1.1. Som en del av mastergraden til Arthur Zink[1] ved University of New Brunswick, Canada ble mekanikken til en slik 1-akse håndleddsprotese bygget. Etter at mekanikken var ferdigstilt trengte man et styresystem for å kontrollere protesen.



Figur 1.1: Vinkel mellom håndprotese og underarm kan varieres. Kilde: Styresystem for kybernetisk håndleddsprotese[3]

Det ble også funnet ut at dette var en gylden mulighet til å teste ut den åpne protesestyrings protokollen PDCP som da var under utvikling hos UNB. Det ble lagt til som et krav i funksjonsspesifikasjonen til NRWD'en at den

skulle kommunisere ved hjelp av denne protokollen, i tillegg til at den skulle kunne styres med de mer konvensjonelle analoge og digitale elektrodene.

1.2 Tidligere arbeide

Realisering av styresystemet til protesen har hittil vært påbegynt i to mastergradoppgaver. Den siste av disse var skrevet av Andreas Kråkenes i 2011[3] og kom langt på vei til å fullføre designet. Hovedgrunnen til at protesen ikke enda har blitt fullført skyldes at det første designet innholdt en alvorlig arkitektur feil som innvolverte kommunikasjonen mellom to mikrokontrollere. Det ble forsøkt fikset på uten hell og til slutt etter mye tid gikk man over til en løsning som besto av at en mikrokontroller skulle styre hele systemet.

Da prosjektet ble overtatt av undertegnede var alt av hardware designet og lagt ut på prototype kort. Dessverre var skjemategningen bare tilgjengelig i pdf format så utlegg må gjøres på nytt når kretsen skal miniatyriseres for å få plass inni protesen. Software for å teste deler av systemet var ferdigskrevet, men koden var av varierene kvalitet og mye funksjonalitet var enda ikke implementert. Der det manglet mest var de ulike regulatorene og implementasjonen av PDCP protokollen.

I 2012 ble laget en implementasjon av PDCP protokollen av to mastergradsstudenter ved NTNU[4, 5]. UNB hadde tildigere laget en implementasjon basert på PIC mikrokontrolleren, mens denne var designet for Atmel sine AVR mikrokontrollere. Implementasjonen var designet for at det skulle være enkelt å overføre koden til ny maskinvare noe som gjorde at den egnet seg godt for dette prosjektet.

1.3 Mine bidrag

En naturlig start på prosjektet var å gå gjennom den overleverte maskinvaren og medfølgende driverkode modul for modul. Detaljer rundt hvordan de ulike modulene ble testet er beskrevet i kapittel 5.

Den overleverte koden var av varierende kvalitet. Koden var strukturert på et fornuftig vis, med en c fil for hver maskinvare modul. Denne strukturen ble beholdt ut prosjektet. Selve driverkoden trengte en del mer arbeid. Det var skrevet nok til å teste basis funksjonaliteten til hver modul, men måtte utvides med tilleggsfunksjonalitet før den kunne brukes.

Under testing av maskinvaren ble det funnet muligheter for forbedringer i spenningsregulator og posisjonsensor modulene. Anbefalinger til endring i maskinvaren er gjort i kapittel 5.

Det første styressystemet som ble designet til NRWD'en benyttet seg av to mikrokontrollere for å håndtere alle oppgavene til systemet. Denne arkitekturen gikk man bort ifra da man fikk problemer med kommunikasjonen mellom de to kontrollerne og man gikk over til å bruke en mikrokontroller. Som en del av oppgaven ble det gjort grundige undersøkelser om den valgte mikrokontrolleren hadde nok ressurser til å styre systemet på egen hånd. Diskusjonen rundt dette er gjort i kapittel 4.

Ifølge funksjonsspesifikasjonene skal protesen ha minst tre forskjellige styringsmoduser, maks fart i valgt retning, posisjon- og hastighetsstyring. De forskjellige regulatorene ble alle skrevet fra bunnen av. For detaljer til hvordan de ulike regulatorene ble implementert se kapittel 6.

På slutten av prosjektet ble styringssystemet integrert med kommunikasjons protokollen PDCP. Implementasjonen av protokollen var all erede gjort, jobben besto av å skrive om maskinvare abstraksjons laget. Hvorfor og hvordan dette ble gjort er beskrevet i kapittel 6.

1.4 Utviklingsmetodikk og fremgangsmåte

Hvordan dette prosjektet best skulle styres falt klart for meg fra begynnelsen av. Det er et maskin- og programvare system som allerede da det ble overtatt var delt opp i klare moduler. Som nevnt tidligere var det meste av maskinvare designet ferdig, mens mye av programvaren enda måtte skrives. Det var også mye programvare som var skrevet av tidligere studenter som skulle flettes inn i prosjektet for ikke å reprodusere arbeide som var gjort tidligere, det tenkes spesielt på lavnivå implementasjonen av PDCP protokollen som ble utviklet av Andrzej Zamojski [5] og Andreas Nordal[4] samt programvare driverne til NRWD'en skrevet av Andreas Kråkenes.

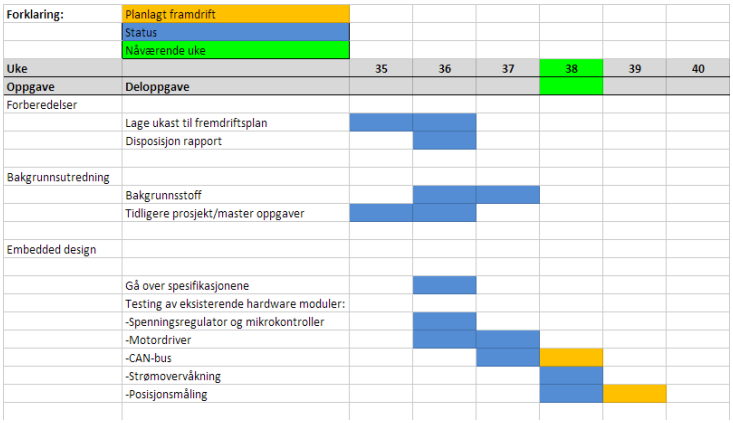
Metoden min sin hovedlinje var og i størst grad arbeide på prosjektet modul for modul, forsikre meg om at alt fungerte separat før til slutt å sette modulene sammen til et komplett styresystem. Kort oppsummert punktvis var planen:

- Sette seg inn i bakgrunnsstoff for oppgaven.
- Test hver maskinvare modul for seg, samt tilhørende programvare driver.
- Sjekk om maskinvare og programvare oppfyller kravene satt av funksjonsspesifikasjonen.
- Fullfør driverkoden til modulen som blir testet.

- Sett seg inn i implementasjonen av PDCP gjort av tidligere NTNU studenter.
- Integrere PDCP med styringssystemet.

Hvordan hver modul ble testet blir gått gjennom i kapittel 5.

Verktøyet som ble brukt for å budsjettere tiden og sørge for at man kom i mål med prosjektet til rigktig tid var, på anbefaling fra veileder, Gantt diagrammet. Figur 1.2 viser framgangsplanen for de første ukene av prosjektet og hvorvidt planen ble overholdt. Gantt diagrammet var et nyttig verktøy å bruke, ikke bare ga det en klar oversikt over hvilke oppgaver man hadde igjen, men også hvorvidt man greide å overholde de tidsfrister man hadde satt.



Figur 1.2: Gantt diagram

Utviklingen av programvare ble gjort på en windows maskin med Atmel Studio 6. For versjonskontroll av kode og dokumenter falt valget på Git.

1.5 Rapportens oppbygning

Skrive om hva som kommer når? JA! Bluphbluhp

Kapittel 2

NRWD Spesifikasjoner

Dette kapittelet går gjennom funksjonsspesifikasjonene til NRWD'en punkt for punkt. Spesifikasjonene beskriver kravene til systemet i sin hellhet fra de fysiske dimensjonene, kommunikasjons grensesnittet opp mot protesen, oppløsningen på de ulike sensorene, maks vinkel hastighet til protesen og så videre.

Delkapittlene som følger tar for seg hver sin del av funksjonsspesifikasjonene. De beskriver hvike av kravene som er oppfylt, de som enda ikke er implementert og de som krever mer arbeid. Dette kapittelet beskriver systemet i sin nåværende tilstand og er ment for å hjelpe neste person som skal arbeide på prosjektet til å få en rask oversikt over hva som er gjort og hva som må gjøres. For noen av kravene vil statusen være kommentert som "Ikke relevant", dette gjelder de spesifikasjonene som omhandler de rent mekaniske sidene av systemet.

Ikke tatt med i dette kapittelet er de overordente funksjonsspesifikasjonene til NRWD'en. De kan leses i det originale spesifikasjons dokumentet vedlagt denne oppgavenB.

2.1 Wrist Joint Function, WJF

Beskriver det roterende leddet i protesen og hvordan festeanordningen til underarm og eventuell håndleddsprotese skal være manuelt justerbar.

Nr	Beskrivelse	Status
WJF-01	The NRWD joint shall be a single, simple revolute joint which axis of rotation can be placed at an attitude with respect to the forearm and terminal device as spedcified in [1], Equations (7.4) and (7.5).	Ikke relevant
WJF-02	The joint axis should be manually adjustable to other attitudes than that specified in WJF-01.	Ikke relevant
WJF-03	The joint shall enable an angular excursion of at least 180 degrees. The excursion should be unlimited.	Ikke relevant

2.2 Wrist Motor Function, WMF

Beskriver kravene til protesens vinkelhastighet og moment. Viktig å merke seg at det her settes krav til at motoren skal være beskyttet mot overopphetning i maskinvare.

Nr	Beskrivelse	Status
WMF-01	The wrist joint (output of the gear train) shall have a maximum angular velocity of at least 1.4 rad/s (81 deg/s). The maximum velocity should be as high as possible.	Høyeste målte vinkel hastighet er målt til 2.72 rad/s (156 grader/sekund)
WMF-02	The wrist joint should have a maximum torque of at least 34,3 mNm.	Ikke målt, men ifølge målinger gjort av Zink [1] er stall torque på 59.6 mNm.
WMF-03	The motor shall have a maximum mechanical output power of at least TBC W.	NA
WMF-04	The motor shall be protected from overheating. The protection should be implemented by hardware.	Dette har blitt implementert i strømovervåkings modulen.

2.3 Wrist Servo Function, WSF

Beskriver kravet til posisjonssensoren og hvilke regulatorer som systemet skal implementere.

Nr	Beskrivelse	Status
WSF-01	The movements of the wrist joint shall be controllable according to the follwing modes: • On/Off-mode • Position mode • Velocity mode	Alle kontrollerne er Implementert i controller.c
WSF-01-01	In On/Off-mode the motor shall be at rest or run at maximum speed (open-loop) in one direction according to a given setpoint.	Implementert ved bruk av PI-kontroller som kjører motoren nærmest mulig maksimum tillatte strømforbruk.
WSF-01-02	In position mode the joint angle shall be proportional to a given angular setpoint.	Implementert med PI-kontroller.
WSF-01-02-01	The WSF shall include an absolute position sensor. This sensor shall provide no less than 10 bits resolution per revolution.	En 12-bits magnetisk enkoder er brukt.
WSF-01-03	In velocity mode the joint angular velocity shall be crudely proportional to a given velocity setpoint.	Implementert med PI-kontroller.
WSF-01-03-01	The WSF shall include a velocity sensor or estimator.	Hastighet er estimert ved hjelp av hall elementene.
WSF-02	The movements of the wrist joint should be controllable according to the follwing modes: • Torque mode • Impedance mode	Ikke implementert.
WSF-02-04	In torque mode the motor torque shall be proportional to a given torque setpoint.	Ikke implementert.
WSF-02-04-01	The WSF should include means for monitoring motor current.	Implementert, se strømovervåknings modulen.
WSF-02-05	In impedance mode the mechanical impedance of the joint shall be determined by a given impedance setpoint	Ikke implementert.
WSF-03	WSF shall provide an interface to WCF through which the modes (described in WFS-01 to WSF-02) can be selected and relevant parameters can be set, and through which WSF can report relevant state variables TBD.	Koden er lagt opp for at dette enkelt kan bli implementert når kommunikasjons spesifikasjonen er klargjort.

2.4 Wrist Communication Function, WCF

Beskriver hva slags kommunikasjonsgrensesnitt systemt skal implementere.

Nr	Beskrivelse	Status
WCF-01	The NRWD shall have a two-wire Proximal Communication Interface (PCI) and a two-wire Distal Communication Interface (DCI).	Implementert ved CAN-bus.
WCF-02	The PCI shall be configurable so that it implements two (0V, 7,2V) analog input lines or a bidirectional twowire CAN interface with the PDCP protocol(Losier,2009).	Implementert ved CAN-bus og PDCP-protokollen.
WCF-02-01	The analog input lines shall be able to sample both lines at a rate of 1 kHz. The sampling rate should be as high as 2 kHz.	Implementert i analog.c
WCF-03	The DCI shall be configurable so that it implements two (0 V, 7.2 V) analog output lines or a bidirec[1]onal two-wire CAN interface with the PDCP-protocol.	Implementert ved CAN-bus og PDCP-protokollen. Implementasjonen av PDCP-protokollen er ikke ferdig.
WCF-04	The WCF shall be configurable to an all-Analog mode, with the PCI as an analog input interface and the DCI as an analog output interface.	Analog output interface ikke implementert i verken HW eller SW.
WCF-05	The WCF shall be configurable to an all-digital mode, with the PCI and the DCI acting as bidirectional CAN bus interfaces.	Implementert. CAN buss trenger bare et tilkoplingspunkt.
WCF-07	The WCF should be configurable to a hybrid mode in which the PCI acts as two analog input lines while DCI acts as a bidirectional CAN interface (cf. WCF-01)	Ikke implementert i HW.
WCF-10	The WCF shall include a serial interface for downloading software and for debugging/diagnostic purposes.	Implementert ved bruk av JTAG.
WCF-11	WCF shall implement an interface to WSF according to WSF-03.	Se WSF-03.

2.5 Wrist Power Function, WPF

Beskriver kraftforsyningen til systemet.

Nr	Beskrivelse	Status
WPF-01	The WPF shall accept external power in the form of an unregulated two-wire DC supply.	Implementert ved brukt av intern spenningsregulator.
WPF-02	The NRWD shall tolerate and run normally when powered with a voltage in the range (6 V, 12 V). The range of usable voltages should be as wide as (5 V, 18 V).	Valgte spenningsregulator opererer normalt innenfor området (7V, 18V). Det er anbefalt å gå over til en Low Droupout Regulator for å tilfredstille kravet.
WPF-03	The NRWD shall tolerate supply voltage in the range (0 V, 12 V) without exhibiting unpredictable behaviour and without getting damaged.	Implementert ved bruk av brown out deteksjon internt på AVR'en. [todo]
WPF-04	The NRWD should automatically limit its motor current to a level that does not reduce the supply voltage below the interval given in WPF-01.	Ikke implementert.

2.6 Proximal Attachment Function, PAF

Beskriver festeanordningen og grensesnittet mellom NRWD'en og underarmen.

Nr	Beskrivelse	Status
PAF-01	The PAF shall comprise two parts, the proximal of which is adapted to be permanently attached to the forearm socket and the distal permanently attached to the wrist unit. The parts must ?mate? to form a mechanically stable connection while also being detachable.	Ikke relevant
PAF-01-01	The proximal part of the PAF shall be hollow to allow access to the space within the socket proximally to the wrist.	Ikke relevant
PAF-02	PAF disconnection should be possible with hand or a simple tool, e.g. a screwdriver.	Ikke relevant
PAF-03	The PAF shall include a four-wire electric coupling, preferably mechanically integrated with the PAF itself.	Ikke implementert i prototype hardwaren.
PAF-03-01	The PAF electrical coupling shall include at least two power supply wires/contacts capable of transferring a constant current of 4 A per wire.	Ikke implementert i prototype hardwaren.

2.7 Distal Attachment Function, DAF

Beskriver festeanordningen og grensesnittet mellom NRWD'en og en eventuell håndprotese.

Nr	Beskrivelse	Status
DAF-01	The DAF should comprise two parts, the proximal of which is permanently attached to the wrist and the distal permanently attached to the terminal device. The parts must ?mate? to form a mechanically stable connection while also being detachable.	Ikke relevant
DAF-02	The DAF shall include a four-wire electric coupling, preferably mechanically integrated with the DAF itself.	Ikke implementert i prototype hardwaren.
DAF-02-01	The DAF electrical coupling shall include at least two wires/contacts capable of transferring a constant current of 2 A per wire, and these wires shall be connected to the power supply wires from PAF.	Ikke implementert i prototype hardwaren.
DAF-02-02	The DAF electrical coupling should be rotatable without twisting the wires.	Ikke implementert i prototype hardwaren.

Kapittel 3

Mekanikken

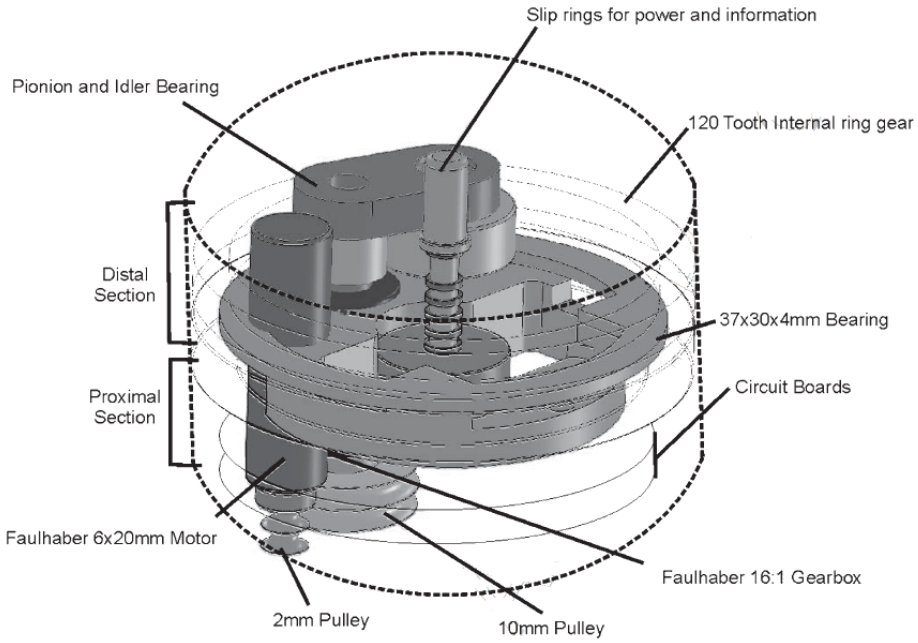
Mekanikken i protesen er et resultat av masteroppgave skrevet ved universitetet i New Brunswick, Canada. Dette ble gjort på oppdrag fra Øyvind Stavdahl for å teste resultatene fra sin doktorgradsavhandling i praksis. Mekanikken består av to seksjoner, den proksimale og distale(henholdsvis seksjonen nærmest underarmen og den lengst unna). De to seksjonene kan rotere fritt mot hverandra ved hjelp av et kulelager. Selve bevegelsen er drevet av en børsteløs DC-motor som er koplet til et girtog med utveksling 120:1 [1].

Under testing av motordriveren ble det funnet at det, ved en forsyningsspenning på 12V, måtte til en dutycycle på oppmot 60% for å i det hele tatt bevege mekanikken. Ved rundt 80% dutycycle ble strømmen til motoren kuttet av strømovervåkingsmodulen da strømmen gjennom motoren oversteg maksimum grensen. Denne grensen er satt i maskinvare for å unngå at motoren blir ødelagt på grunn av overopphetning. Dette ga regulatoren et veldig lite rom å jobbe innenfor noe som gjorde implementasjonen av posisjon og hastighetskontrolleren meget vanskelig.

Mekanikken ble tatt med til mekaniske verksted ved NTNU for å se om de kunne løse problemet. Det viste seg at aksen til det minste tannhjulet var bøyd noe som førte til ekstra friksjon i systemet. Tannhjulet det er snakk om er det som er koblet til Faulhaber girboksen i den distale seksjonen, se figur 3.1. Verkstedet fikk bøyd det til slik at det gikk fra å ha en kast på 0.22 mm til 0.03 mm. Dette reduserte friksjonen betraktelig og mekanikken kunne nå beveges med en dutycycle ned mot 40%.

Da mekanikken ble undersøkt ble det også funnet at kulelageret i protesen begynte å bli slitt noe som førte til yterligere friksjon i systemet.

Om rettingen av giraksen løser problemet på langt sikt vil vise seg med tiden. At problemet i det hele tatt har oppstått tyder på at mekanikken ikke er riktig dimensjonert for de krefter den kan bli utsatt for. En teori på hvordan problemet kan ha oppstått er at man har vridd på den distale



Figur 3.1: Oversiktbilde over den indre mekanikken til protesen. Figur hentet fra [1]

seksjonen, noe som påfører store krefter på den indre mekanikken. Dette tyder på at mekanikken kan ha behov for en ny revisjon før den er robust nok til å kunne brukes i eksprimenter.

For fremtidig bruk er det lagt ved en oversikt over rotasjonshastigheten til protesen ved forskjellige dutycycler etter at mekanikken var rettet av verkstedet, se vedlegg C.1. Denne kan brukes for å se om problemet har gjenoppstått.

Kapittel 4

Interrupt diskusjon

Hvorfor diskutere interrupt

Problemer med gamle arkitektur

Hva slags system er dette her? Soft real time?

Forklar hvordan systemet er bygget på interrupts

Hvorfor er det viktig at alle interrupts blir håndtert

Hvordan håndterer AVR interrupts

KOrt om hvilke interrupt systemet må håndtere

Tidligere forsøk på å realisere styresystemet til NRWD'en har brukt to mikrokontrollere. Den ene skulle brukes til motor kommuteringen mens den andre skulle håndtere alle de andre oppgavene som regulering og kommunikasjon. Da systemet ble testet oppsto det problemer med kommunikasjonen mellom mikrokontrollerene og selv etter mye testing fant man aldri årsaken til problemet. Dette førte til at man gikk over til en arkitektur som brukte en enkelt mikrokontroller.

Spørsmålet var da om den valgte mikrokontrolleren, AT90CAN128 kjørende på 16 MHz, var nok til å håndtere alle oppgaver som før skulle blitt utført av to mikrokontrollere. Dette kapitellet tar for seg denne problemstillingen.

Etter at alle programvare modulene har blitt initialisert er det bare den valgte regulatoren(posisjon/ hastighet) som beregner pådrag som blir håndtert inni while loopen. Alt annet blir håndtert inni interrupt rutinene, som motor kommutering, posisjonsoppdatering, kommunikasjon osv.

Om ikke systemet har nok resurser til å håndtere alle interruptene som kommer vil man risikere å miste kommunikasjons meldinger og at motoren ikke lenger blir kommutert riktig som begge vil medføre at du får et system som virker tregt og uresponsivt. For å kunne gi en garanti om at dette ikke ville skje ble systemet analysert for tidsbruk i interrupt rutiner.

Typen interrupts som systemet skal håndtere kan deles inn i to typer, periodiske og aperiodiske.

De periodiske er som følger:

- HALL_X_vect - Trigger hver gang en av de tre hall sensorene i motoren skifter nivå
- TIMER3_CAPT_vect - Trigger på hver flanke til PWM signalet fra posisjons sensoren
- TIMER0_OVF_vect - Brukes til å beregne hastigheten til motoren

De aperiodiske er:

- CANIT_vect - Trigger hver gang det kommer en CAN melding
- CURRENT_vect - Trigger når hardwaren slår av strømmen til motoren
- RESET

Varigheten til hvert interrupt ble målt ved å sette en GPIO-pinne høy ved inngangen av interruptet og lav i den gikk ut og lese av oppetiden på oscilloskopet. Deretter ble tiden det tar for mikrokontrolleren å gjøre kontekst skiftet lagt til. Se tabell 4.1 for oppsumering av tidsbruk til de periodisk interruptene.

Navn	Varighet (μs)	Maks frekvens	μs per sekund
HALL_X_vect	4.5	3600	16200
TIMER3_CAPT_vect	4.8	490	2352
TIMER0_OVF_vect	3.2	61	196
Totalt			18742

Tabell 4.1: Timing av periodiske interrupts

Vi ser av tabell 4.1 at mikrokontrolleren bruker mindre enn 2% av tiden inni en interrupt rutine. Det gir rikelig med tid til å håndtere kontroll loopen som kjører utenfor interruptene.

Det ble også undersøkt om mikrokontrolleren er rask nok til å håndtere kommuteringen av motoren. En den for treig vil den ikke kunne kommutere raskt nok til at motoren oppnår den ønskede hastighet. ATMEL sin applikasjonsnote AVR452 som tar for seg motor styring av BLDC motorer ved bruk av AT90CAN mikrokontrolleren setter en teoretisk maksgrense ved 3478K rpm ved 8 MHz, da maksfarten til NRWD motoren er satt til 3600 rpm befinner vi oss godt innenfor kravene.

Kapittel 5

Modul testing/ implemenstasjon og diskusjon

??Før arbeidet startet ble det skrevet korte testplaner for hver maskinvare modul. De ble så gjennomgått og testet en etter en slik at jeg kunne stole på at det av maskinvare jeg brukte fungerte som beskrevet. Det falt seg naturlig å teste driverkoden sammen med maskinvare modulen under testing, og skrive den om der det ble funnet feil eller mangler. ??

Dette prosjektet startet ikke på bar bakke, men bygger videre på arbeidet til flere studenter. Således var det mye dokumentasjon, maskinvare og programvare som måtte gjennomgåes før det kunne bygges videre på. For å være sikker på at det som ble bygget på var av akseptabel kvalitet ble hver modul gjennomgått for om den fungerte i henhold til det som sto beskrevet i tidligere oppgaver. Masteroppgaven til Andreas Kråkenes var spesielt nyttig her siden han var siste student som jobbet på prosjektet og således er nesten alt av kode og hardware for NRWD'en som ble overlevert, produsert og dokumentert av han.

Før arbeidet startet ble en kort testplan for hver modul utarbeidet, den ble laget slik at da alt i planen var testet skulle man være sikker på at maskinvare og programvare for den modulen oppførte seg som forventet. Under følger testplanene som ble brukt, samt en diskusjon rundt hver modul.

5.1 Mikrokontroller

Det første som ble testet var om det var mulig å få kontakt med mikrokontrolleren. Til det ble det brukt en JTAGICE 3. Alt fungerte bra.

5.2 Spenningsforsyning

Spenningsregulatoren brukt i NRWD'en er en linær regulator som i henhold til databladet gir ut 5V +- 0.2V ved inngangsspenninger 7-20 V. Ifølge funksjonsspesifikasjonen WPF-02 skal NRWD'en håndtere spenninger mellom 6-12 V. Det er viktig at NRWD'en kan operere på spenninger ned til 6V siden de vanligste batteriene som er brukt i slike proteser gir ut en spenning som kan gå ned til minst 6V. Når differansen mellom inngangs- og utgangsspenning kan være så lav som 1V er en LDO (Low Dropout Regulator) et bedre valg.

5.3 Posisjonsmåling

Ifølge kravspesifikasjonene B til NRWD'en skal den være utstyrt med en absolutt posisjonssensor med en oppløsning på minst 10 bit. Løsningen som har blitt valgt av min forfølger er en Hall sensor enkoder som leser av vinkelen til magnetfeltet gikk av en magnet festet på den distale[?] delen av NRWD'en. Den gir ut et PWM signal med dutycycle [?] proporsjonalt med vinkelen til magnetfeltet på sensoren. Til nå har en av interrupt pinnene blitt satt opp med flanke deteksjon hvor den først ser etter en stigende flanke, skrur på timeren og konfigurerer pinnen til å få interrupt på synkende flanke. Når neste interrupt blir trigget leser man av timeren og får derved vinkelen. Problemet med dette oppsettet kommer ved vinkler nær 0 og 360 grader. Ved så små og store vinkler vil tiden mellom flankene være så korte[sett inn figur] at mikrokontrolleren ikke vil rekke å behandle interruptet før det neste kommer og avlesningen vil bli feil. Det har derfor blitt valgt å gå over til å bruke et lavpassfilter på sensorsignalet så vinkelen kan leses av ved bruk av en ADC. Det vil ha tilleggsfordelen at man ikke bruker så mye tid på å lese av posisjon som nå blir gjort siden frekvensen til signalet fra sensoren allerede er satt til 240[ref], noe som er mye høyere enn det som kreves for god kontroll.

Siden signalet har en frekvens på 204[?] har det blitt valg er RC filter med knekkfrekvens []. ADC pinnen har en intern pullup motstand så den eneste eksterne komponenten som trengs er en kondensator. Verdien på den interne pulloppmotstanden er [?] så for å få den ønskede knekkfrekvensen ble en [?] F stor kondensator valgt.

5.4 Motordriver

Ifølge funksjonsspesifikansjonen, punkt WMF-01B, skal protesen hvertfall ha en maksimums omdreiningshastighet på 81 grader/sek. Da motoren er giret

ned med en ratio 800:1 vil det tilsvare en motorhastighet på 10800 rpm.

(81 grader/sek * 800 * 60 sek / 360 grader = 10800 rpm)

Ifølge databladet[ref] er makshastigheten til motoren 35600 rpm noe som faller godt innenfor kravet. Da motorens hall sensorer genererer 6 interrupt per omdreining vil mikrokontrolleren måtte kunne håndtere minst opptil 3560 interrupts i sekundet, se formell 5.1.

$$\frac{35600 \text{ rpm}}{60 \text{ s}} \times 6 \text{ interrupts} = 3560 \text{ interrupts} \tag{5.1}$$

En testbenk ble satt opp for å teste om mikrokontrolleren greie å håndtere alle motor sensor interruptene. Et arduino brett ble brukt for å simulere utgangene til motoren sine hall sensorer med opptil en hastighet som tilsvarer 400'000 rpm. Arduino brettet var programmert til å påføre et vist antall interrupts, mikrokontrolleren ble satt opp til å telle inkommende interrupts og da antallet stemte overens viste man at mikrokontrolleren taklet disse hastighetene.

5.5 Strømovervåkning

Strømovervåkningsmodulen skal sørge for at motoren ikke overoppheter og gi mikrokontrolleren mulighet til å måle hvor mye strøm motoren trekker. Dette i henhold til de to funksjonsspesifikasjons kravene, WMF-04[ref] som sier at motoren skal være beskyttet mot overopphetning i hardware og WSF-02-04-01[ref] som sier at man skal kunne lese av hvor mye strøm motoren trekker.

[legg inn skjemategning]

Som vi ser av skjemategningen går motorstrømmen over den 1 ohms målemotstanden. Deretter blir signalet lavpassfiltrert, motoren er styrt med en 20 kHz PWM signal, og forsterket opp av en operasjonsforsterker. For å få strømovervåkning har utgangen fra operasjonsforsterkeren blitt lagt ut til en av mikrokontrolleren sine ADC innganger. Da kan strømmen leses av ved denne formelen.

```
/* Multiply K with ADC value to get current in uA
* KuA = ADCref * (1'000'000/1024)/(Aopamp) = 311
* ADCref = 5 V
* Aopamp = 15.7 (opamp gain)
*/
uint16_t K = 31; //311
//kAdc = K*adc;
current_mA = (K*ADC)/100;
```

Utgangen fra operasjonsforsterkeren går også inn på en komparatorkrets som setter en SR-vippe, denne sørger for å slå av strømforsyningen til motoren når den trekker mer enn den maksimalt tillatte strømmen. SR-vippen kan bli resatt av mikrokontrolleren.

Når komperatorkretsen detekterer overstrøm vil det også bli generert et eksternt interrupt på mikrokontrolleren så den kan behandle situasjonen.

Utregningen av den maksimalt tillatte strømmer er tidligere gjort av Andreas Kråkens [3] utifra motorprodusentens tekniske informasjonshefte og er satt til 145 mA.

Kretsen ble testet ved å måle strømmen gjennom motoren ved hjelp av et multimeter. Komparatorkretsen slo av strømmen idet den nådde den maksimalt tillatte verdien.

5.6 CAN

Kapittel 6

Programvare implementasjon

Det som i hovedsak manglet ved overtagelsen av prosjektet var å skrive ferdig programvare til NRWD'en. Det var skrevet drivere til de fleste modulene hvor koden var av varierende kvalitet, skrevet mer for å teste hardware'en enn for å kunne brukes i det ferdige produktet.

Modulariseringen av koden ble beholdt da den allerede var delt opp i fornuftige deler.

Systemet er for det meste drevet av interrupts, både interne og eksterne. Motorkommuteringen, posisjonsmålingen, motagelser av CAN-meldinger, fartsmåling og så videre skjer alle inni interruptrutiner. Når koden som kjører til en hver tid kan bli avbrutt av en interrupt rutine er det meget viktig å kode slik at det ikke oppstår race[?] kondisjoner. Den enkleste måten å forhindre dette på er å sørge for best mulig seperasjon mellom modulene, så ikke en modul kan endre på variabler brukt av en annen modul.

Det nederste laget er driverne til de forskjellige hardware modulene, current.h for strømovervåkning, position.h for posisjonssensoren, motor.h for motorstyring og can.h for det nederste laget av PDCP protokollen.

Selve implementasjonen av PDCP protokollen er en porting av programvare skrevet av Andreas Nordal og Andrzej Zamojski i sine masteroppgaver [4, 5].

Kapittel 7

Diskusjon

Diskutere, reparere, filsofere..

Kapittel 8

Konklusjon

Bra jobba geir

Bibliografi

[1] Arthur Zinck, *Design of a compact, reconfigurable, prosthetic wrist.* 2011.

[2] Øyvind Stavdahl *Optimal wrist prosthesis kinematics: Three-dimensional rotation statistics and parameter estimation.* Department of Engineering Cybernetics, Norwegian University of Science and Technology. 2002. PhD Thesis.

[3] Andreas Kråkenes, *Styresystem for kybernetisk håndleddsprotese.* Master i teknisk kybernetikk 2011.

[4] Andreas Nordal, *Implementasjon og testing av en åpen bussprotokoll for armproteser.* Master i teknisk kybernetikk 2012.

[5] Andrzej Zamojski, *Design, Implementation and Testing of Low-level Layers of the PDCP for the AVR Platform.* Master i teknisk kybernetikk 2012.

[6] Inge Brattbakken *Embedded control system for cybernetic wrist prosthesis* Master of Science in Engineering Cybernetics 2010.

[7] Yves Losier *Prosthetic Device Communication Protocol for the AIF UNB Hand Project* UNB Hand Project Systems Integrator 2012.

Tillegg A

Innhold CD

Tillegg B

Funksjonsspesifikasjon NRWD

Tillegg C

Diverse

Duty cycle	Vinkelhastighet
80	156 grader/sekund
65	94 grader/sekund
50	44 grader/sekund

Tabell C.1: Vinkelhastighet til NRWD'en ved ulike duty cycles, 12V spenningsforsyning