BEERTRESS

System test

Semesterprojekt 3, Gruppe 8

Peter Wann
201907121

August Hjerrild Andersen
201907251

Simon Phi Dang
201705957

Henry Pham
201606071

Alexander Flarup Wodstrup
201810602

Lucas Friis-Hansen
201811527

Jim Sørensen
201602614

Shynthavi Prithviraj
201807198

17. december 2020

Indhold

1 Hardware Test

1.1 Fremgangsmåde

Alle komponenter, med undtagelse af spændingsregulator, er komponenter der er lånt udefra. Derfor er der under hardware testing sikret, at alle komponenter virkede efter hensigten. De vil derimod blive testet mere dybdegående under software-afsnittet, eftersom komponenterne har brug for kode. Derfor er der heller ikke integrationstest i dette afsnit, da der under integrationstesten for software bliver gjort brug af hardware komponenterne.

1.2 Modultest

1.2.1 Køretøj

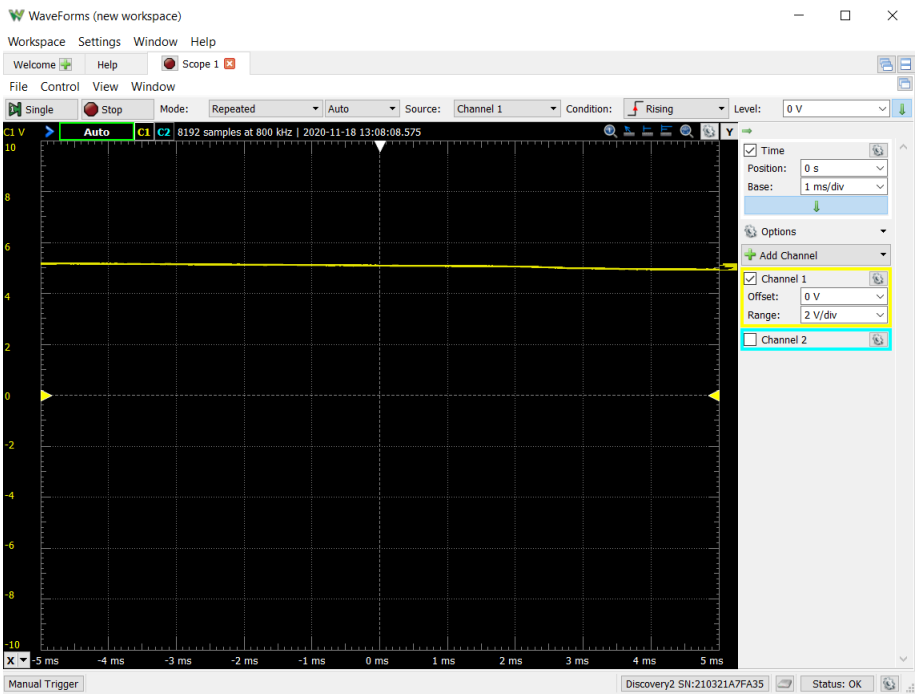
Det udlånte køretøj blev testet ved at sætte en forsyning på 10V/1A til et fumlebræt og derefter tilslutte de parvis DC-motorer en af gangen. Der blev observeret at larvefødderne bevægede sig efter hensigten efter at blive tilsluttet spænding. Derudover blev der observeret, at køretøjets larvefødder bevægede sig invers af hinanden, så en implementering med H-bro blev løsningen på dette problem. Køretøj med H-bro blev ikke testet af i hardware testen, da man, for at teste H-bro, har brug for kode til PWM- og pinstyring. Det blev derfor først testet af under modultesten for motorstyring.

1.2.2 Batteri og Spændingsregulator

I denne test blev hvert batteri først testet af for at sikre, at de begge havde det korrekte output. Derfor blev de tilsluttet batteriterminalen uden spændingsregulator og målt med voltmeter. Begge batterier viste her den korrekte måling på hhv. 9.6V og 7.2V. Dernæst blev batteriet tilsluttet 7.2V på batteriterminalen med spændingsregulator og målte spændingen på udgangen af LM7805. Den blev målt til 5V og virkede derfor efter hensigten. Dernæst blev der målt på udgangen af LM1085 og her blev spændingen målt til 3.3V og virkede derfor også efter hensigten.

1.2.3 Load cell og Load cell amplifier

For at teste om load cell sender korrekt signal videre til systemet, blev det sat op og testet med et analog discovery. For at sikre at 1 kg kan måles korrekt, blev en 1 kg vægt sat på vægten og load cell amplifieren blev justeret til at give præcis 5V som output på Aout pin. Denne test kan ses på nedenstående figur 1



Figur 1: Aout med 1 kg vægt kalibrering

Ved at have 1 kg resulterer i 5V kan input pin på PSoC aflæse korrekt imellem 0-5V.

2 Software Test

2.1 Fremgangsmåde

I software delen, er der først blevet lavet modultest på alle de forskellige moduler, så sensorerne, motoren, I2C og kunde- og personalegrænsefladerne er blevet testet individuelt. Derefter blev der påbegyndt en integrationstest med motoren i centrum, da de fleste test var afhængig af denne, før at man kunne få en idé om, om systemet fungerede. Hvert modul som havde forbindelse til vores PSoC blev testet med motoren for at teste funktionaliteten, og til sidst blev alle modulerne sat sammen i en systemtest. Der er altså benyttet en "bottom-up"strategi, hvor der til sidst har været en form for "big-bang"strategi.

2.2 Modultest

2.2.1 Motorstyring

For at teste motorstyring blev der implementeret en UART i en main, så man via en terminal (f.eks. RealTerm) kunne indtaste input, alt efter hvilken funktion man ønskede at teste. Dette blev gjort ved hjælp af switch cases, så hvis man f.eks. trykkede '1' i terminalen ville man teste driveForward osv. Dette blev programmeret til PSoC. Der blev valgt en PWM på 100 procent under hele testen. Der blev også i main initieret PWM ved hjælp af initMotor-funktionen.

Vi satte dernæst køretøjet med H-broer sammen med PSoC og forsynede de nødvendige dele med spænding på hhv. 9.6V, 5V og stel og åbnede terminalvinduet. Den printede de tilgængelige kommandoer ud i outputvinduet. Testen af de forskellige funktionen kan ses i nedenstående tabel 1

Kommando	**Funktion**	**Observering**
'1'	driveForward	Køretøjet kører fremad og begge larvefødder bevæger sig i samme tempo.
'2'	driveBackward	Køretøjet kører baglæns og begge larvefødder bevæger sig i samme tempo.
'3'	driveStop	Køretøjet stopper.
'L'	driveLeft	Køretøjets larvefødder bevæger sig **kun** i venstre side.
'R'	driveRight	Køretøjets larvefødder bevæger sig **kun** i højre side.

Tabel 1: Komponentliste for Batteritilslutning og Spændingsregulator

Som det kan ses på tabellen virkede alle funktionerne efter hensigten, så Motorstyring var nu klar til integrationstesten.

2.2.2 I2C Kommunikation

For at teste I2C blev der sat to simple dokumenter op. Der blev først sat et PSoC-Creator dokument op, som bestod af en I2C-Slave, en LED og en read- og write buffer. I dette dokument blev der skrevet kode med inspiration fra datasheet[1], som skulle håndtere en dataoverførsel fra masteren på write bufferen og gemme disse data på en read buffer, så masteren ville kunne læse de afsendte data. På den måde ville det være muligt at observere, om det kunne lykkedes med at sende en data-

overførsel og læse den igen. Hvis den første byte i dataoverførslen var 0x1, ville en LED på PSoC'en tænde. Dette blev implementeret i et for-loop.

Det andet dokument blev sat op i Linux, og bestod, udover open() og ioctl() som er beskrevet i implementeringsafsnittet, af en write funktion samt write buffer, en kort pause og derefter en read funktion samt en read buffer. Disse to buffere blev så printet ud i terminalen. På den måde var det forventet, at de data som man sendte via write ville blive læst via read.

Der opstod dog det problem, at kun den første byte blev sendt korrekt, imens de næste to bytes altid blev sendt som 0 og 0. Efter mange test, bl.a. med brug af Analog Discovery, blev fejlen fundet og rettet - de to read og write buffere i linux dokumentet var int arrays, og eftersom int består af 32 bit, fungerede overførslen ikke, da der skulle sendes 8 bit af gangen. Da disse blev rettet til char arrays, fungerede I2C-Kommunikationen som forventet og ønsket.

2.2.3 Farvesensor

På samme måde som for motorsytringen, er der i farvesensor implementeret en UART. Dette gør, at der via et konsolvindue, kan udskrives hvilken farve, farvesensoren detekterer. Der oprettes 4 switch cases i main, og alt efter hvilket tal der returenes fra getColor(), udskrives den detekterede farve på konsolvinduet i RealTerm. F.eks. hvis sensoren detekterer rød, returneres '1', som resulterer i, at der udskrives 'RED' ude på konsolvinduet.

Farvesensoren testes ved at den pakkes i noget pap (Forklaring kan ses i Hardware implementering). Sensoren løftes ca. 3cm fra bordet og holdes stabilt. Der anvendes farvet karton til at teste de forskellige farver, og disse sættes ind under sensoren en efter en. Når sensoren ikke detekterer farverne rød, blå og grøn, skal den udskrive 'unknown'.

I tabel 2 er de fire cases opstillet. De fire cases skal alle testes igennem for, at have en funktionsdygtig farvesensor. Dette er nødvendigt for, da man skal kunne kende forskellen på disse farver for at navigere med robotten.

Case	Returværdi	Observering
'1' Red color	'1'	Når der sættes et stykke rødt pap under sensoren returneres '1' og **RED**.
'2' Green color	'2'	Når der indsættes et stykke grønt karton under sensoren returneres '2' og **GREEN**.
'3' Blue color	'3'	Når der indsættes et stykke blåt karton under sensoren returneres '3' og **BLUE**.
'4' Unknown color	'4'	Når der **ikke** er indsat noget karton under sensoren returneres '4' og **UNKNOWN**.

Tabel 2: colorSensorTest

Det kan ses, at testen af hver af de 4 cases resulterede i de forventede resultater. Dermed kan man se, at farvesensoren lever op til forventningerne om at kunne skelne mellem de tre farver, samt når ingen af farverne er registreret. Modulet er derfor klar til at blive testet i integrationstesten.

2.2.4 Scale

For at teste Scale programmet at det virker som forventet blev der testet ved hjælp af en UART, og en konsolvindue som kan udskrive værdien af ADC og vægten der konverteres om til. Ved hjælp af kalibreringen kan der omregnes direkte om til kg. og der skal ses om vægten er korrekt lavet om til kg. I RealTerm udskrives værdierne og ved hjælp af kendte vægte, som blev målt på forhånd med køkkenvægt sammenlignes værdierne.

Faktisk vægt	Returværdi	Observering
Vægt på 1 kg	0,9982 kg	1 kg blev placeret og den målte ADC værdi konventeres til kg.
Vægt på 0,502 kg	0,4977 kg	0,502 kg blev placeret og målte ADC værdi konventeres til kg.
Vægt på 0,301 kg	0,3017 kg	0,301 kg blev placeret og målte ADC værdi konventeres til kg.
Vægt på 0,156 kg	0,1507 kg	0,156 kg blev placeret og målte ADC værdi konventeres til kg.

Tabel 3: Scale Test

Efter at have testet vægten kom frem til vægten vejer som forventet. Blev der testet ved hjælp af en LED på PSoC, at ændringen af vægten kan tænde og slukke en LED. Ved at have en grænse på hvis vægten er under 300 g vil LED ikke lyse, hvor hvis den er over vil LED lyse.

2.2.5 RPi

For at kunne teste om RPi programmet virker som forventet er det blevet forbundet med hjemmesiden. Ved at have RPi'en tilsluttet en computer har vi haft mulighed for at følge med i en terminal og printe de forskellige data ud så vi kan følge med i at programmet arbejder som forventet. Her er alle 6 knapper blevet testet og det kan aflæses i terminalen at RPi koden håndtere beskederne korrekt og printer de forskellige bestillinger ud på terminalen afhængig af hvad der bliver valgt på hjemmesiden.
Efterfølgende er der blevet inkluderet en PSoC med i testdelen så det kunne afprøves om RPi koden kunne håndtere og sende de korrekte bestillinger igennem I2C. Her er der brugt det I2C som tidligere er blevet testet for sig selv. Dog er der blevet tiltøjet noget kode til PSoC'en så den vil blinke med dens indbykkede LED afhængig af hvilken bestilling der er blevet sendt.

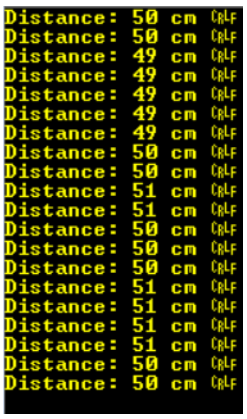
2.2.6 Afstandssensor

Afstandssensoren blev testet ved at tilslutte et oscilloskop og benytte et målebånd til at måle afstand. Først blev sensoren testet i stationær tilstand. Et objekt blev, ved at måle med et målebånd, placeret 50 cm væk fra afstandssensoren. Oscilloskopet viste tidsperioden hvor echo var høj. Denne tidsperiode

blev omregnet til en afstand med formel (gældende ved en stuetemperatur på $20C^o$):

$$afstand = tid(us) \times 0.034(cm/us)/2$$

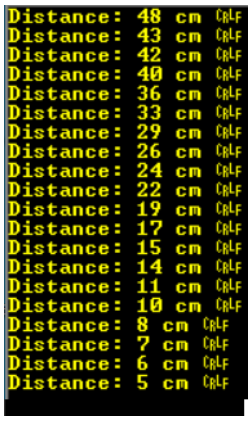
Den blev derefter sammenlignet med afstanden målt med målebåndet, som burde være ens. Terminalen blev også observeret samtidig med, som det ses på figur 2 .



Figur 2: Terminalvindue hvor afstanden vises

Som det ses på figur 2, er sensoren forholdsvis præcis, hvor den svinger med +/-2%.

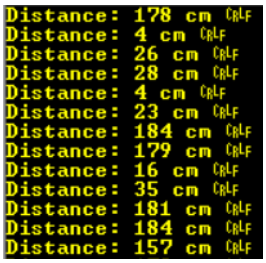
Til næste test testes sensoren i mobil tilstand. Samme opstilling blev benyttet som før, men i denne test benyttes målebånd ikke. Her var forventningen, at afstanden ændrede sig, som objektet bevægede sig tættere på. Objektet bevæges tættere på, mens der tages 20 målinger, som er efterfulgt af hinanden. Størrelsen på dem alle observeres i terminalen, og måles på oscilloskopet for bekræftelse, og de bør blive mindre og mindre.



Figur 3: Terminalvindue hvor afstanden vises, mens objekt er i bevægelse

Det ses i figur 3, at sensoren kan detektere og hurtigt nå at opdatere afstanden på objektet.

Der beskrives i datasheetet for HC-SR04, at sensoren er dårlig til at detektere bløde objekter. Sensorens følsomhed overfor disse testes derfor. Til testen benyttes en t-shirt som det bløde objekt, og holdes op ca. 25 cm væk fra sensoren.



Figur 4: Terminalvindue hvor afstanden for et blødt objekt vises

På figur 4 ses det, at sensoren har svært ved at registrere t-shirten. Til bagvæggen bag t-shirten er der ca. 180 cm, og det er denne væg, som sensoren i nogle tilfælde detekterer i stedet. Dette kan muligvis blive et problem, når Beertress kører rundt og har svært ved at detektere bløde objekter. En pose som ligger på banen, vil Beertress sandsynligvis køre lige over, og derved påvirke farvesensoren, som sender Beertress ud af kurs fra den opsatte bane.

2.3 Integrationstest

Først blev alt kode samlet i en main, hvorefter hvert moduls funktionalitet blev udkommenteret en efter en, så det blev muligt at teste hvert modul sammen med motoren. Til sidst blev alle modulerne udkommenteret, og der blev observeret, om Beertress kunne fungere i systemtesten. Integrationstesten kan ses i tabel 4.
Grunden til, at farvesensor og afstandssensor ikke blev puttet ovenpå I2C var, at det tog lang tid for at få forbindelse til hjemmeside. Efter at I2C fungerede, blev det vurderet, at der ikke var behov for at teste den igen før systemtesten.

Moduler under test	**Beskrivelse**	**Observering**
Motor, I2C og RPi	Sender 6 forskellige kommandoer via kundegrænsefladen, og observerer om Beertress gør 6 forskellige ting.	Beertress laver 6 forskellige ting, alt afhængig af modtaget ordre.
Motor, farvesensor og afstandssensor	Det blev testet om Beertress kunne navigere rundt på banen ved hjælp af farvesensoren, og samtidig testet for om afstandssensor virkede ved at sætte hånden ind foran	Beertress navigerer rundt på banen som ønsket. Når man satte hånden ind foran afstandssensoren stoppede Beertress, og kørte igen når man fjernede hånden.
Vægtsensor	Vægten blev testet ved at stille en øldåse på vægten og når den blev fjernet skulle LED på PSoC lyse op	Øldåsen blev fjernet og LED lyste som forventet.

Tabel 4: Integrationstest af de forskellige moduler

2.4 Systemtest

I systemtesten bliver det testet, om alle modulerne kan snakke sammen og fungere som en helhed. Eftersom denne prototype kun kan indeholde 1 øl, bliver der kun testes med en øl. Alle moduler blev udkommenteret, og der blev først via hjemmesiden sendt en ordre om 1 øl til bord 1 og derefter en øl til bord 2. Systemtesten kan ses i tabel 5.

Beskrivelse	**Observering**
Bestilling af 1 øl til bord 1	Bilen kører hen til bord 1 og stopper. Når øllen tages, kører Beertress videre til start og stopper. Når ny ordre sendes, kører Beertress igen.
Bestilling af 1 øl til bord 2	Bilen kører forbi bord 1 og kører videre til bord 2 og stopper. Når øllen tages, kører Beertress videre til start og stopper. Når ny ordre sendes, kører Beertress igen.

Tabel 5: Systemtest af Beertress

Litteratur

[1] *Datablad for I2C-slave komponent i PSoC.* `https://www.cypress.com/file/130946/download`.