

Angry Birds In Real Life: Et legetøj

Eksamensprojekt

Fag: Teknik A Design og Produktion

Klasse: Design og produktion TK 1

H.C. Ørsted Gymnasiet, Lyngby

Udarbejdet af:

Elev: Simon Rumle Tarnow Dato

Elev: Daniel Muff Laporte Dato

Elev:
Christoffer Irvall Rasmussen Dato

14. maj 2017

Resume

Vi har i forbindelse med vores eksamensprojekt i Teknik A (Design og Produktion) udarbejdet et legetøj kaldet Angry Birds In Real Life, der drager inspiration fra Angry Birds spillet udviklet af. I denne rapport dokumenterer vi produktudviklingen med hovedfokus på El-tekniske aspekter og programmering.

Vi har i forbindelse med vores undersøgelse af problemstilling kommet frem til at mange unge spiller meget videospil, dette forsøger vi at hjælpe på ved at gøre videospillet mere interaktivt og derved socialt, så det således er en lille ”angry bird”kanon. Af anvendt teknik benyttes der motorer til gearing og retningsbestemmelse af kanonen, farvesensor benyttes til at give forskellige farve projektiler forskellige hastigheder, samt programmering af Arduino. Vi kom frem til at de tekniske aspekter af vores produkt fungerede fint, men vi har en noget mangelfuld prototype i og med at mange af delene ikke er sat fast til selve kanonen.

Indhold

1 Læsevejledning	1
1.1 Afsnit	1
1.2 Bilag	1
1.3 Kilder	1
1.3.1 Websider	1
1.3.2 Bøger og skrifter	1
2 Projektbeskrivelse	2
2.1 Problemanalyse	2
2.2 Problemformulering	2
2.3 Projektafgrænsning	2
2.4 Overordnede produktkrav	4
2.5 Tidsplan for projektet	5
3 Overordnet løsningsforslag	5
3.1 Gauss-kanon	5
3.1.1 Magnetisk løb	6
3.1.2 Magnetisk aftrækker	7
3.2 Elastik-kanon	7
3.3 Valg af overordnet løsning	8
4 Samlet kredsløb	9
4.1 Hastighedsmåler	9
4.2 Farve sensor	9
4.3 Controller	9
4.4 Motorkontrol	9
5 Retningsregulerende kreds	10
5.1 Komponenter	10
5.1.1 N-Channel power MOSFET - F12N10L	10
5.1.2 Stepper motor - RS191-8328	11
5.1.3 Arduino	11
5.2 Teori	11
5.2.1 MOSFET	11
5.2.2 Stepper motor	12
5.3 Test	12
6 Hastighedsregulerende kreds	13
6.1 Komponenter	14
6.2 Lego 9 V DC motor	14
6.3 Dual H-bridge motor driver - L293D	14
6.4 Arduino	14
6.5 Teori	14
6.5.1 H bridge	14
6.6 Test	15

7 Hastighedsmåler	15
7.1 Komponenter	17
7.1.1 IR afsender diode - L-34F3BT	17
7.1.2 IR modtager diode - BPW 34 FA	17
7.1.3 555 timer - NE555P	18
7.1.4 NPN-transistor - BC547	18
7.1.5 Signal invertering - HEF4001B	19
7.1.6 Arduino	19
7.2 Teori	20
7.2.1 Benyttelse af diode som variabel spænding	20
7.2.2 Nor-gate	20
7.2.3 Hastighedsmålingsmetoder	20
7.3 Beregninger	20
7.3.1 555 timer modstande	20
7.3.2 Bestemmelse af modstande for peak-detektoren	22
7.4 Test	23
7.4.1 Afsender dioden	23
7.4.2 Modtager dioden	26
8 Controller	27
8.1 Komponenter	27
8.2 Teori	27
9 Display	28
9.1 LCD-kredsløb	28
9.1.1 LCD setup	28
9.2 Komponenter	28
9.2.1 LCD display	28
9.2.2 Arduino	29
9.3 Teori	29
9.3.1 LiquidCrystal library	29
9.4 Test	30
10 Ladningssensor	31
10.1 Komponenter	31
10.1.1 OP Amp - MCP601	31
10.1.2 BPW21 - Fotodiode	32
10.1.3 LED - RGB WHITE DIFFUSED LENS COMMON CATHODE	33
10.1.4 PNP transistor	33
10.1.5 Arduino	33
10.2 Teori	34
10.2.1 OP Amp	34
10.2.2 Analog input til registrering af farve	35
10.3 Beregninger	35
10.3.1 Modstanden af LED	35
10.3.2 OPAMP modstande	37
10.4 Test	38

11 Arduinoen og programmering	40
11.1 Arduino som komponent	40
11.1.1 Analog inputs og outputs (PWM)	40
11.1.2 Digital inputs og outputs	41
11.1.3 Portmanipulation og registrer	41
11.1.4 I2C - Synkroniseret kommunikations bus	43
11.2 Kort beskrivelse af programmets formål	43
11.2.1 Master Arduino	43
11.2.2 Controller Arduino	44
11.3 Oversigt over inputs og outputs og lokation	44
11.4 Kort beskrivelse af funktioner	45
11.4.1 Master Arduino - Loop()	45
11.4.2 Master Arduino - getinput()	46
11.4.3 Master Arduino - fire()	47
11.4.4 Master Arduino - determinespeed()	48
11.4.5 Master Arduino - determinecolor()	48
12 Fremstilling	48
12.1 Projekttiler	48
12.2 Fumlebræt-modeller	49
12.3 PCB - fremstilling	49
12.3.1 Kemisk fremstilling	50
12.3.2 Problemer med printboards	50
12.4 Endelige prototype	50
13 Slut afprøvning	51
14 Videreudvikling	51
15 Prisliste	51
15.1 Produktionsomkostninger	51
15.2 Estimering af udsalgspris	51
15.3 Konkurrencedygtighed	52
16 Konklusion	52
17 Evaluering	52
Referencer	52
A Samlet kredsløb	56
B Endelig prototype - kredsløb	57
C Endelig prototype - kanonkonstruktion	58
D PCB artwork til hastighedssensor	59
E PCB artwork til arduino shield	60

F Fumlebrætmodel anden del	61
G Program til Master Arduino	62
H Program til Controller Arduino	70
I Logbog	75

1 Læsevejledning

1.1 Afsnit

Overordnede afsnit inklusiv produkt-blokke-afsnit gives et nummer højere end tidligere overordnede afsnit. Alle underafsnit markeres med det overordnede nummer til det afsnit det tilhører med en punktum-indeksering som fungerer hierarkisk. Så f.eks. afsnit ”Projektbeskrivelse” vil blive markeret med ”2”. Hvor deraf et underafsnit som ”Problemanalyse” vil blive markeret ”2.1”. Afsnit der er sideordnede med ”Problemanalyse” og underordnede med ”Projektbeskrivelse” vil så blive markeret ”2.2”, ”2.3”, ”2.4” etcetera.

1.2 Bilag

Bilag er til forskel fra normale afsnit i rapporten angivet med et stort bogstav i titlen, således at hvis det skrives ”Se i bilag A” vides det at man skal se sidst i vores indholdsfortegnelse hvor der er et bilag indikeret med A i begyndelse. I PDF-format kan man trykke på references for at blive taget direkte til bilaget. Jo større i omfang bilagene er desto senere i rapporten placeres de. Så blandt vores sidste bilag er programkoden til arduinoen og vores logbog.

1.3 Kilder

Kilderne er nummereret efter alfabetisk rækkefølge. Kilder er refereret til i teksten ved at de er nummereret og med kantede parenteser som f.eks. [1]. Kilderne kan klikkes på i PDF-format for at blive taget til kilden direkte til litteraturlisten. Der refereres forskelligt afhængig af hvilken type kilde det er:

1.3.1 Websider

Først indikeres firmaet eller forfatteren af kilden, dernæst overskriften på afsnittet kilden har. Derefter er der et hyperlink til URL med angivelse af dato sidst set. Til websider, som ikke er links til datablade, gives der kort kildekritik.

[Forfattere]. [Overskift]. URL: [link] ([dato sidst set]). [Kildekritik]

1.3.2 Bøger og skrifter

For bøger og andre skrifter. Indikeres først forfattere, derefter titlen på værket, efterfulgt af forlagets adresse og forlaget. Der knyttes evt. noter til værket så som hvilke sider vi har benyttet.

[Forfattere]. [Titel]. [Forlag adresse]: [Forlag]. [Noter]

2 Projektbeskrivelse

2.1 Problemanalyse

I det moderne samfund bliver TV og iPads mere og mere inddraget i børns opvækst. Ifølge en undersøgelse af Northwestern University - Center of Human Development¹, er 27% af alle amerikanske familier *media-centric*, hvilket betyder at disse familier benytter en stor del af deres tid foran en digital skærm af en eller anden form, dette indebærer også deres børn.

På trods af denne gængse tendens for IT-brug, udviser forældre en generel nervøsitet vedrørende konsekvenserne af børn (under 8 år) forbrug af digitale medier, herunder specielt computerspil. Ifølge undersøgelsen af Northwestern University, er forældre mest af alt nervøse om børns fysiske helbred og sociale evner som konsekvens af meget brug af computerspil på iPads og smart phones. Selvom videnskaben om konsekvenserne af forbrug af computerspil stadig er ret tvivlsomt og på et tidligt forskningsstadie, er det en relevant problemstilling.

2.2 Problemformulering

Det er et samfundsmæssigt problem at børn har svære ved at socialiserer sig pga. de bruger for lang tid på smartphone apps.

2.3 Projektafgrænsning

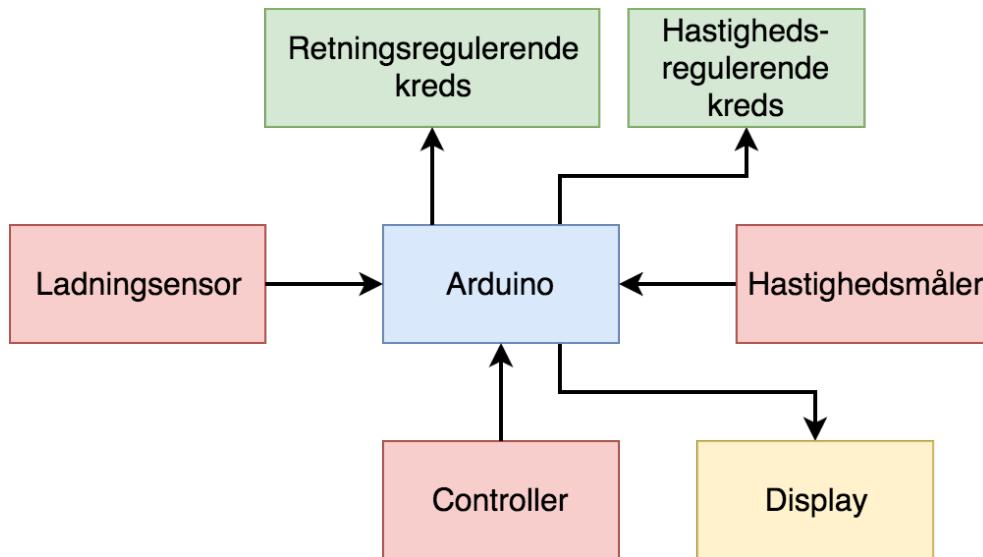
For at specificerer en løsning til denne problemstilling tages der udgangspunkt i videospillet Angry Birds af Rovio Entertainment. Ifølge Michael Chorost beskrevet i Psychology today² er der 4 overordnede grunde til at Angry Birds er let at blive afhængig af:

- Det er simpelt, ingen ”learning curve”.
- Det er en primitiv nydelse i at destruerer ting.
- Selve fysikken i spillet virker realistisk og forudsigeligt.
- Det er sjovt. Dydrene i spillet laver backflips og siger sjove lyde.

Vi har således tænkt os at udforme en fysisk udgave af et Angry Birds lignende spil, hvor børn kan interagerer socialt ved at spille mod hinanden i stedet for at sidde foran deres iPad. For at omgå copyright kalder vi vores produkt *Moody Feathercreatures*.

¹Northern University; Parenting in the Age of Digital Technology - A national survey; revised 2014

²Chorost, Michael How I kicked my addiction to the iPhone game Angry Birds; <https://www.psychologytoday.com/blog/world-wide-mind/201101/how-i-kicked-my-addiction-the-iphone-game-angry-birds>; 2011



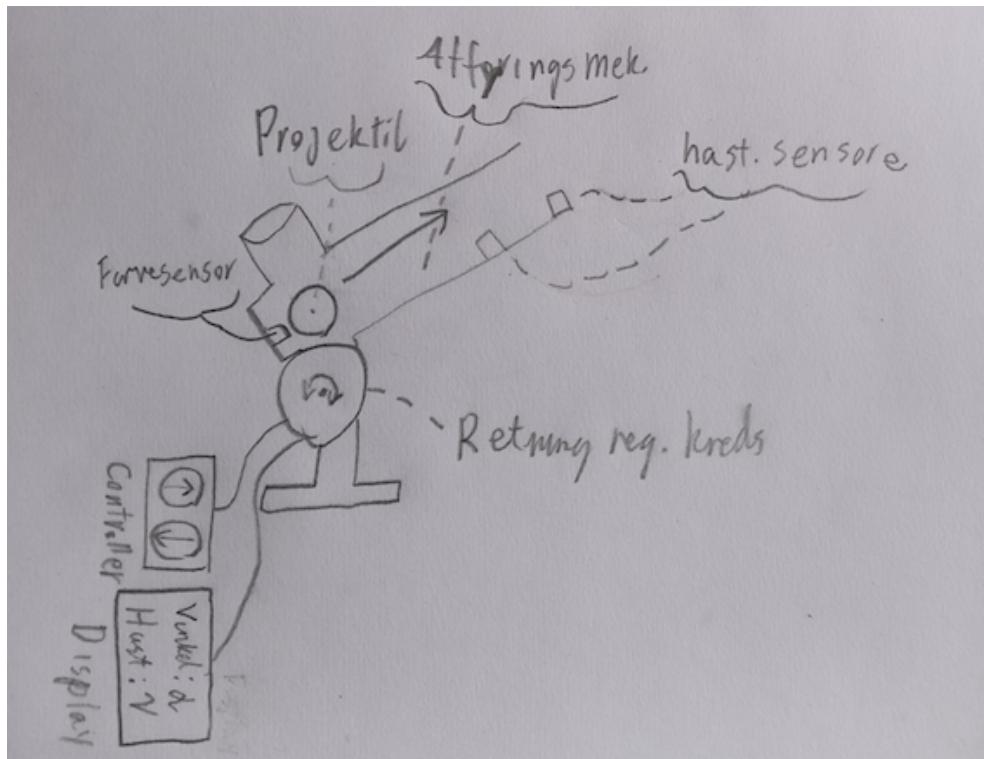
Figur 1: Blokdiagram af overordnet løsning. Input blokke(røde), Output bloke (gule), Kredse (grønne)

Vi har opstillet et overordnet blokdiagram for hvordan vi har tænkt os at efterligne spillet. Se Figur 1.

Vi har valgt at lave et produkt der kan skyde nogle paptårne fra hinanden. Den skal også kunne justerer hastigheden på projektilet afhængigt af projektilets farve (ligesom farverne på fuglene i Angry Birds) og man skal kunne bruge en form for controller til at justere kanonens vinkel i forhold til vandret. Se Figur 2.

Som man kan se på Figur 1 er blokkende inddelt således:

- Input blokke
 - Ladningssensor
 - * Denne del skal kunne se på farven af et indsat projektil og derfra sende det til arduinoen.
 - Controller
 - * Til at bestemme retningen på selve ”kanonen”.
 - Hastighedsmåler
 - * For at kunne måle hastigheden ved udgangen af ”kanonen”.
- Output bloke
 - Display
 - * Til at vise point og hastighed af projektilet. Muligvis andet.
- Kredse



Figur 2: En overordnet skitse af produktet

- Hastighedsregulerende kreds
 - * Benyttes til at regulerer hastigheden af projektet afhængigt af dens farve.
- Retningsregulerende kreds
 - * Benyttes til at bestemme retningen alt efter inputtet fra controlleren.

2.4 Overordnede produktråd

Da vores produkt er beregnet til børn og unge, skal produktet være sikkert at bruge for børn og unge. Dette betyder at vores "kanon" ikke må skyde hårdt nok, til at volde skade på børn og unge. Derudover skal projektillet "kanonen"skyder, ikke være skarpt eller meget hårdt, da der er risiko for at børnene, ved en fejltagelse, skyder på hinanden. "Kanonen" og projektillet skal være i stand til at vælte papboksene ned. Vi skal sikre os as vores "kanon" og projektil ikke er i strid med den danske våbenlov, og eventuelt udenlandske våbenlove. Vores produkt skal altså opfylde følgende overordnede produktråd:

- Sikkert at bruge for børn.
- Projektilerne skal ikke have en farlig form.
- Produktet skal skyde hårdt nok til at vælte små papbokse.
- Produktet skal ikke være i strid med våbenloven.

2.5 Tidsplan for projektet

uge nr.	10	11	12	13	14	16	17	18	19	20
Projektbeskrivelse										
Ladningsensor										
Retningsregulerende kreds										
Hastighedsregulerende kreds										
Hastighedsmåler										
Display										
Controller										
Material udformning										
Arduino program										
Printkort										
Rapport										

Figur 3: Tidsplan

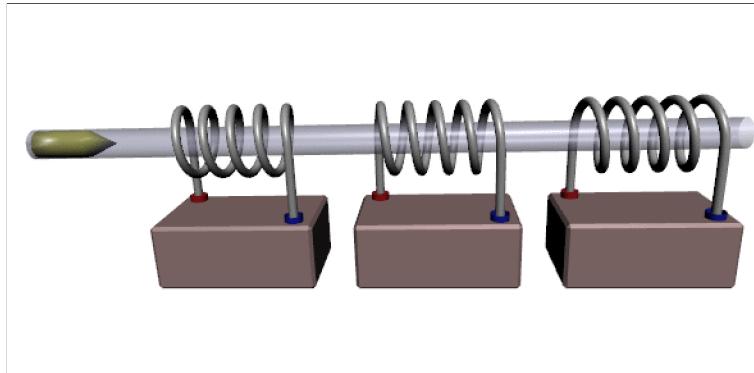
3 Overordnet løsningsforslag

Vi har gennem vores idegenerering kommet frem til overordnet løsningsforslag. Vi kom frem til at den mest kompliceret dele af kredsen som er vigtigst at designe, og have på plads fra begyndelsen er affyringsmekanismen. Heraf har vi kommet frem til to typer afføringsmekanismer: *Gauss-kanon* og *Elastik-kanon*. Som overordnet løsning til hastighedssensoren har vi tænkt os at se på om vi kan udforme et pass band filter og IR lyd til at sanse hvornår bolden kommer forbi.

3.1 Gauss-kanon

Her har vi tænkt os at benytte principippet om magnetfelter i spoler til at drive et projektil fremad. Heraf kan vi ved at variere i den tilførte strømstyrke og spænding for at ændre magnetfeltet. Ud fra vores brainstorm har vi to måder at udforme sådan en affyringsmekanisme: Magnetisk løb og Magnetisk aftrækker

3.1.1 Magnetisk løb



Figur 4: Et billede af en gausskanon, hvor projektillet bliver accelereret af tre spoler. Kilde: [35]

For en enkelt spole, kan et projektils udgangshastighed modelleres udtrykket udformet af PhD i anvendt matematik Don Pettibone (se [11]).

$$v_{slut} = \frac{\mu_0 \cdot N_0 \cdot I_0}{2 \cdot r_0} \cdot \sqrt{\frac{2 \cdot \mu_r - 1}{\rho \cdot \mu_0}}$$

- v_{slut} : Sluthastigheden ved enden af spolen for projektillet [$\frac{m}{s}$]
- μ_0 : Vakuumpermeabiliteten ($\mu_0 = 1.257 \cdot 10^{-6} \frac{T \cdot m}{A}$).
- I_0 : Strømstyrke [A].
- μ_r : Relativ permeabilitet af projektilets materiale.
- r_0 : Radius af spolen [m].
- N_0 : Antallet af vindinger for spolen.

Dog er modellen meget optimistisk og i et reelt eksempel (se [20]) udført af elektroingenøren Mehdi Sadaghdar så vurderer vi at vi kan let komme til at arbejde med effekter på omtrent:

$$P \approx 2000 \text{ W}$$

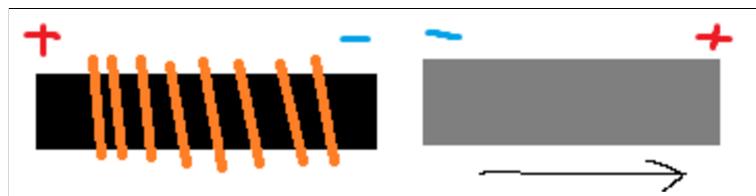
Måder vi kan holde os fra så store effekter er at benytte jernkerne, for at øge den relative permeabilitet og magneter. Dette er svært at implementere i en Magnetisk løb løsning, idet spoler med jernkerne sjældent har et hul et projektil kan passere. Det er dog lettere at få implementeret i Magnetisk aftrækker løsningen.

En anden måde at få implementeret Magnetisk løb løsningen ved brug af lavere effekter er at benytte et par (knap så lange) solenoider, for at danne en længere spole, således

benytte en kombination af sensorer og transistorer til at slukke for den tidligere spole og tænde den næste for at sikre sig at den bliver ved med at accelerer undervejs.

3.1.2 Magnetisk aftrækker

Den Magnetiske aftrækker fungerer ved at placerer en magnet tæt ved en slukket spole. Hvis magneten berøre spolens ende der har en tilsvarende ladning vil de afstøde hinanden og således affyre projektet. Se Figur 5.



Figur 5: Et billede af en spole på venstre side, der benyttes til at affyre et magnetisk projektil på højre side.

Denne løsning er dog svære at matematisk modellere, men med sikkerhed må det gælde at (se [15] - side 129), hvis der benyttes jernkerne

$$B = \mu_r \cdot \mu_0 \cdot \frac{N_0 \cdot I_0}{l}$$

Hvor B er magnetfeltstyrken [T] og l er længden af spolen [m]. Her fremgår det at hvis μ_r bliver meget høj (hvilket den kan blive med jernkerne), vil magnetfeltet stige proportionalt. Med passende legeringer kan man få en relativ permeabilitet på op til 15 000.

Ulempen ved denne løsning er dog at der nok skal være ret så høj en effekt der skal sendes gennem spolen indenfor et kort tidsrum for at få det til at virke.

3.2 Elastik-kanon

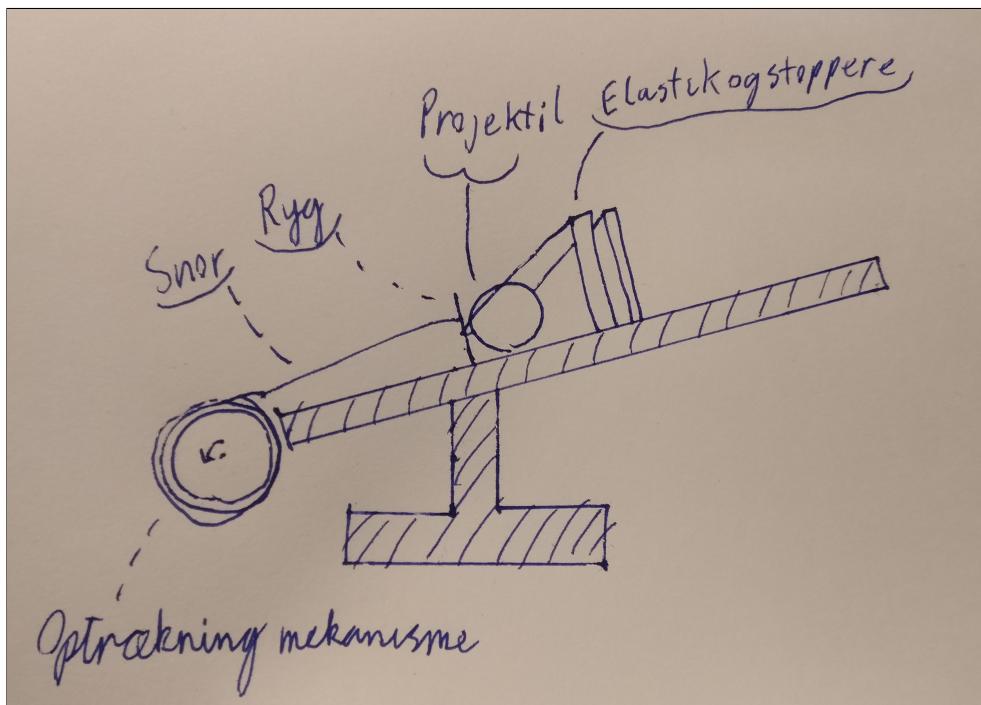
Denne løsning bliver vores produkt udformet som en slangebøsse med variabel styrke. Således bliver den variable: hvor langt væk elastikken trækkes væk fra hviletilstand. En model for dette princip kan simpel konstrueres ud fra energiomdannelse fra potentiel energi i Hooks lov til kinetisk energi.

$$\begin{aligned} -k \cdot x &= \frac{1}{2} \cdot m \cdot v^2 \\ \Leftrightarrow v &= \sqrt{\frac{-2 \cdot k \cdot x}{m}} \end{aligned}$$

- k : Fjederkonstanten for benyttet elastik [$\frac{N}{m}$]
- m : Massen af projektilet [kg]

- v : Udgangshastigheden af projektillet [$\frac{m}{s}$]
- x : Afstand fra hviletilstand [m]

Vores implementering af en elastik-baseret løsning er at vi benytter os af en snor der er forbundet til elastikken og et hjul, hvor hjulet drejes af en motor (se Figur 6). For at sikre os at projektillet bliver affyret benytter vi gear, så når optrækning-mekanisme trækker snoren er hjulet i gear, hvorimod når vi skal affyre projektillet sættes den i frigear. Dette kan gøres ved at lade en motor være den der trækker snoren op, og en anden motor være den der ændrer gear.



Figur 6: Affyringsmekanisme for en Elastik-kanon, der er ikke fokus på andre blokke beskrevet i projektbeskrivelse.

3.3 Valg af overordnet løsning

Vi har valgt at udarbejde en Elastik-kanon, idet vi synes hvor får en del EL-problemer at se til i forhold til hastighedssensoren. Samt hvis vi skulle udarbejde en Gauss-kanon, ville vi bruge lang tid på at teste for vakuumpermeabilitet og benytte høje strømstyrker, hvilket i sig selv kunne give en del problemer, da vi ikke ved meget om at arbejde med høje strømstyrker. Samt ville man kunne argumentere for at så høje strømstyrker som benyttes i forbindelse med en Gauss-kanon kan muligvis være for farligt til et børnelejetøj.

4 Samlet kredsløb

Det samlede kredsløb kan findes i Bilag A. Vores kredsløb er delt op i mange forskellige delkredsløb som er med til at styre vores kanon. Her har vi 3 motorer til at styre fysiske bevægelser. Hertil har vi en farve sensor og infrarøde sensorer til at læse projetelets hastighed og farve. Her bruges arduinoen til at kordinere disse delkomponenter.

4.1 Hastighedsmåler

Her er der brugt en 555-timer til at lave en puls, som sendes til modtager delen som så forstærker signalet og der bliver brugt en peak detector far at få et stabilt signal der så kan sendes til arduinoen.

4.2 Farve sensor

Denne del fungere ved at der er 3 lysdioder. en rød, en grøn og en blå. ardinoen giver et signal om hvilken diode der skal være tændt hvorved at en phototransistor opfanger det lys der bliver reflekteret dette give så forskellige værdier i forhold til hvilken farve der reflekterende objekt har. dette signal bliver sendt tilbage til arduinoens analog input.

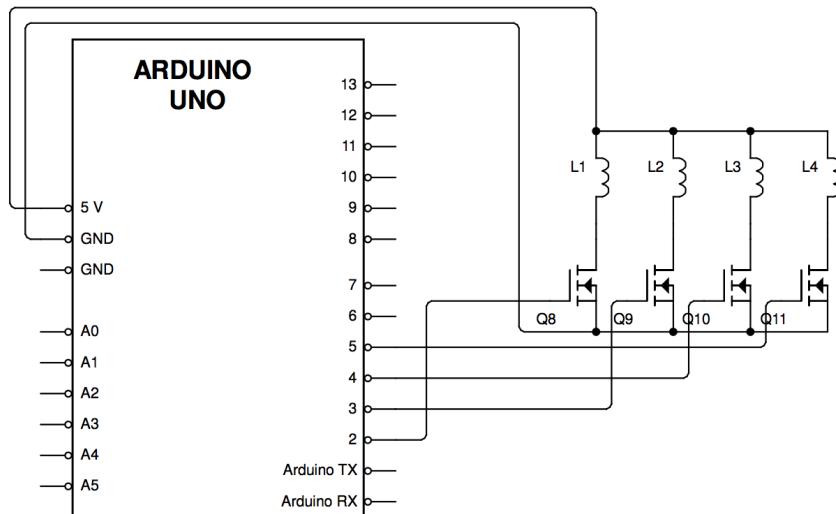
4.3 Controller

Inputet er bare et par knapper som går direkte ind i arduinoens digitale porte. På samme måde bliver displayet også for det meste tilkoplet direkte til arduinoen over en enkelt 10k potentiometer som bruges.

4.4 Motorkontrol

Vi har to forskellige motore til at styre vores kanon. en til at dreje den op og ned og 2 til at styre dens skyde mekanisme. Den ene bliver brugt til at hive i elastikken mens den anden er en trigger som holder tandhjulene sammen som giver mulighed for at motoren elastikken bevæger sig frit for motoren.

5 Retningsregulerende kreds

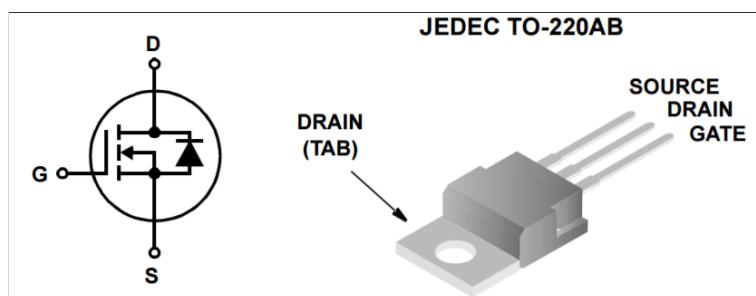


Figur 7: Et billede af kredsløbet for den retningsregulerende kreds.

I vores retningsregulerende kreds, har vi valgt at benytte en stepper motor, til at styre hvilken vinkel bolden skydes ud i. Her der bliver der brugt en ekstern strømforsyning på 5 V, så der kan løbe nok strøm igennem spolerne i stepper motoren. For at bruge den eksterne strømforsyning bruger vi MOSFETs som digitale switches.

5.1 Komponenter

5.1.1 N-Channel power MOSFET - F12N10L



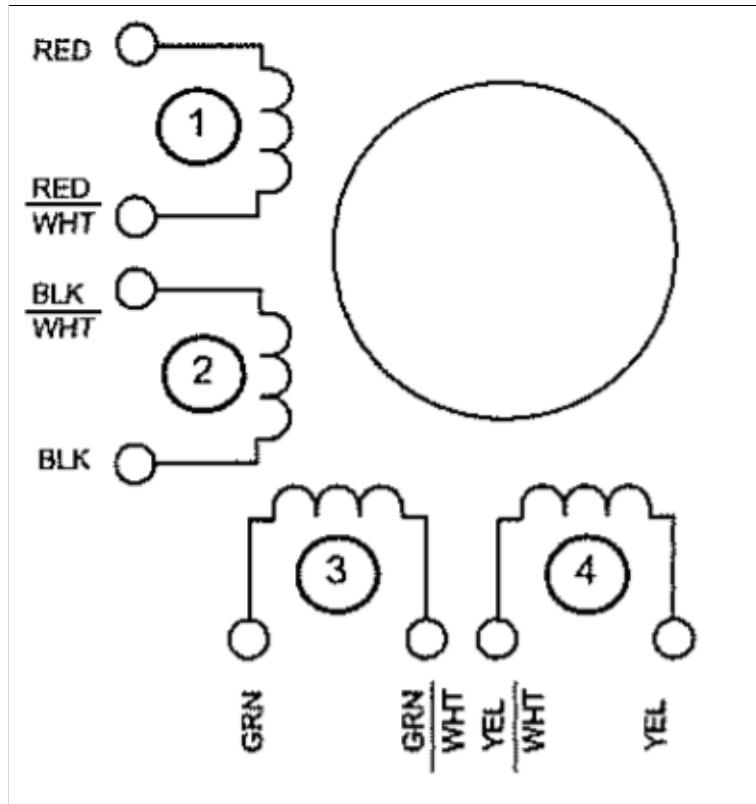
Figur 8: Pindiagram og symbol af F12N10L. Kilde:[14]

På Figur 8 er der et symbol og pindiagram over MOSFET komponenten. Denne MOSFET er bygget til 5 V logik, samt har det en lav rise og fall time på et par hundrede nanosekunder

lav kred-
sen om
igen,
således
at
dioden
over mos-
fettet
er
med,
se
MOS-
FET

og således vil det fungerer fint for en steppermotor. Databladet vi har benyttet kan findes i kilde [14].

5.1.2 Stepper motor - RS191-8328



skriv om spændingsreguleret - derfor er det godt med arduino

Figur 9: Diagram af RS191-8328. Kilde: [28]

5.1.3 Arduino

Se afsnit 11.

Har brug for noget tekst under billede

5.2 Teori

5.2.1 MOSFET

Vi benyttede MOSFET for at få en højere strøm igennem spolerne end Arduinoen kan leve. Med en MOSFET kan man kontrollere hvor meget strøm der løber gennem Gate til Source, med spændingsfaldet over Drain og Source. Dette gør en MOSFET optimal som en digital switch.

5.2.2 Stepper motor

Vi benyttede en stepper motor til at styre hvor meget vi drejer kanonen. En stepper motor fungere ved at vi har et vis antal ”steps” på en omdrejning. Man kan sende strøm gennem en af spolerne, som så vil trække stepper motoren et ”step” frem eller tilbage. Man sender så skiftevis strøm igennem spolerne, for at få stepper motoren til at forsætte i en retning. Vores stepper motor har 200 ”steps” på en omdrejning.

5.3 Test

Vi benyttede et Stepper library fra firmaet Arduinos hjemmeside[6]. Koden kan ses på Figur 10.

Figur 10: Kode til steppermotor for at angive antal steps der skal roteres

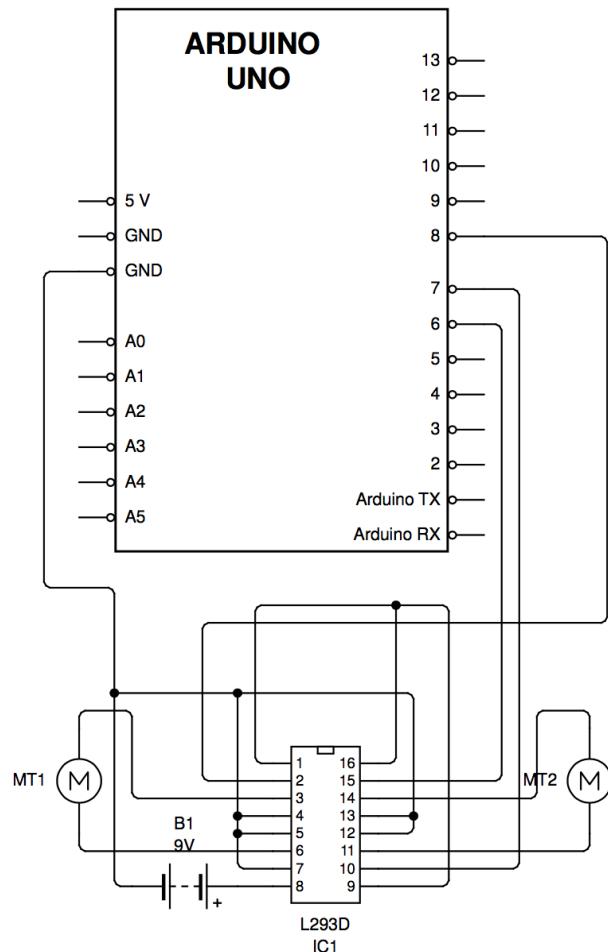
```
1 #include <Stepper.h>
2 // Antallet af steps på vores motor
3 const int stepsPerRevolution = 200;
4
5 // Initialiserer stepper biblioteket i pin 2 til 5:
6 Stepper myStepper(stepsPerRevolution, 2, 3, 4, 5);
7
8 // Antallet af steps motoren har taget
9 int stepCount = 0;
10
11 void setup() {
12 //Initialiserer serial porten
13   Serial.begin(9600);
14 }
15
16 void loop() {
17   // Step et step:
18   myStepper.step(1);
19   Serial.print("steps:");
20   Serial.println(stepCount);
21   stepCount++;
22   delay(500);
23 }
```

Husk
at an-
give
Uni-
polar
eller
Bipo-
lar

References
fun-
gerer
meget
under-
ligt.
Burde
sige
Figur
10

Det fungerede fint efter hensigten og vi kunne let styre antallet af steps

6 Hastighedsregulerende kreds



Figur 11: Et billede af kredsløbet for den hastighedsregulerende kreds.

Denne del af kredsen benyttes til at sætte elastikstrammeren i gear og frigear og stramme den op.

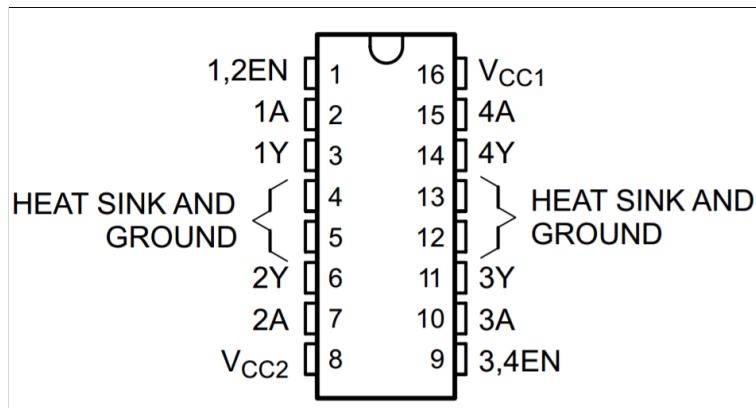
Som der kan ses på Figur 6, er der motorene MT1 og MT2. Heraf gælder det at MT2 fungerer som et gear og MT1 fungerer som selve elastik-optrækkeren. Når motoren trækkes op løber der først strøm fra Arduinoens pin 6 hvorfra gennem L293D pin 14 kan strøm true? løbe gennem MT2 og låse gearet fast. Derefter sendes der strøm gennem Arduinoens pin 8 som trækker MT1 op. Til sidst slukkes for signalet til MT1 og derefter ændres polariteten i MT2, så affyringsmekanismen er i "frigear". Det skal således bemærkes at MT2 modtager 2 signaler fra arduinoen for at kunne vende polariteten.

6.1 Komponenter

6.2 Lego 9 V DC motor

Vi kunne ikke finde et specifik datablad, men vi ved at normal lego-mindstorm DC motor der kører på 9 V, hvilket er fint til vores formål, men gør at vi har brug for 9 V batterier.

6.3 Dual H-bridge motor driver - L293D



Figur 12: Pindiagram af L293D

L293D er en dual H-bridge motor driver. H-bridges fungerer som beskrevet i teoriansnittet 6.5.1. Pindiagram kan ses på Figur 6.3. Pin 1 og 9 er aktiverings pins for H broerne. Pin 1 aktiverer H broen på venstre side og pin 9 aktiverer H broen på højre side. Pin 2 og 7 bruges til at styre motoren koblet til 3 og 6. Pin 10 og 15 styre motoren på 11 og 14. Pin 4,5,13 og 12 er forbundet, og skal forbindes til jord. Datablad kan findes i kilde nr. [36].

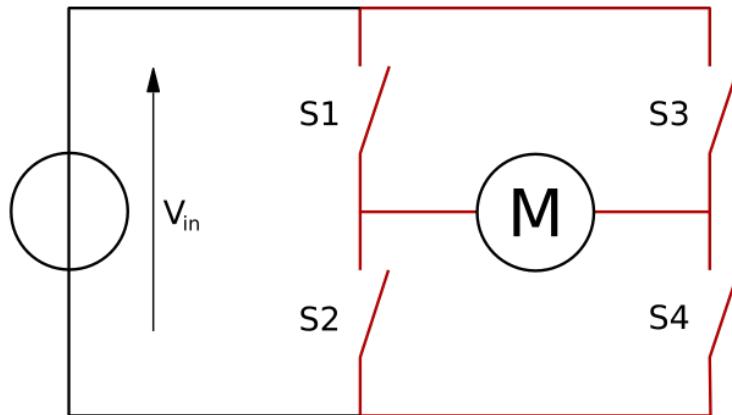
6.4 Arduino

Se afsnit ??.

6.5 Teori

6.5.1 H bridge

En H-bridge er et komponent der benyttes til at vende polariteten i vores DC-motor som fungerer som gear. Dette gøres overordnet ved at H-formede kredse med switches der kan enten være on eller off.



Figur 13: Et billede af en H-bridge, selve H-bridge struturet er markeret med rød. Kilde: [32]

Det fremgår på Figur 13 at switchene 1 til 4 er åbne. Disse switches kan så hvis de modtager et signal kan man forbides således at strømmens retning løber i en bestemt retning. F.eks. Hvis S1 og S4 er lukkede switches, vil motoren løbe i en retning, end hvis S3 og S2 er lukkede switches. Kilde: [32].

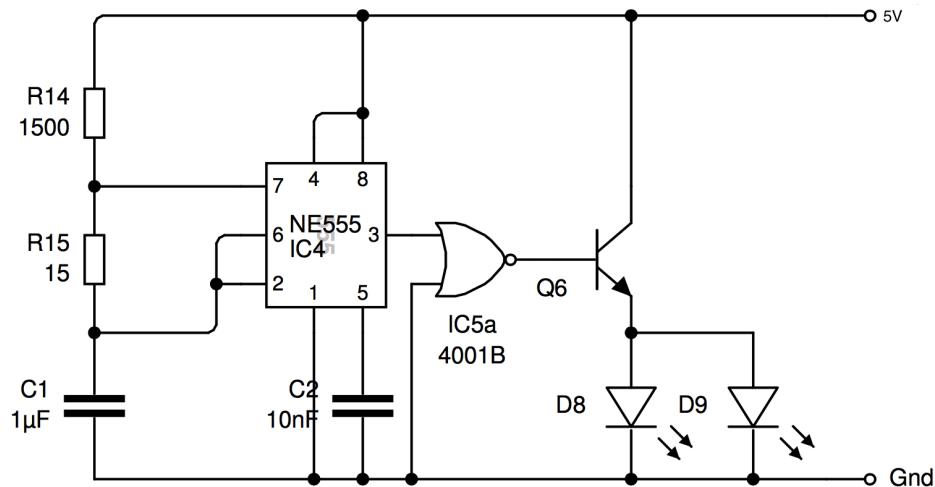
6.6 Test

Da denne kreds var ret simpel testede vi den ikke. Vi testede den dog i Slutafprøvning afsnittet på side 51.

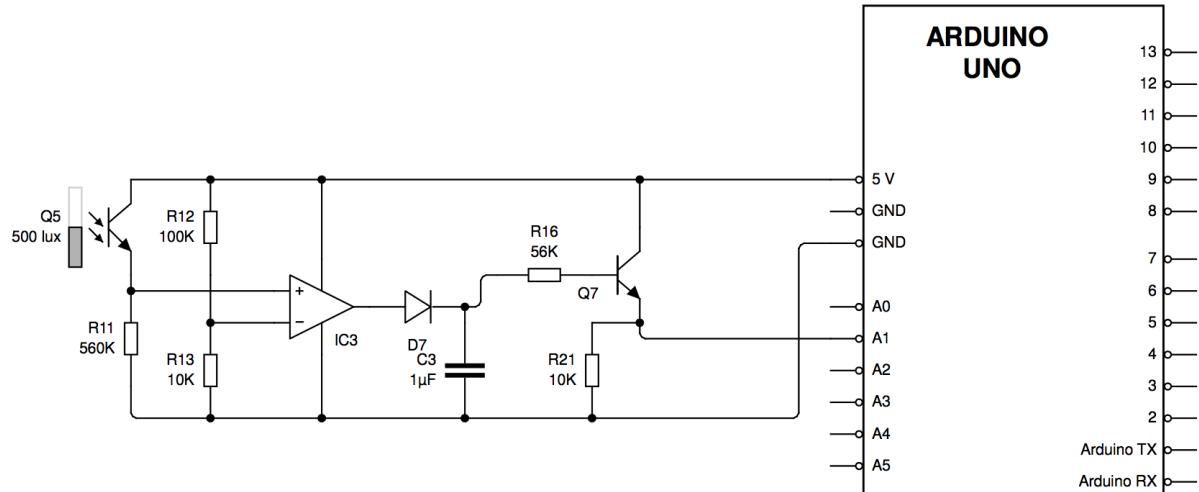
7 Hastighedsmåler

Vi skal not beskrive hvordan vi testede denne kreds og have noget kode på det.

Vi skal have skrevet til IR



Figur 14: IR-afsender



Figur 15: IR-modtager

På Figur 14 modtager en 555 timer strøm fra en 5 V strømforsyning, hvor vi har justeret modstandende til at give en frekvens på 1000 Hz og en duty cycle tæt på 99%, heraf bliver signalet inverteret, hvor duty-cyclen således bliver tilnærmelsesvis 1% (mere om valget af frekvens og dutycyclen omtalt i afsnit 7.1.1 og 7.3.1). Dette signal benyttes så til at føre en strøm ind til IR-dioder (D8 og D9), hvor vi styrer det med en NPN transistor.

Se på 15. IR-dioder lyser således på en IR-modtager diode (indsat reverse biased, markeret Q5), heraf benyttes først en comparator til at forstærke signalet når differensen

melleml + og - indgangen er positiv og dernæst benyttes der en peakdetector til at opfange hvornår signalet er der omtrent ved 4 V og fastholder det output, som så registreres af arduinoen.

7.1 Komponenter

7.1.1 IR afsender diode - L-34F3BT

Relevante størrelser på dioden [17] er bestemt ud fra absolute maximum ratings ved en temperatur på 25 °C Duty cyclen er

$$D = \frac{1}{100}$$

med en puls længde på

$$PW = 10 \mu s$$

Dette betyder at KingBright har vist at dioden kan med sikkerhed klare en frekvens på

$$f = \frac{D}{PW} = \frac{\frac{1}{100}}{10 \mu s} = 1000 \text{ Hz} \quad (1)$$

Hvilket vi vil forsøge at tilnærme.

Samt kan den klare en strømstyrke på

$$I_{FS} = 1.2 \text{ A}$$

7.1.2 IR modtager diode - BPW 34 FA

Relevante størrelser for IR modtager dioden [30] er at den har en kort switching tid på

$$t = 20 \text{ ns}$$

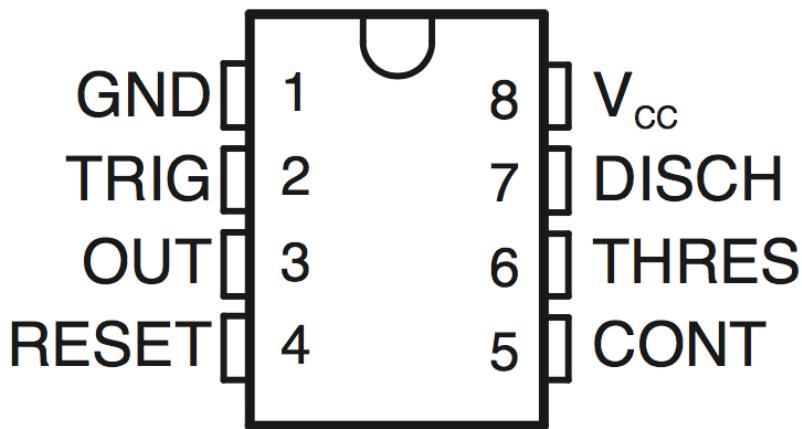
Den er bygget til at opfange lys med frekvensen på

[730 nm; 1100 nm]

Hvilket er fint for vores formål med infrarød lys.

7.1.3 555 timer - NE555P

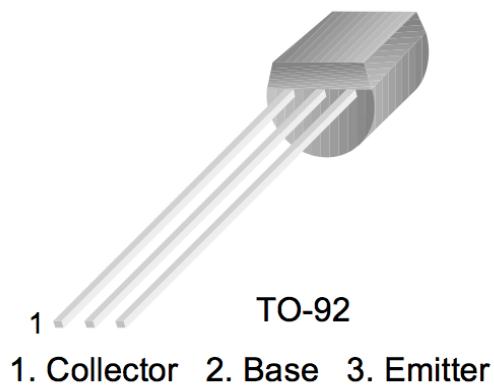
Vi benyttede en 555 timer[33] til at producere ønskede signaler med en frekvens på 1000 Hz, som beskrevet i ligning 1. Problemet ved at benytte en normal monostabil 555 timer er at duty cycle aldrig kan komme under 50% (kilde: [12]), dette betyder at vi ikke kan producerer korte signaler som ønsket, ved direkte at udvælge bestemte modstande. Derfor udnytter vi at duty cyclen kan sættes væsentligt højere op (nær 99%) og vi kan derfra inverterer signalet. Beregninger for udvælgelse af korrekte modstande for at få en ønsket duty cycle kan findes i afsnit 7.3.1.



Figur 16: Pindiagram over NE555P

7.1.4 NPN-transistor - BC547

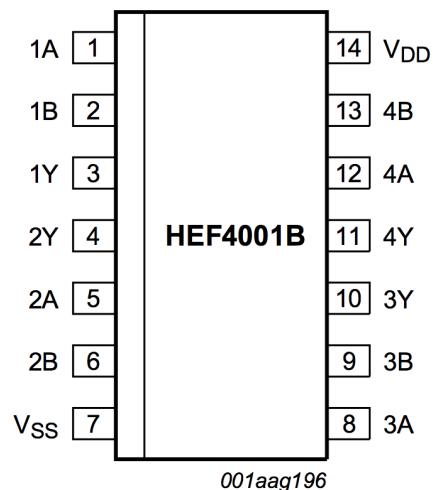
Vi benyttede en NPN transistor til at forstærke signalet, da man ikke kan trække en særlig høj strøm fra en NOR gate. Vi ved dog at når der kommer over de 0.7 V fra Base af transistoren, vil strømmen løbe fra collecter til emitter. Den er således ”mættet” eller ”aktiv”. Et pindiagram kan findes på Figur 17. Databladet kan findes i kilde nr. [13].



Figur 17: Fra venstre mod højre 1. Collector, 2. Base, 3. Emitter. Kilde:[13] - side 1

7.1.5 Signal invertering - HEF4001B

Kilde: [26] Vi har benyttet komponenter HEF4001B som er en IC med OR gates, som vi benytter til at inverterer signalet fra 555-timeren.



Figur 18: Pindiagram over HEF4001B

7.1.6 Arduino

Se afsnit 11.

7.2 Teori

7.2.1 Benyttelse af diode som variabel spænding

7.2.2 Nor-gate

En norgate er en af de logiske gate der tager et digital signal som input og laver et nyt digitalt signal som output baseret på hvilke dens input. I vores tilfælde havde vi tænkt os at bruge en not gate til at invitere vores input. Men siden at det var nemmere at få fat på en nor-gate i stedet for en not gate har vi brugt den i stedet for og sat den ene inputpin til ground så den fungere ligesom en notgate.

7.2.3 Hastighedsmålingsmetoder

Tidsmålingen bestemmes ved at vi kender et hvis strækning bolden har bevæget sig i, samt at man kender det tidsinterval bolden bevæger sig strækningen. Således bestemmes hastigheden:

$$v_{bold} = \frac{\Delta s}{\Delta t}$$

Heraf er der to løsninger til at bestemme dette. Den ene er at benytte to infrarød afsender og modtagere kredse, som er sat sammen i par. Således at når den første del registrerer at en bold passerer undervejs så begyndes der at tage tid, indtil den næste del registrerer at bolden passerer.

Den anden metode er at benytte en infrarød afsender og modtager-kreds og en knap, så når der trykkes på knappen tages der tid fra at bolden skydes af sted til at den registreres af infrarød afsender og modtager kreds.

Vi endte med at vælge den anden metode, grundet tidspres da vi efter vores fremstilling stødte på problemer, da kredsløbet ikke fungerede på PCB. Se afsnit 12.4.

7.3 Beregninger

7.3.1 555 timer modstande

For 555 gælder det at vi ønsker en frekvens på

$$f_{target} = 1000 \text{ Hz}$$

Hold øje med om der refereres til det rigtige afsnit

mangler nogle enkelte beregninger

Der kan opstilles 4 ligninger for 555 timeren således: For mark-time gælder det at

$$T_m = 0.7 \cdot C_1 \cdot (R_1 + R_2) \quad (2)$$

For space-time gælder det at

$$T_s = 0.7 \cdot C_1 \cdot (R_2) \quad (3)$$

For perioden gælder det at

$$T = \frac{1}{f} = T_s + T_m \quad (4)$$

Og da dutycyclen skal være så tæt på 100 som muligt sættes følgende forhold til at gælde

$$T_m = 100 \cdot T_s \quad (5)$$

Ved at løse ligningssystemet for ligningerne 3, 5, 2, 4, og isolerer for R_1 og R_2 og opskriver den som funktion af kapacitoren og frekvensen fås udtrykkende:

$$R_1(C_1, f) = 1.400 \frac{1}{f \cdot C_1}$$

$$R_2(C_1, f) = 0.01414 \frac{1}{f \cdot C_1}$$

Så antages at kondensatoren er

$$C_1 = 1.0 \cdot 10^{-6} \text{ F}$$

Da det antages vi bare benytter standardværdien.

Frekvensen er nævnt og således bliver modstandende

$$R_1(1.0 \cdot 10^{-6} \text{ F}, 1000 \text{ Hz}) = 1400 \Omega$$

$$R_2(1.0 \cdot 10^{-6} \text{ F}, 1000 \text{ Hz}) = 14.14 \Omega$$

Er det standard-værdien?***

7.3.2 Bestemmelse af modstande for peak-detektoren

Vi har besluttet at benytte en peak-detektor til at opfange IR signalet til modtageren. Heraf benyttes der en simpel peak-detektor opbygning. Udtryk og teori benyttet er fra kilde [23].

Ud fra diodens karakteristika fås en forward-biased resistans på

$$r_{df} = 607 \Omega$$

og en reversed bias resistens på

$$r_{dr} = 2 \cdot 10^7 \Omega$$

Således gælder det at vi skal vælge en modstand R hvor det gælder at

$$r_{df} < R < r_{dr}$$

Vores tidskonstant τ_2 kan defineres som

$$\tau_2 = R \cdot C$$

hvor C er størrelsen på kapacitoren benyttet i peakdetektoren. Vi har besluttet at peaket skal være omtrent ved

$$U_{peak} = 4 \text{ V}$$

Og at differensområdet er på

$$\Delta U = 0.5 \text{ V}$$

Vi beslutter at den modstand der benyttes i kredsen skal være på

$$R = 56\,000 \Omega$$

Vi regner med at få en frekvens på

$$f = 1000 \text{ Hz}$$

som vi prøver at tilnærme Således hvis man skal opfylde følgende sammenhæng, må det

Er det
den
rigtige
beteg-
nel-
se***

gælde at:

$$\begin{aligned}\Delta U &= \frac{U_{peak}}{f \cdot t_2} \\ \iff \Delta U &= \frac{U_{peak}}{f \cdot R \cdot C} \\ \iff C &= \frac{U_{peak}}{f \cdot R \cdot \Delta U} \\ &= \frac{4 \text{ V}}{1000 \text{ Hz} \cdot 56000 \Omega \cdot 0.1 \text{ V}} \\ &= 7.14 \cdot 10^{-7} \text{ F} \\ &\approx 1 \mu\text{F}\end{aligned}$$

Således skal vi benytte en kondensator til peakdetectoren på $1 \mu\text{F}$

7.4 Test

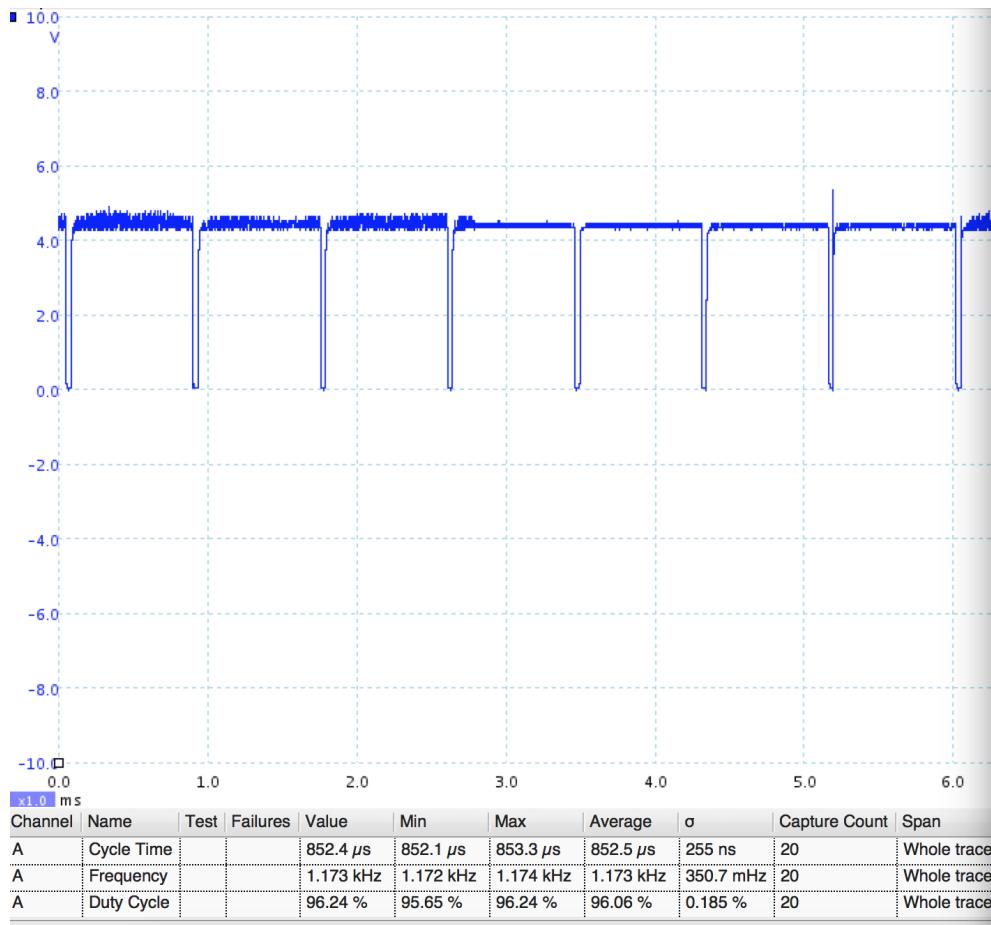
7.4.1 Afsender dioden

Vi testede 555-timeren med et PC-oscilloskop.

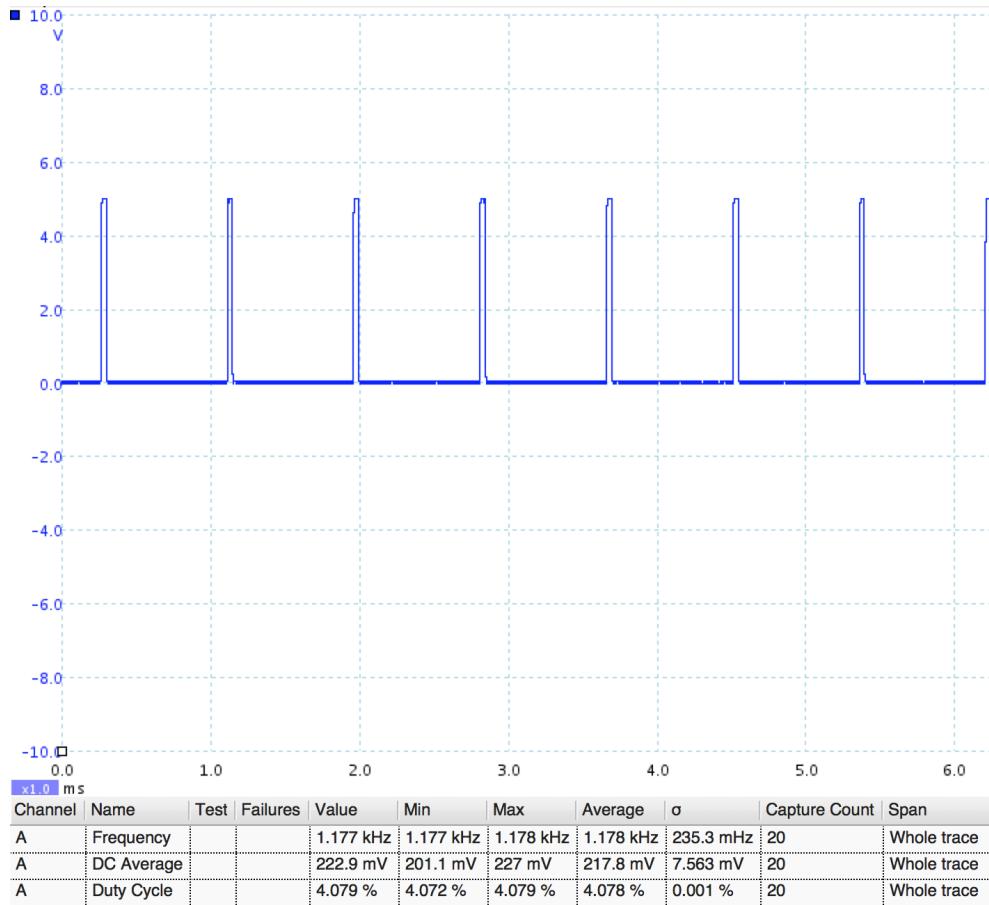
På Figur 19 er der en graf af den 555-timerens output, som det fremgår på info-boksene er duty cyclen er høj som beregnet. Det bemærkes også at frekvensen er lidt større end 1000 Hz men vi forventer ikke det giver os problemer.

Det inverterede signal kan ses på 20 og det fungerer som ønsket.

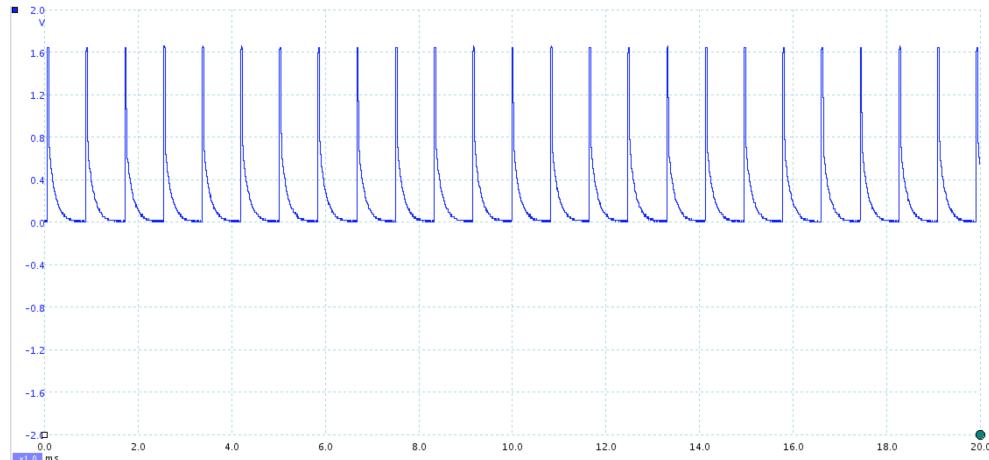
Der skal nok referes til billede og navne på komponenter i billede når der snakkes om tests***



Figur 19: Spænding som funktion af tiden med info-bokse i bunden



Figur 20: Det inverterede signal som funktion af tiden med info-bokse i bunden



Figur 21: Spænding over dioden som funktion af tiden

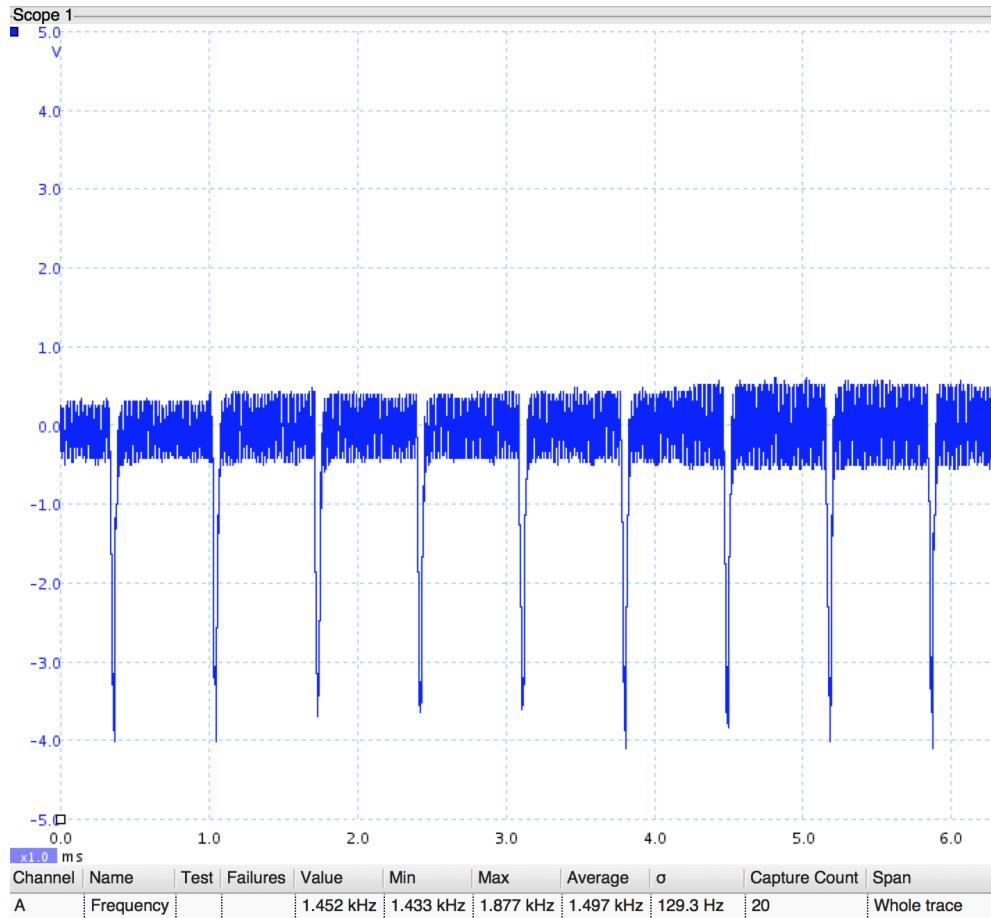
Spændingen over dioden kan ses på Figur 21. Frekvensen estimeres ud fra de 12 første bølgetoppe. Heraf bemærkes at

$$f_{diode} \approx \frac{12}{10 \cdot 10^3 \text{ s}} = 1200 \text{ Hz}$$

Hvilket svarer fint til frekvensen af outputtet fra 555-timeren.

7.4.2 Modtager dioden

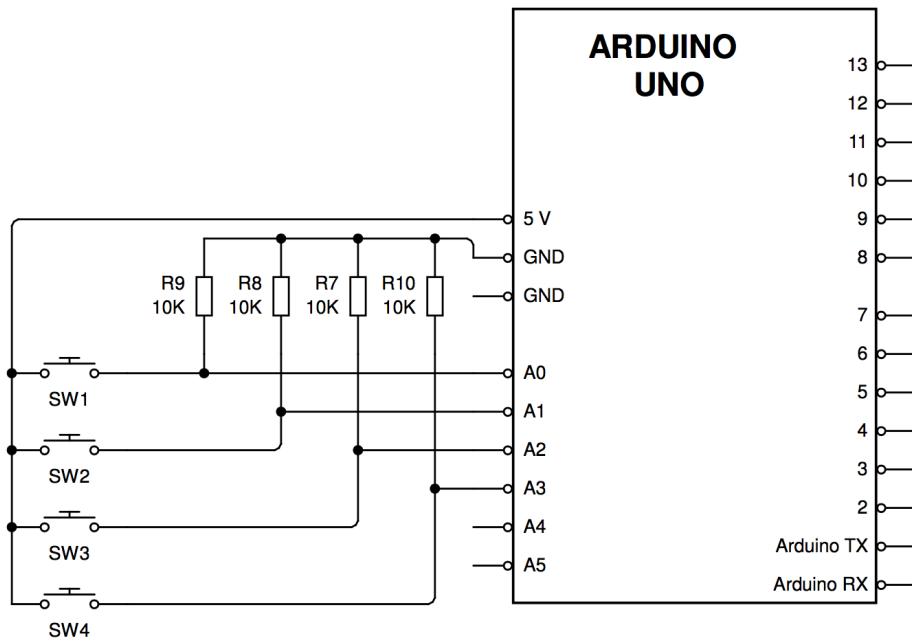
For modtager-delen af kredsen fremgår det på 22 at frekvensen er omrent $1.4 \cdot 10^3$ Hz, og et tydelig spændingsforskel på omrent -5 V hvilket er et fint og tydeligt signal. Dog afviger frekvensen væsentlig fra vores ønskede frekvens, men vi regner ikke med at det giver problemer med hensyn til udformningen af produktet.



vi kunne muligvis godt have mere tekst her

Figur 22: Spændingen som funktion af tiden af modtagerkredsens signal

8 Controller



Figur 23: Et billede af kredsløbet for controller kredsen.

8.1 Komponenter

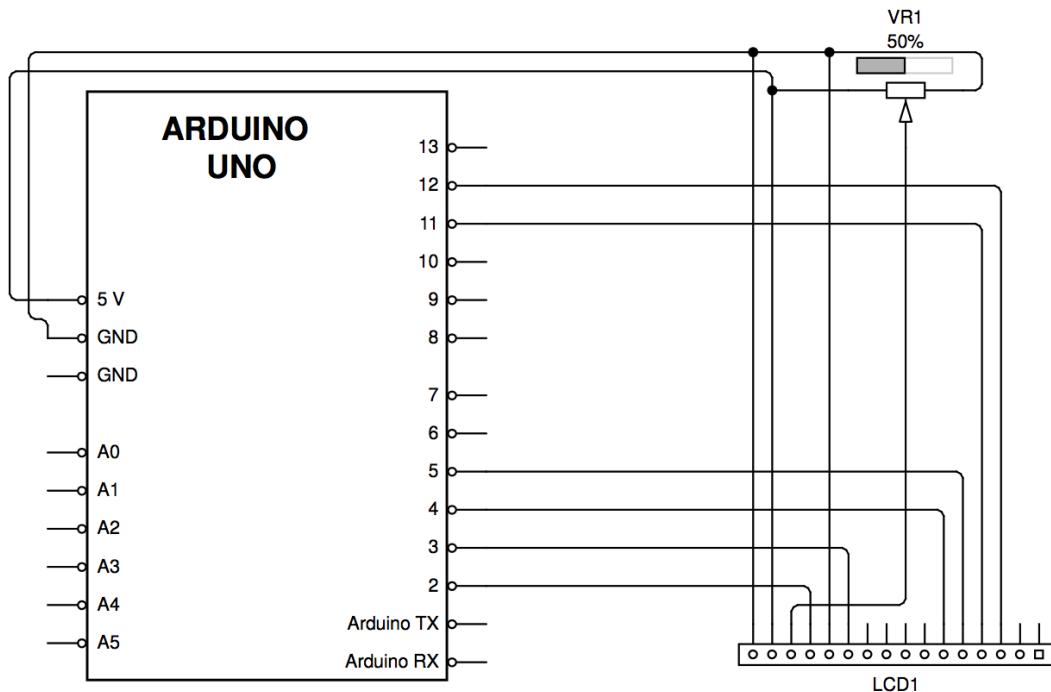
I kontrolleren bliver der kun brugt simple fysiske switches, modstande og en Arduino. Se afsnit ?? for beskrivelse af Arduino.

8.2 Teori

For at undgå en kortslutning bliver der sat en modstand imellem den ene switch side og ground. På den anden side a switchen er der 5V.

9 Display

9.1 LCD-kredsløb



Figur 24: Arduino LCD kreds lavet ud fra arduino siden, se [4]

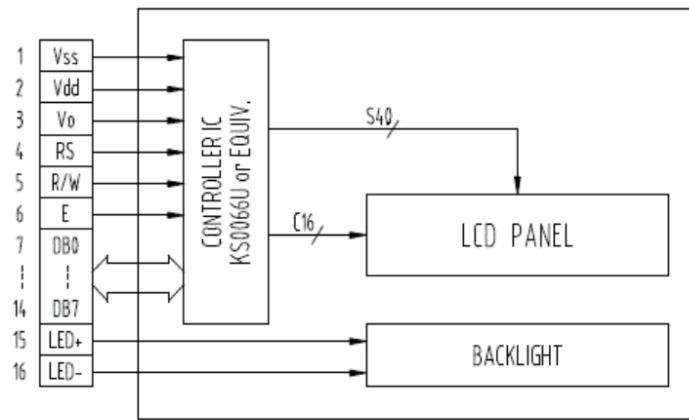
9.1.1 LCD setup

Vi benytter et LCD display, da det let kan forbindes direkte til arduioen uden et ekstra interface ved brug af LiquidCrystal biblioteket. Biblioteket vi benytter gør brug af ASCII kode se figur 26 til at vise symboler på et display. Vi benyttede et 4-bit datainterface setup (se [4]), hvor displayet skriver på to linjer.

9.2 Komponenter

9.2.1 LCD display

LCD displayet vi benyttede havde en indbygget controller som vist på Figur 25.



Figur 25: Et blok diagram over databen og kontrolben til kontroleren, LCD panel. Kontrolleren er indbygget i displayet. kilde: [Uddrag fra projektvejledning]

9.2.2 Arduino

Se afsnit 11

9.3 Teori

9.3.1 LiquidCrystal library

LiquidCrystal library bruger vi til at kunne få arduinoen til at sende den ønskede tekst af f.eks. af en måling til et LCD. LiquidCrystal gør brug af ASCII tegnsæt til at tolke binære signaler til tekst. Et uddrag på ASCII karakterer kan ses på Figur 26.

		upper 4 bit	0000	0010	0011	0100	0101	0110
		lower 4 bit	0000	CG RAM (1)	0001	0010	0011	0100
		0000	0000	0010	0011	0100	0101	0110
0000	(1)	0000	0000	0010	0011	0100	0101	0110
0001	(2)	0001	0001	0010	0011	0100	0101	0110
0010	(3)	0010	0010	0011	0100	0101	0110	0110

Figur 26: Et eksempel på nogle ASCII koder benyttet af LiquidCrystal library - kilde: [Uddrag fra projektvejledning]

9.4 Test

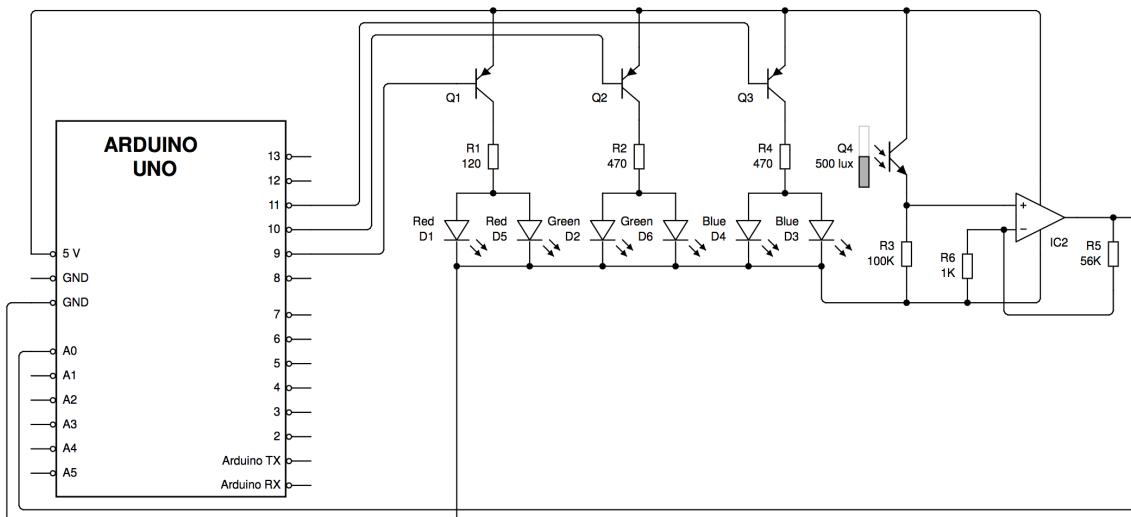
Vi benyttede en simpel ”Hello World” program til at teste skærmens, koden fik vi fra [4].

Listing 1: kodeeksempel ”Hello World” med timer

```
1 // Inkluder koden fra liquidcrystal library:  
2 #include <LiquidCrystal.h>  
3  
4 //Initialiser biblioteket  
5 //med disse interfacepins.  
6 LiquidCrystal lcd(12, 11, 5, 4, 3, 2);  
7  
8 void setup() {  
9 //Bestemmer antallet af hhv.  
10 //kolonner og rækker for displayet  
11 lcd.begin(16, 2);  
12 //Printer besked til displayet  
13 lcd.print("hello, world!");  
14 }  
15  
16 void loop() {  
17 // Sætter cursor til at være på anden række,  
18 // som er markeret 1.  
19 lcd.setCursor(0, 1);  
20 //Printer antallet af sekunder der  
21 //er gået siden reset.  
22 lcd.print(millis() / 1000);  
23 }
```

Potentiometret kunne justerer skærmens styrke fint.

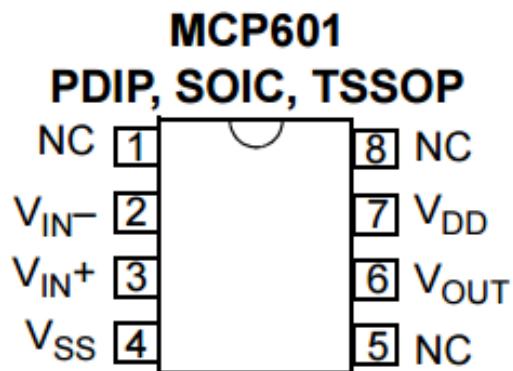
10 Ladningssensor



Figur 27: Et billede af kredsløbet for ladningssensor kredsen.

10.1 Komponenter

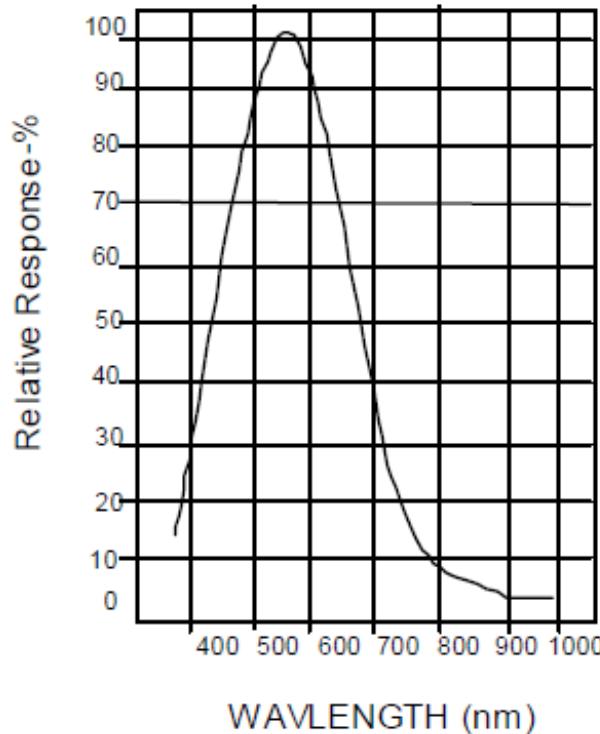
10.1.1 OP Amp - MCP601



Figur 28: Bendiagram af MCP601
Se databladet i kilde [24].

Vi benytter en OP Amp, til at forstærke signalet som vi modtager fra fotodioden, således at den bruger hele spændingsintervallet ($[0 \text{ V}; 5 \text{ V}]$) som Arduinoen kan måle i. Der er beskrevet mere om OP Amps i teoriafsnittet 10.2.1.

10.1.2 BPW21 - Fotodiode

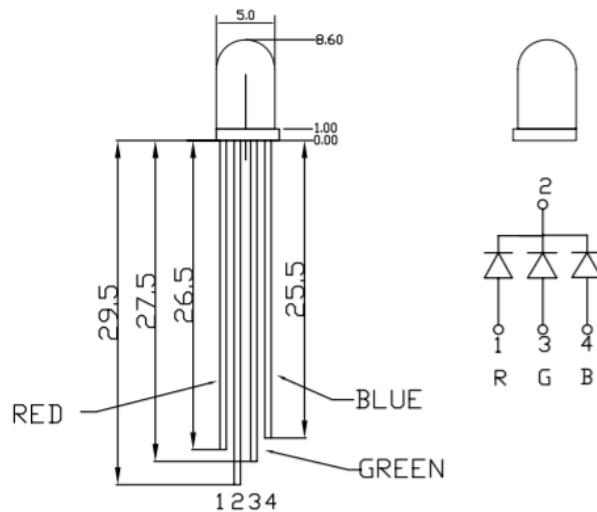


Figur 29: Normal spæltral respons angivet i procent
Se databladet i kilde [8].

BPW21 er en fotodiode, som betyder at det er en diode som omdanner lys til elektrisk strøm. En fotodiode operer i reverse biased, hvilket betyder at cathoden skal stå mod positiv strømretning, så der dannes et spærret PN-overgang. Denne PN-overgang kan så lede en strøm hvis den bliver belyst og således giver ladning til en overgang. Denne effekt af lysintensitet og reverse current er omrent proportional. Dette gør at vi kan modtage et svagt signal der derefter skal blive forstærket, og afhængigt af bølgelængden, vil den effektive strøm der kan ledes over PN-overgangen varierer, se Figur 29. For at finde ud af størrelsen af signalet for forskellige farver producere ar der blevet foretaget målinger, se tabel 1, på side 39. Se hvordan forstærkningen er blevet bestemt (ved brug af en OPAMP) i afsnit 10.3.2.

skriv
teori
om
BPW21

10.1.3 LED - RGB WHITE DIFFUSED LENS COMMON CATHODE



Figur 30: Diagram af LED'en
 Se databladet i kilde [9].

ITEMS	Color	Symbol	Condition	Min.	Typ.	Max.	Unit
Forward Voltage	Red	V_F	$I_F=20\text{mA}$	1.8	2.0	2.2	V
	Green			3.0	3.2	3.4	
	Blue			3.0	3.2	3.4	

Figur 31: Spændingsfaldet over de forskellige LED'er
 Se databladet i kilde [9].

Vi benytter en LED som kan lyse i 3 forskellige farver, rød, grøn og blå. Vi benyttede en LED med en fælles katode. Dette betyder at vi kan koble LED'ens katode til jord, også styre spændingen sat på hver af deres anoder. For ikke at trække for meget strøm fra arduinoen, benytter vi en PNP transistor som kontakt til LED'erne.

10.1.4 PNP transistor

Den funktionelle forsker på en PNP-transistor og en NPN transistor er hvordan den er biased i forhold til base. Det fremgår derfor på Figur 27 at den modtager 5 V forsynings-spænding ved Emitter-benet, og den skal således stadigt modtage et signal fra arduinoen for at få strømmen til at løbe videre til dioderne.

Skriv om PNP transistor

10.1.5 Arduino

Se afsnit 11

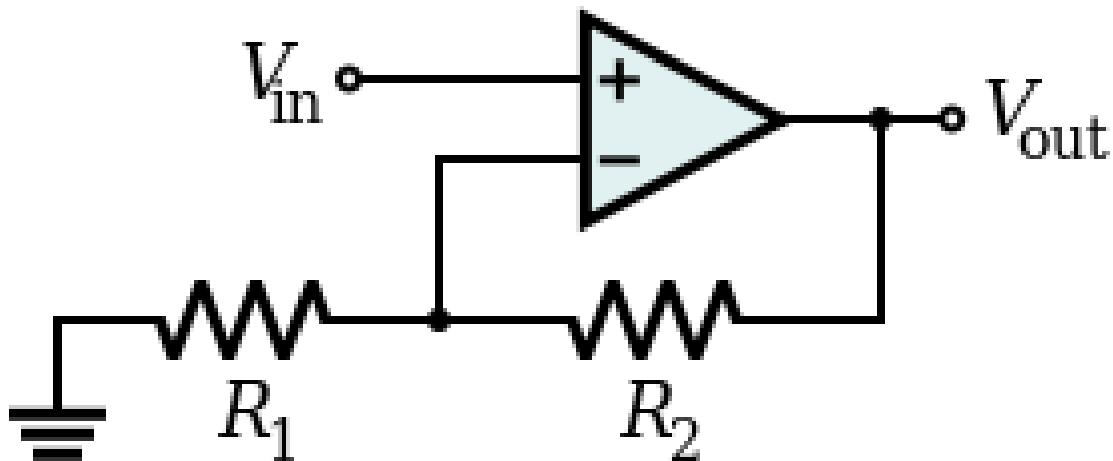
10.2 Teori

10.2.1 OP Amp

En operationsforstærker (OP Amp) er en type forstærker der tager imod to inputs, inverting input voltage (V_{in-}) og non-inverting input voltage (V_{in+}). Outputtet fra en operationsforstærker (V_{out}), kan beskrives med formlen

$$V_{out} = A(V_{in+} - V_{in-}) \quad , \quad V^- \leq V_{out} \leq V^+$$

Her er A den elektriske gain af operationsforstærkeren. A er som reel omkring 10^5 til 10^6 , men man anser den som at gå imod uendelig. Vi benyttede os af en non-inverting amplifier.



Figur 32: El diagram for non-inverting amplifier

Som man kan se ud fra el diagrammet 32, er der en spændingsdeler imellem V_{out} og V_{in-} . Vi ved at for en spændingsfordeler gælder der

$$U_{out} = U \cdot \frac{R_2}{R_1 + R_2}$$

Hvor at U_{out} er spændingen over R_2 og U er spændingen over R_1 og R_2 . Dette betyder at

$$V_{in-} = V_{out} \cdot \frac{R_2}{R_1 + R_2}$$

Hvis vi indsætter dette i vores udtryk for V_{out} gælder der

$$\begin{aligned}
 V_{out} &= A(V_{in+} - V_{in-}) \quad , \quad V^- \leq V_{out} \leq V^+ \\
 V_{in-} &= V_{out} \cdot \frac{R_2}{R_1 + R_2} \\
 \rightarrow V_{out} &= A(V_{in+} - V_{out} \cdot \frac{R_2}{R_1 + R_2}) \\
 \rightarrow V_{out} + A \cdot V_{out} \cdot \frac{R_2}{R_1 + R_2} &= A \cdot V_{in+} \\
 \rightarrow V_{out}(1 + A \cdot \frac{R_2}{R_1 + R_2}) &= A \cdot V_{in+} \\
 \rightarrow V_{out} &= \frac{A \cdot V_{in+}}{1 + A \cdot \frac{R_2}{R_1 + R_2}}
 \end{aligned}$$

Da A er et meget højt tal, så kan 1 i nævneren negligeres.

$$\begin{aligned}
 V_{out} &= \frac{A \cdot V_{in+}}{A \cdot \frac{R_2}{R_1 + R_2}} \\
 &= \frac{V_{in+}}{\frac{R_2}{R_1 + R_2}} \\
 V_{out} &= V_{in+} \cdot \left(\frac{R_1}{R_2} + 1 \right) \\
 \rightarrow \frac{V_{out}}{V_{in+}} &= \frac{R_1}{R_2} + 1
 \end{aligned}$$

Dette udtryk kan vi benytte til at beregne forstærkningen af vores non-inverting amplifier. (Kilde: Teknik A - OP-amp DC-kobling (teori udleveret af læreren uden yderligere kildeangivelse))

10.2.2 Analog input til registrering af farve

Arduinoens analog input er beskrevet i afsnit 11.1.1. Vi benytter denne funktion til at få outputtet fra vores farvesensor ind i arduinoen.

10.3 Beregninger

10.3.1 Modstanden af LED

Ifølge [kompLED] gælder det for LED'en at dens typiske spænding for den blå og grønne LED

$$V_{GBtypisk} = 3.2 \text{ V}$$

Og for den røde

$$V_{Rtypisk} = 2 \text{ V}$$

Og kan klare en strømstyrke på

$$I = 20 \cdot 10^{-3} \text{ A}$$

Ud fra ohms lov må det betyde at hvis indgangsspændingen er på 5 V, at modstanden til de enkelte LEDer skal være

$$R = \frac{U}{I}$$
$$R_{RLED} = \frac{5 \text{ V} - 2 \text{ V}}{2.0 \cdot 10^{-2} \text{ A}} = 150 \Omega$$
$$R_{GLED} = \frac{5 \text{ V} - 3.2 \text{ V}}{2.0 \cdot 10^{-2} \text{ A}} = 90 \Omega$$
$$R_{BLED} = \frac{5 \text{ V} - 3.2 \text{ V}}{2.0 \cdot 10^{-2} \text{ A}} = 90 \Omega$$

Ud fra dette valgte vi at benytte modstandene

$$R_{RLED} = 180 \Omega$$

$$R_{GLED} = 100 \Omega$$

$$R_{BLED} = 100 \Omega$$

Det vidste sig dog at en LED ikke lyste nok op, og derfor satte vi en ekstra parallelt med den anden som vist på kredsløbstegningen. Dette betød at vi skulle beregne vores modstande igen. For en parallelforbindelse gælder

$$\frac{1}{R_{tot}} = \frac{1}{R_1} + \frac{1}{R_2} + \dots + \frac{1}{R_n}$$

hvor at der er n parallelle modstande, som ønskes at erstattes med en samlet modstand R_{tot} . Da hver af de parallelle modstande, var de modstanden beregnet før, må den nye

samlede modstand for hver farve være

$$\begin{aligned} R_{R-tot} &= \frac{1}{\frac{1}{R_{RLED}} + \frac{1}{R_{RLED}}} \\ &= \frac{1}{\frac{1}{180\Omega} + \frac{1}{180\Omega}} \\ &= 90\Omega \end{aligned}$$

$$\begin{aligned} R_{G-tot} &= \frac{1}{\frac{1}{R_{GLED}} + \frac{1}{R_{GLED}}} \\ &= \frac{1}{\frac{1}{100\Omega} + \frac{1}{100\Omega}} \\ &= 50\Omega \end{aligned}$$

$$\begin{aligned} R_{B-tot} &= \frac{1}{\frac{1}{R_{BLED}} + \frac{1}{R_{BLED}}} \\ &= \frac{1}{\frac{1}{100\Omega} + \frac{1}{100\Omega}} \\ &= 50\Omega \end{aligned}$$

Da vi havde beregnet de minimum modstande vi kunne bruge, prøvede vi os frem for at finde modstande, så hver LED lyste lige stærkt. Vi kom frem til disse modstande

$$R_{R-tot} = 120\Omega$$

$$R_{G-tot} = 470\Omega$$

$$R_{B-tot} = 470\Omega$$

Her blev der lavet en test, af indgangsspændingen til Arduinoen, når der blev lyst med forskellige farver, på materialer med forskellig farve. Testen kan findes i tabel 1. Hvis man ser på tegning 27, er den spænding vi måler i testen, spændingsfaldet fra A0 til ground.

10.3.2 OPAMP modstande

Vi besluttede os for at benytte en OPAMP, da vores største signal fra test 1 (se 1) var på 0.06 V, hvis man ser bort fra når alle lyser. Da vores precision af analog inputtet på arduinoen er omkring 0.005 V (se 11.1.1), er disse udsving for små til at med sikkerhed kunne identificere farver. Vi benytter derfor en OPAMP, så det er nemmere at identificere hvilken farve der bliver brugt. Vi valgte at den maksimale indgangsspænding til OPAMP'en er

$$V_{in+} = 0.08\text{ V}$$

Vi ønsker at dette skal være det maksimale som Arduinoen kan måle (5 V)

$$V_{out} \approx 5 \text{ V}$$

Heraf er den ønskede forstærkning

$$A_{target} = \frac{V_{out}}{V_{in+}} = \frac{5 \text{ V}}{0.08 \text{ V}} = 62.5$$

Denne forstærkning er ret høj og vi benytter derfor en non-inverting amplifier.

Vi bestemmer en rimelig værdi til den ene modstand er

$$R_2 = 1000 \Omega \quad (6)$$

Og udtrykket for forstærkningen er

$$A_{target} = 1 + \frac{R_1}{R_2} \quad (7)$$

Ved at isolerer R_1 i ligningerne 6 og 7 fås

$$R_1 = 61.5 \cdot 10^3 \Omega$$

Da vi ikke har denne modstand direkte, vælger vi at benytte en $56 \cdot 10^3 \Omega$ modstand i stedet. Hvilket giver os en reel forstærkning på

$$A_{reel} = 1 + \frac{56 \cdot 10^3 \Omega}{1000 \Omega} = 57$$

Hvilket svarer til et outputsignal på

$$V_{out} = 0.08 \cdot 57 = 4.56 \text{ V}$$

Hvilket er tilnærmelsesvis hvad vi gerne vil have. Derefter kørte vi den samme test som før vi havde forstærket signalet. Resultaterne kan ses i tabel 2.

10.4 Test

Vores teste foregik i en mørklaegt kasse, hvor afstanden mellem sensoren og objektet var 2 cm. Vi testede med 5 forskellige farver: hvid, rød, grøn, blå og gul. Vi kom frem til, ud fra vores test, at vores farvesensor efter forstærkning er tilstrækkelig til at kunne adskille farverne, hvid, rød, grøn og blå. Vi mener også dette lever op til vores krav for ladningssensoren, da vi så kan have 4 indstillinger, som vi mener er tilstrækkelig.

Tabel 1: Signaler uden forstærkning

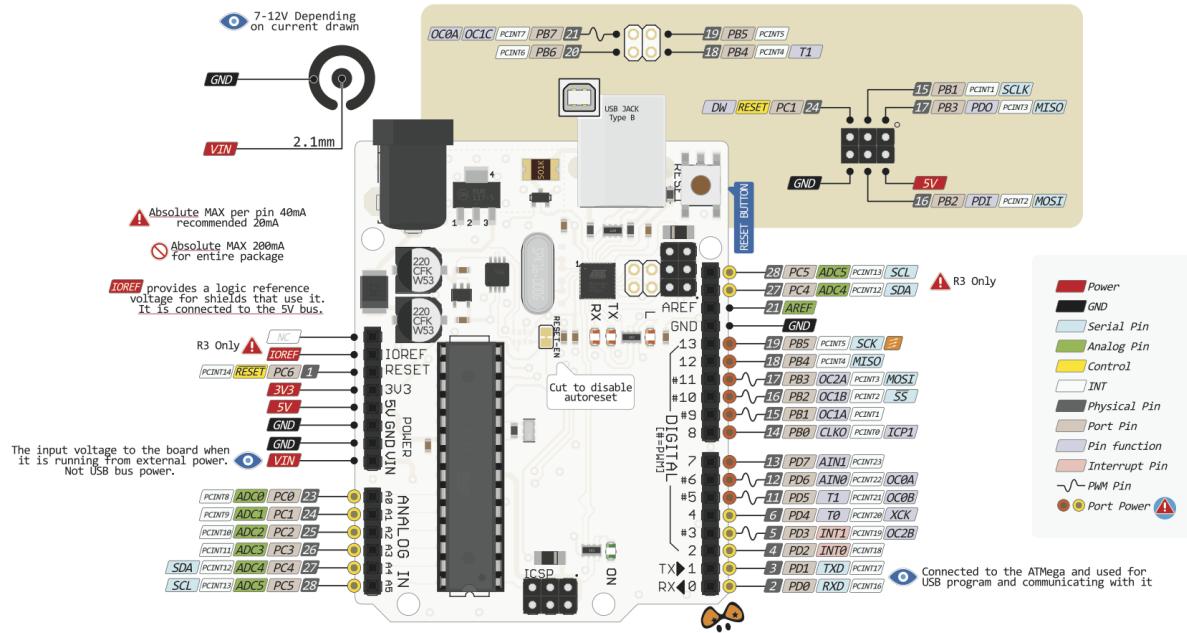
	Alle[V]	Intet [V]	Rød[V]	Grøn[V]	Blå[V]
Sort	0.03	0.01	0.01	0.02	0.02
Hvid	0.12	0.01	0.02	0.06	0.06
Rød	0.03	0.01	0.02	0.01	0.02
Blå	0.05	0.01	0.01	0.02	0.03
Gul	0.06	0.01	0.02	0.04	0.02
Grøn	0.04	0.01	0.01	0.03	0.02

Tabel 2: Signaler med forstærkning

	Alle[V]	Intet [V]	Rød[V]	Grøn[V]	Blå[V]
Sort	0.47	0.05	0.08	0.23	0.25
Hvid	4.8	0.04	0.77	4.04	4.38
Rød	1.03	0.06	0.53	0.29	0.28
Blå	2.72	0.06	0.06	0.77	1.95
Gul	3.8	0.06	0.75	2.42	0.75
Grøn	0.9	0.05	0.06	0.49	0.4

11 Arduinoen og programmering

11.1 Arduino som komponent



Figur 33: Arduino uno med ATmega328

Her kan man se hvilke ben der er på Arduinoen, og hvilke porte de fordelt på (PORTB, PORTC og PORTD). Hvis man ser på ”Port pin”, står der P for port, efterfulgt af bogstavet for hvilken port den er tildelt, efterfulgt af dens bit position i den port. Vi har også opgivet den påkrævede spænding for at køre Arduinoen. Vi kan også se Analog input ben (A0-A5) og de digital input og output ben (0-13). Vi kan se ud fra analog ben A4 og A5 at de er SDA og SCL, som er relevant når vi skal benytte I2C kredsen.

Kilde: <http://pighixxx.com/unov3pdf.pdf>

11.1.1 Analog inputs og outputs (PWM)

Analog input i Arduinoen er et input som beskriver spændingsfaldet over inputtet og jord med et 10 bit tal.[2] Dette betyder at den laveste værdi og højeste værdi af et analog input er

$$U_{MIN} = b00\ 0000\ 0000 \rightarrow 0 \rightarrow 0\ V$$

$$U_{MAX} = b11\ 1111\ 1111 \rightarrow 1023 \rightarrow 5\ V$$

Med Arduinoen kan man få et analog input igennem en analog ben (se 33), og kalde metoden *analogRead()* til benets nummer, eksempelvis A0. Da vores precision er på 10

bit, til at beskrive et maks spændingsfald på 5 V, må det gælde at den laveste ændring i spændingsfaldet vi kan måle er

$$U_{prec} = \frac{5 \text{ V}}{1023} \approx 5 \cdot 10^{-3} \text{ V}$$

11.1.2 Digital inputs og outputs

Digital input på Arduinoen er en boolean værdi, som enden kan være sand eller falsk (også kendt som TRUE/FALSE eller HIGH/LOW). Disse værdier reflektere den digitale logik på det digitale ben, som fungere således: når spændingsfaldet mellem bennet og jord er 5 V er værdien sand og når spændingsfaldet er 0 V er den falsk. Mere præcist skifter værdien mellem at være sand og falsk ved omkring 3 V, på 5 V logik Arduinoer. [3]

Et digital output følger samme logik, her der sætter man dog udgangsspændingen på bennet. Det betyder altså at hvis man sætter et digital ben til sandt, vil udgangsspændingen på bennet være 5 V og hvis man sætter det til falsk vil det være 0 V. [3]

11.1.3 Portmanipulation og registrer

I Arduinoen bliver der brugt 8 bit registrer til at styre Arduinoens ben. Et register en type hukommelse der ofte er sammensat af flip flops. En type register er et parallelt register, som har et ben for hvert bit, således at værdien af den bit kan sættes med det ben. Disse kan bruges til eksempelvis at bestemme retningen af et ben (input/output) eller huske tilstanden af digitale ben.

I Arduinoen bliver der brugt 8 bit register, som betyder at der skal flere forskellige registre til at kortlægge alle benene på Arduinoen. Disse forskellige grupper af registre kalder man porte og på figuren 33, kan man se hvilke ben der er på hvilke porte. Som man kan se på figuren, er der tale om tre forskellige porte, B, C og D.

I hver port er der primært tale om tre forskellige register. DDR registret er registret hvor at benenes tilstand er gemt i. Det betyder for eksempel at når man kalder funktionen *pinMode(pin, mode)* finder Arduinoen først ud af hvilket registre *pin* høre til, og derefter finder dens plads i registret og derefter ændre tilstanden af den bit til *mode*. Her har man simplificeret processen i at ændre tilstanden af et ben, men ofret hastighed.

De to andre primære registre er PORT og PIN. PORT registret indeholder værdien af benene (højt eller lavt), som man også kan ændre på, hvis benet er registreret som et

output i DDR registret. PIN indeholder også værdien af benene, har har man dog kun mulighed for at læse på værdien, og ikke ændre værdien. Man ville sige at PIN har samme funktion som PORT, men er *write protected*, således at man ikke ved et uheld ændre en værdi på et ben.

Man kan med fordel benytte registrer, hvis programmet skal køre hurtigere, eller en process kræver at flere bens værdi bliver læst samtidig. Et eksempel på dette er når vi gerne vil overføre knapperne tilstand fra Controlleren. Her kan vi benytte portmanipulation til at få værdien af flere ben som en enkelt byte, hvis man da har valgt ben som er på samme port. I vores tilfælde benytter vi A0, A1 og A2 til knapper, som heldigvis ligger på samme port. Dette betyder at vi kan læse alle benenes digitale værdier ved at læse PORT eller PIN til porten. I programmet for Controlleren er der en funktion *requestEvent()*, som skal sende tilstanden af knapperne. Den ser således ud når portmanipulation bruges.

```
1 void requestEvent() {  
2     //Antager at A2, A1 er høje og A0 er lav  
3     byte data = PINC; //Port C input registret  
4     //data = b0000 0110  
5     Wire.write(data); //Sender dataen over I2C  
6 }
```

Den ville se således ud hvis portmanipulation ikke bliver brugt.

```
1 void requestEvent() {  
2     //Antager at A2, A1 er høje og A0 er lav  
3     byte data = digitalRead(A2);  
4     //data = b0000 0001  
5     data = data << 1 + digitalRead(A1);  
6     //data = b0000 0011  
7     data = data << 1 + digitalRead(A0);  
8     //data = b0000 0110  
9     Wire.write(data); //Sender dataen over I2C  
10 }
```

Har kan vi se at ved at benytte PINC registret fik vi både simplificeret processen, og gjort den hurtigere. Et kald til et register som PINC tager omkring 1 instruktion som er meget hurtigere end at kalde *digitalRead(pin)* 8 gange, eller i vores tilfælde 3 gange. Resten af programmet kan findes i bilag H.

Kilde [5].

11.1.4 I2C - Synkroniseret kommunikations bus

I2C står for ”Inter-Integrated Circuit” som er en type serial kommunikations bus. I2C tillader flere komponenter at kommunikere med hinanden, kun ved brug af en fælles data ledning (SDA) og clock ledning (SCL). Dette betyder at alle komponenter på en I2C bus har en adresse, ofte i form af en 7 bit eller 8 bit værdi. I en I2C bus omdeler man komponenterne i to grupper; slaver og master. Slaver har ikke tilladelse til at sende data på bussen, ved mindre den er blevet anmodet om det, mens master kan sende data ud uden at blive anmodet.

Hvis vi tager udgangspunkt i vores produkt, er Controlleren en slave og den anden en master. Når master anmoder om noget fra Controlleren, så som værdien af knapperne, sender den først et START signal som indikere til andre master på bussen at bussen er optaget, derefter sender den adressen af det komponent den anmoder data fra. Her skal slaven som bliver anmodet sende et ”Acknowledge”, som indikere at den er klar til at modtage. Her indikere master så hvor mange bytes den anmoder om, hvorefter slaven sender det antal bytes. Når master så har modtaget dette bliver der sendt et STOP til bussen, som betyder at bussen ikke er optaget længere.

Det praktiske ved I2C bussen er at det kræver få ledninger, netværket er nemt at udvide og et komponent på bussen kan blive ”interrupted” (afbrudt). Dette betyder at hvis slaven er i gang med nogle beregninger, kan den blive afbrudt, hvis masteren anmoder om data. Dette betyder at I2C er ideel i vores tilfælde, hvor at vi har brug for at Controlleren (slaven) svare hurtigt når master anmoder om knappernes tilstande, så brugeren ikke føler ”forsinkelse” fra når man klikker på knapperne og kanonen rykker sig. Vi benytter biblioteket ”Wire” til at håndtere I2C kommunikation.

Kilde [10]

11.2 Kort beskrivelse af programmets formål

11.2.1 Master Arduino

Vores Master Arduino står for styring af motorerne, ladningssensor og hastighedssensor. Programmet til Master Arduinoen skal kunne kommunikere med Controlleren, så den kan se om der skal rygges op eller ned på kanonen eller om der skal affyres. Den skal også kunne bestemme farven af bolden i røret og om der faktisk er en bold i røret. Når bolden bliver affyret skal hastigheden af bolden også kunne bestemmes. Til sidst skal den kunne sende information som farven af bolden, hældningen af kanonen og hastigheden af bolden til Controlleren, så det kan skrives på displayet.

11.2.2 Controller Arduino

Controller Arduinoen skal kunne sende data om knappernes tilstand og kunne modtage data om hvad der skal stå på displayet. Da vores bruger benytter Controlleren til at styre kanonen, skal forsinkelsen imellem at man trykker på en knap, til at kanonen ryger sig være relativ lille.

Programmet til master Arduinoen kan findes i bilag G og Controller Arduinoen i bilag H, som er kommenteret. De mest komplekse og primære funktioner vi har programmeret, er beskrevet i sektion 11.4.

11.3 Oversigt over inputs og outputs og lokation

Tabel 3: Inputs og outputs for Master Arduinoen

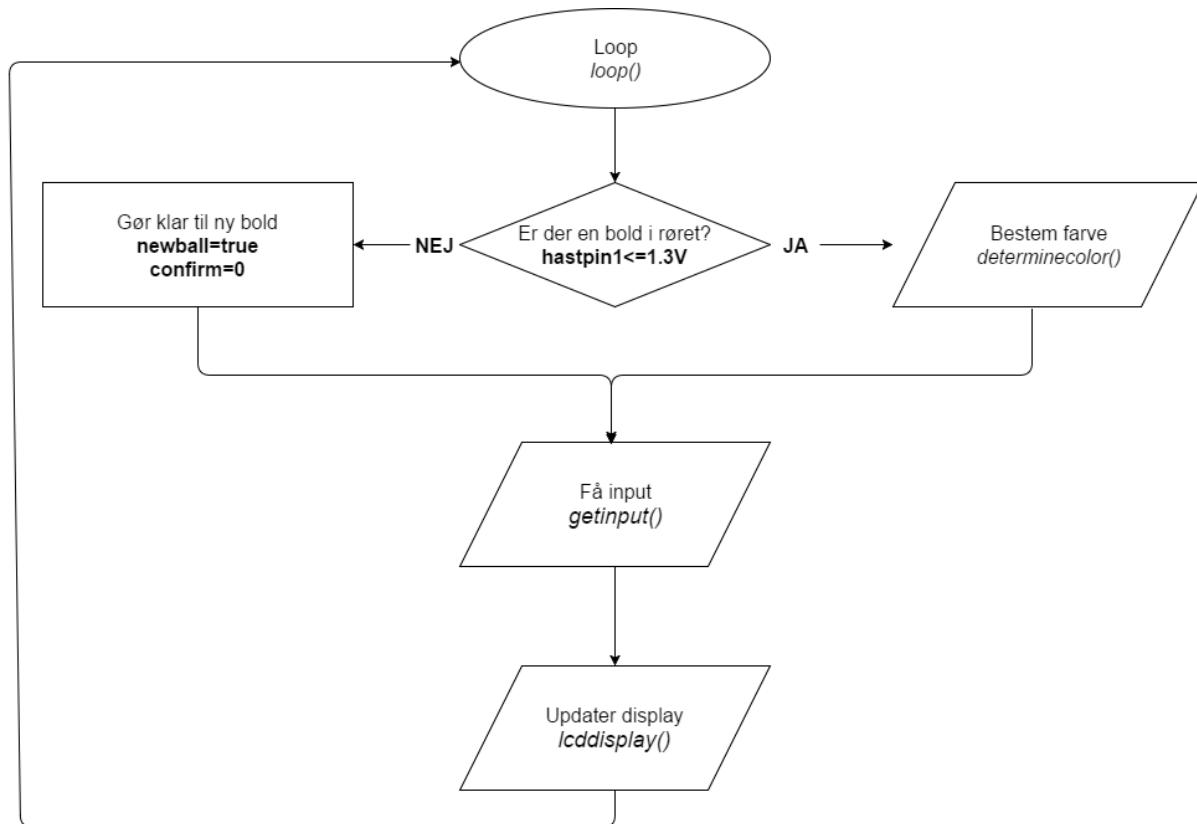
Ben	I/O	Forbundet komponent
2	Output	Stepper motor
3	Output	Stepper motor
4	Output	Stepper motor
5	Output	Stepper motor
6	Output	Trækmotor
9	Output	Gearmotor (fri)
10	Output	Gearmotor (lås)
11	Output	Ladningssensor LED (grøn)
12	Output	Ladningssensor LED (rød)
13	Output	Ladningssensor LED (blå)
A0	Input	Ladningssensor Fotodiode
A1	Input	Hastighedssensor 1
A2	Input	Hastighedssensor 2
A4 (SDA)	I/O	I2C kommunikation (data)
A5 (SCL)	I/O	I2C kommunikation (clock)

Tabel 4: Inputs og outputs for Controller Arduinoen

Ben	I/O	Forbundet komponent
2	I/O	LCD (DB7)
3	I/O	LCD (DB6)
4	I/O	LCD (DB5)
5	I/O	LCD (DB4)
11	Output	LCD (E)
12	Output	LCD (RS)
A0	Input	Knap (ryg kanon op)
A1	Input	Knap (ryg kanon ned)
A2	Input	Knap (affyre bold)
A4 (SDA)	I/O	I2C kommunikation (data)
A5 (SCL)	I/O	I2C kommunikation (clock)

11.4 Kort beskrivelse af funktioner

11.4.1 Master Arduino - Loop()



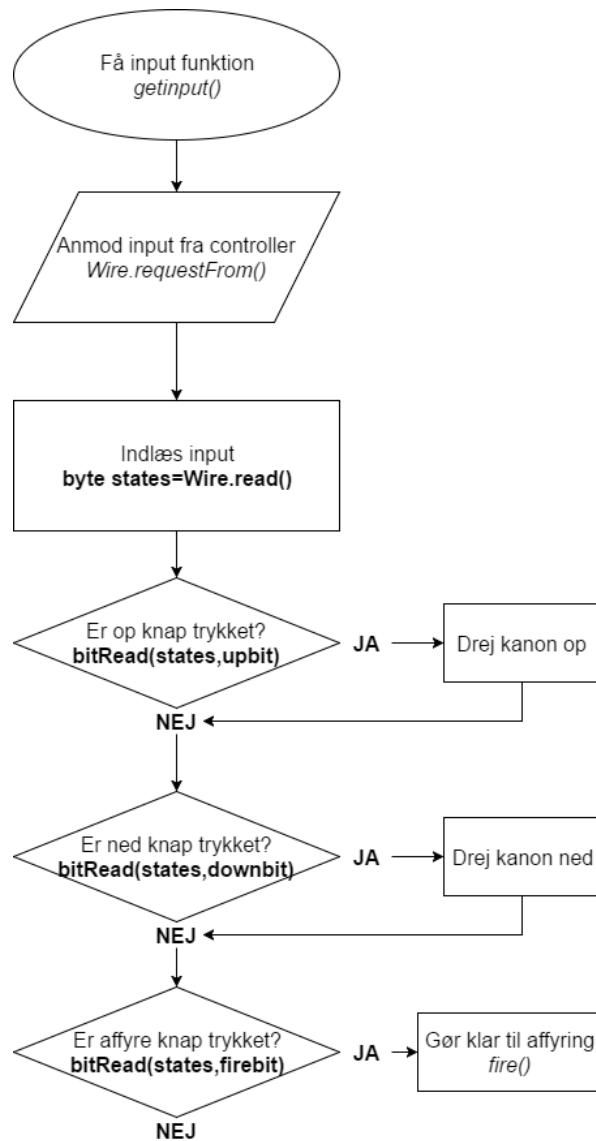
Figur 34: Flowchart af main loop hos Master Arduinoen

Da vores første hastighedssensor er det sted i røret, hvor bolden vil hvile, ses der først om den første hastighedssensor er blokeret. Hvis den er blokeret, så kald funktionen

determinecolor(). Denne funktion bestemmer farven af bolden. Hvis hastighedssensoren ikke er blokeret, skal den nulstille nogle værdier, så den er klar til den næste bold.

Efter alt dette skal funktionen *getinput()* kaldes. Denne funktion får knap værdierne fra Controlleren, altså brugerinputtet. Derefter opdateres displayet med funktionen *lcd-display()*, som sender dataen som skal være på displayet over til Controlleren. Derefter starter *loop()* forfra.

11.4.2 Master Arduino - *getinput()*

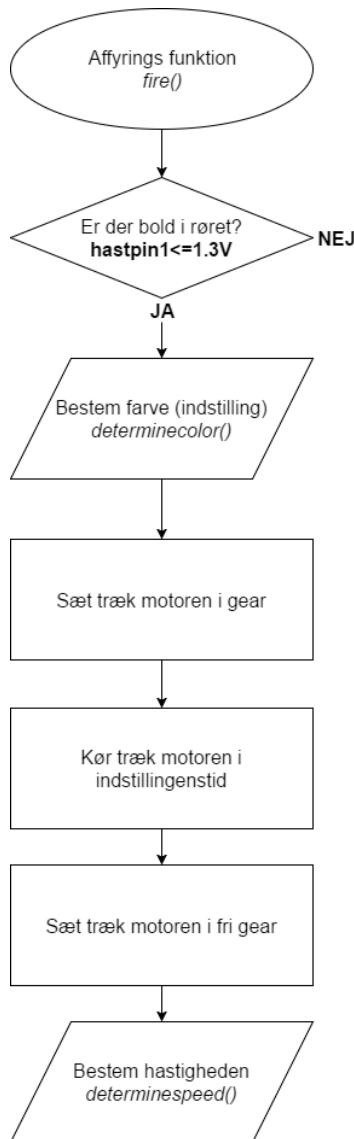


Figur 35: Flowchart af *input()* funktionen hos Master Arduinoen

Det første der sker når *getinput()* bliver kaldt, er at anmode om knappernes værdier fra Controlleren, med funktionen *Wire.requestFrom()*. Derefter læses den byte som der sendes

fra Controlleren. Denne byte indholder værdien af de 3 knapper. Her benyttes *bitRead()* funktionen til at læse den bit på position upbit. Hvis denne bit er sand skal kanonen drejes op. Derefter tjekkes om nedbitten er sand. Hvis den er dette så skal kanonen drejes ned. Derefter tjekkes om affyringsbitten er sand. Hvis den er dette skal funktionen *fire()* kaldes, som affyre bolden, hvis der er en bold i røret.

11.4.3 Master Arduino - fire()



Figur 36: Flowchart af fire() funktionen hos Master Arduinoen

Når *fire()* funktionen bliver kaldt, ses der først om der er en bold i røret, ved at se om den første hastighedssensor er blokeret. Hvis den ikke er dette, så stop funktionen. Ellers fortsæt til at bestemme farven, som er indstillingen. Når farven er bestemt, sæt træk motoren i gear, således at elastikkerne kan trækkes i. Derefter kør træk motoren i den

mængde tid som indstillingen siger. Derefter sæt træk motoren i frigear og stop træk motoren. Nu er elastikkerne i gang med at trække sig sammen igen. Her kaldes funktionen *determinespeed()* til at bestemme hastigheden som bolden bliver skudt ud med.

11.4.4 Master Arduino - *determinespeed()*

I *determinespeed()* beregner vi hastigheden af bolden. Dette gøres ved at se hvor lang tid det tager bolden at flyve afstanden imellem hastighedssensor 1 og 2. Hvis vi siger at afstanden imellem dem er s og tiden det tager bolden at flyve denne afstand er δT , så kan hastigheden beregnes

$$v = \frac{s}{\delta T}$$

11.4.5 Master Arduino - *determinecolor()*

determinecolor() bestemmes farven. Vi arbejder med 4 forskellige farver; hvid, rød, grøn og blå.

12 Fremstilling

12.1 Projektler

Vi udformede projektlerne ved at benytte normale 40mm bordtennisbolde, hvor 3 blev overmalet med akrylmaling for at give dem stærk farve af hhv. rød, grøn og blå. Se Figur 37

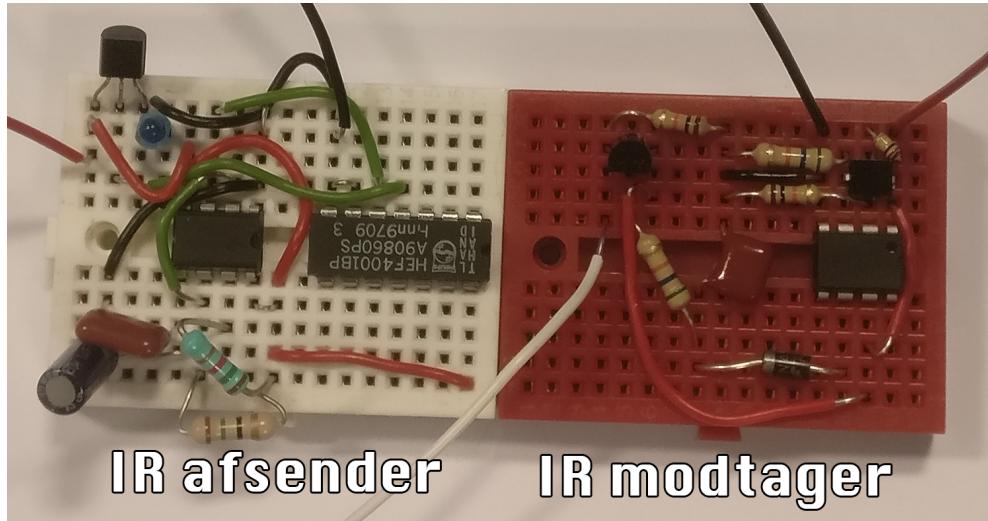
Vi
skal
nok
have
noget
om
hvor-
dan
kanon-
prototypen
blev
frem-
stillet



Figur 37: Et billede af vores benyttede projektiler

12.2 Fumlebræt-modeller

Vi lavede en række fumlebrætsmodeller for forskellige dele af det samlede kredsløbet.



Figur 38:

12.3 PCB - fremstilling

Vi ønskede vores kredsløb på et par mindre PCB. Vi designede to PCB i programmet Livewire i kombination med PCB Wizard. Det ene PCB havde 4 delkredse som skulle skilles fra hinanden ved sawning. Disse delkredse var 2 af dem IR-afsender kredsløbet og 2 af dem var IR-modtager kredsløbet. Se det første PCB i Bilag D.

Det andet PCB var et shield til arduinoen der skulle forbindes til diverse kredsløb. Se Bilag E.

12.3.1 Kemisk fremstilling

Ved at printe vores print på transfer-papir, kan det benyttes som beskyttelse på kobberplader. Kobberpladerne bliver først UV-belyst, hvor de derefter skylles i en blanding af vand og kaustisk soda indtil printet kan tydeligt ses. Herefter skylles kobberpladen og den lægges i et syrekar indtil de ubeskyttede områder er tilstrækkeligt ætset væk.

12.3.2 Problemer med printboards

Vi havde produceret 3 forskellige printboards som vi prøvede at få til at virke. På det først bræt var det ikke alle steder hvor syren havde ætset kopperet væk i den kemiske fremstillingsfase, hvilket kunne være medførst af at syren ikke blev fordelt ordenligt da den skulle i syrebadet. med vores næste forsøg blev der brugt en for høj varme til at lodde de forskellige komponenter på at noget af koppen blev loddet af sammen med det. Dette kunne nemt blive undgået ved at man fremover kun bruger temperature der skal til for at smelte lige netop den type af loddetin der blev brugt til vores produkt. På det sidsde har det været meget svært at se en tydelig fejl da alle de forskellige tænkelige problemer der kunne medføre at det ikke virker er blevet testet. Test 1 har været at der muligvis har været forbindelse forskellige steder på vores PCB. Dette blev dog testet ved at bruge et multimeter og teste for kortslutninger. Test 2 har været at der muligvis mangler ledninger på selve PCB layoutet. Der er stor mulighed for at der er nogle ledninger der ikke bliver placeret i PCB-wizard hvis man laver manuelle ændringer som vi har gjort for at placere 4 forskellige dele overskueligt på PCBet. Vi har dog gjort en betydelig insats for at dette ikke bliver et problem ved bare at lave det med så få menneskelige inflydelse som muligt. og ved at checke det igennem bagefter. test 3 som er at der er noget galt med nogle af komponenterne. Der er forskellige ting som kunne være skyld i dette. De kunne være placeret med forkert rotation eller at der er brugt forkerte komponenter eller det kunne være muligt at der er nogle af komponenterne der simpelthen ikke virker. Vi har dog prøvet at kigge på komponenterne og se om der står det samme på dem som der gjorde i vores forrige prototype, og det er ikke særlig ofte at komponenter ikke virker når de bliver sat i vores kredsløb. Men hvis vi selv har været skyld i at de ikke virker længere af forskellige årsager er det stædig det problem vi mener har størst mulighed for at være skyld i problemet af de overfor nævnte mulige problemer.

Beskriv
hvordan
PCB-
baseret
produkt
ikke
fun-
grede,
har vi
nogen
mulige
forkla-
rin-
ger?

12.4 Endelige prototype

Det lykkedes os ikke at få sammensat vores kredsløb med vores konstruerede kanon-model.

Den endelige prototype for vores kredsløb kan findes i bilag B.
Prototypen for vores kanonkonstruktion kan findes i bilag C.

13 Slut afprøvning

14 Videreudvikling

15 Prisliste

15.1 Produktionsomkostninger

For at estimerer omkostninger antages det at der skal produceres 100 stykker legetøj. Ydermere antages det at de forskellige modstande der benyttes koster omtrent det samme og at kondensatorerne koster omtrent det samme

Få skre-
vet om
hvorf-
or bolden
ikke passe-
de

Komponent	Tabel 5: Omkostningspris for prototype			
	Stk. pr. produkt	Pris stk. [USD]	Kilde	Samlet pris [USD]
PCB	-	5.81	[25]	5.81
Resistorer	20	0.01	[1]	0.2
Kondensatorer	5	0.18	[31]	0.9
MCP601	5	0.36	[22]	1.8
HEF4001	1	0.18	[7]	0.18
BPW21	1	6.26	[34]	6.26
Steppermotor	1	31	[29]	31
DC motor	2	14	[19]	28
Arduino	2	30	[27]	60
Kanon-konstruktion	1	8.81	Estimat	8.81
Total pris	-	-	-	142.95

Den totale pris angivet på Tabel 5 svarer til omtrent 970 DKK, hvilket er rigtig meget for et legetøj. Men der er væsentlige måder at sænke denne pris på. F.eks. i stedet for at benytte hele Arduino UNO benytter dens microcontroller ATmega328P. Disse kan fås til en ret lav prisleje på under 2 dollars (Se kilde: [21]). Hvilket sænker prisen med omtrent 58 USD. Dette gør at et bedre estimat svarer til 84.95 USD som er på nuværende tidspunkt ækvivalent med 578.23 DKK. Hvilket er en mere rimelig pris.

15.2 Estimering af udsalgspris

En rimelig avance vurderer vi til at være på 30%, heraf bliver den samlede salgspris:

Tabel 6: Salgspris

Omkostningspris	578.23 DKK
Avance	173.469 DKK
Pris uden moms	751.70 DKK
Pris med moms	939.625 DKK

15.3 Konkurrencedygtighed

Vores produkt minder ikke meget om nuværende alternativer til Angry Birds spillet, men det tætteste er nok modeller udviklet af LEGO. LEGO har lavet et Angry Birds spil som legetøj koncept, dog uden nogen avanceret affyringsmekanisme eller andet elektronik, hvor man kan affyrer fugle mod legoborge. Disse legomodeller kan koste hele vejen op til 700 DKK (kilde: [18]), men vi vurderer at vores produkt er i forhold til de dyre LEGOmodeller. Dog er det meget muligt at vores produkt bliver væsentligt dyre end 939.625 hvis man også skal gøre det ligeså æstetisk som LEGO's Angry birds.

16 Konklusion

17 Evaluering

Referencer

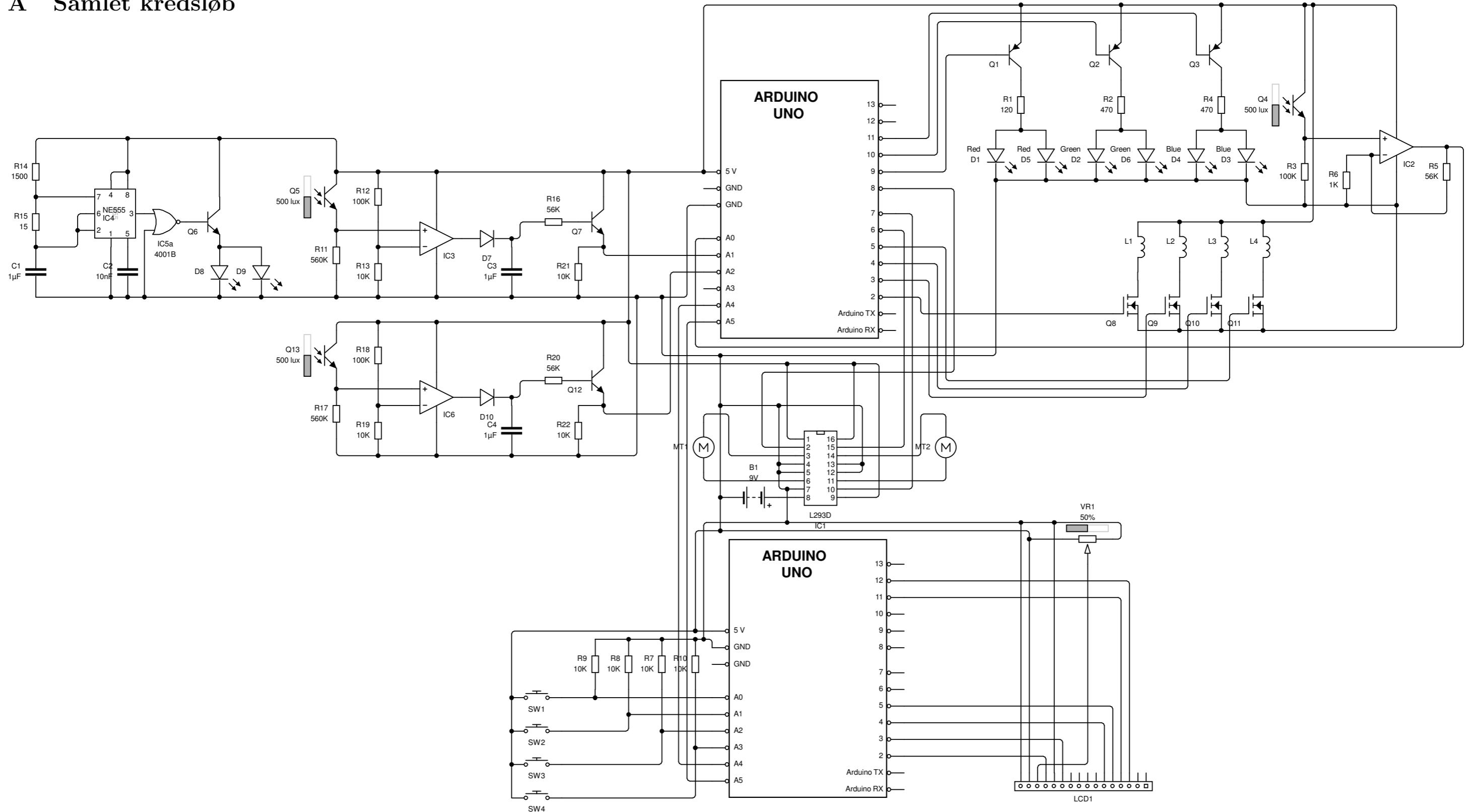
- [1] AliExpress. "*10k resistor*". URL: https://www.aliexpress.com/price/10k-resistor_price.html (sidst set 18. januar 2017). Kildekritik: digikey er en stor forhandler af elektroniske varer som gerne vil sælge så meget som muligt. Derfor kan der være mulighed for at prisen ikke inkludere moms og fragt. Udenfor dette antages det at alle modstande koster det samme.
- [2] Arduino LLC. *Analog input*. URL: <https://www.arduino.cc/en/Tutorial/AnalogInput> (sidst set 1. maj 2017). Kildekritik: Denne hjemmeside er lavet af firmaet bag arduinoen og vi vurderer at de derfor har interesse til at kommunikere hvad arduinoen er i stand til korrekt til hobbyister og andre Arduino brugere.
- [3] Arduino LLC. "*constants*". URL: <https://www.arduino.cc/en/Reference/Constants> (sidst set 14. maj 2017). Kildekritik: Denne hjemmeside er lavet af firmaet bag arduinoen og vi vurderer at de derfor har interesse til at kommunikere hvad arduinoen er i stand til korrekt til hobbyister og andre Arduino brugere.

- [4] Arduino LLC. *"Hello World!"* URL: <https://www.arduino.cc/en/Tutorial>HelloWorld> (sidst set 22. november 2016). Kildekritik: Denne hjemmeside er lavet af firmaet bag arduinoen og vi vurderer at de derfor har interesse til at kommunikere hvad arduinoen er i stand til korrekt til hobbyister og andre Arduino brugere.
- [5] Arduino LLC. *"Port Registers"*. URL: <https://www.arduino.cc/en/Reference/PortManipulation> (sidst set 14. maj 2017). Kildekritik: Denne hjemmeside er lavet af firmaet bag arduinoen og vi vurderer at de derfor har interesse til at kommunikere hvad arduinoen er i stand til korrekt til hobbyister og andre Arduino brugere.
- [6] Tom Igoe - Arduino. *One step at a time*. URL: <https://www.arduino.cc/en/Tutorial/StepperOneStepAtATime> (sidst set 28. april 2017).
- [7] Arrow. *HEF4001BT,652*. URL: <https://www.arrow.com/en/products/hef4001bt652/nexperia> (sidst set 13. maj 2017). Kildekritik: ***.
- [8] Centronic LTD. *SERIES BPW 21*. URL: <http://ticsante.u-strasbg.fr/lib/exe/fetch.php?media=0900766b800a603b.pdf> (sidst set 7. maj 2017).
- [9] CHINA YOUNG SUN LED TECHNOLOGY CO., LTD. *RED/GREEN/BLUE Triple Color LED white diffused lens (Common Cathode) - YSL-R596CR3G4B5WF12*. URL: <https://www.sparkfun.com/datasheets/Components/LED/YSL-R596CR4G3B5W-F12.pdf> (sidst set 12. maj 2017).
- [10] Dejan Nedelkovski. *"How I2C Communication Works and How To Use It with Arduino"*. URL: <http://howtomechatronics.com/tutorials/arduino/how-i2c-communication-works-and-how-to-use-it-with-arduino/> (sidst set 14. maj 2017). Kildekritik: Da det er en hjemmeside lavet af hobbyister, kan man antage at noget af teorien er lidt sjusket, men at forklaring om hvordan det bruges i praktis er rigtig.
- [11] Don Pettibone. *Re: How do I calculate the velocity of a projectile exiting an solenoid?* URL: <http://www.madsci.org/posts/archives/1998-08/897010420.Ph.r.html> (sidst set 27. marts 2017).
- [12] Electronics Tutorials. *555 Timer Tutorial*. URL: http://www.electronics-tutorials.ws/waveforms/555_timer.html (sidst set 8. maj 2017).
- [13] Fairchild Semiconductor. *BC546/547/548/549/550*. URL: <https://www.sparkfun.com/datasheets/Components/BC546.pdf> (sidst set 14. maj 2017).
- [14] Fairchild Semiconductor Corporation. *12A, 100V, 0.200 Ohm, Logic Level, N-Channel Power MOSFET*. URL: https://people.ece.cornell.edu/land/courses/ece4760/FinalProjects/s2012/tcj26_ecs227/tcj26_ecs227/F12N10L.pdf (sidst set 2. april 2017).

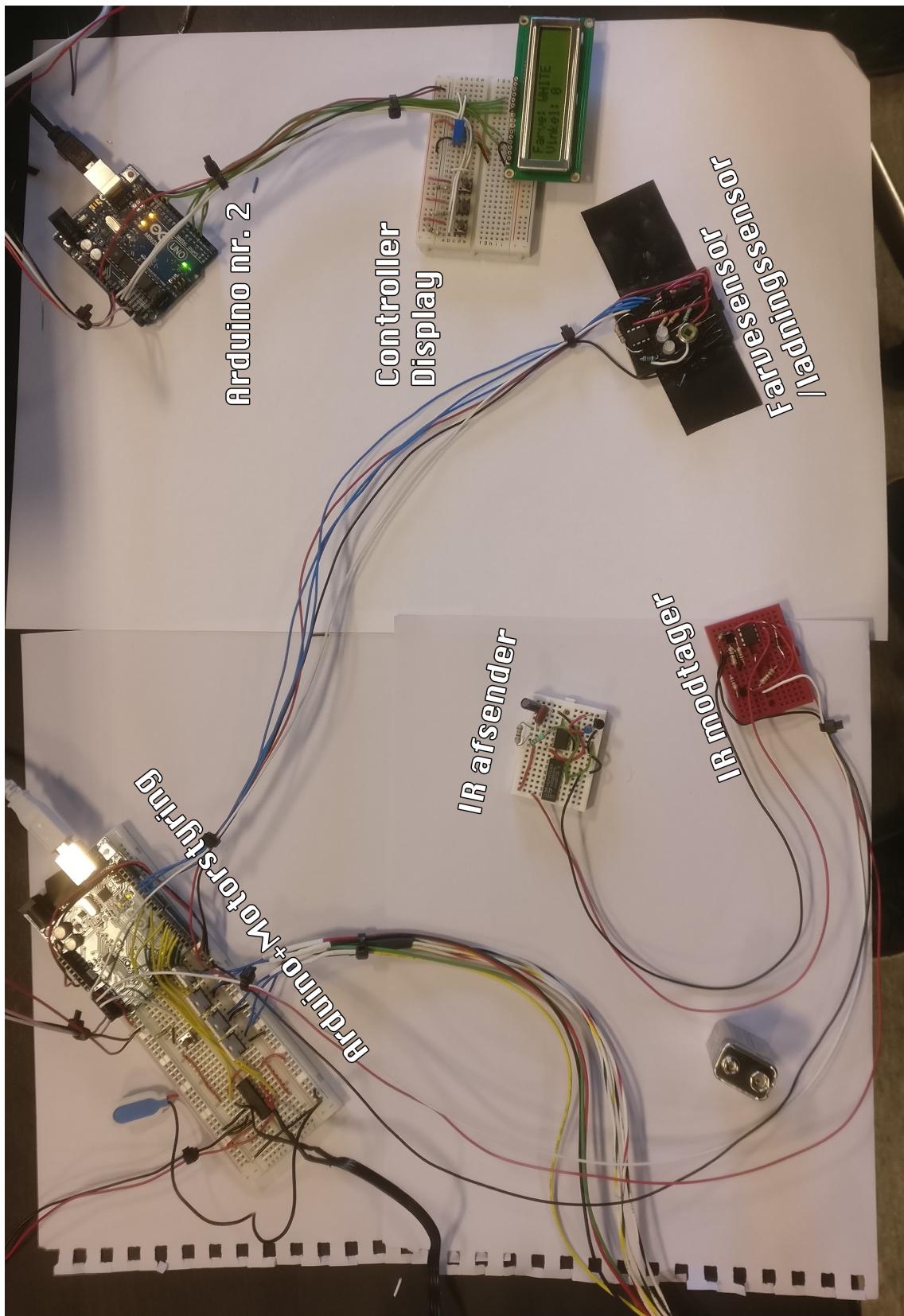
- [15] Per Holck, Jens Kraaer og Birgitte Merci Lund. *ORBIT A htx.* Skt. Pauls Gade 25, Århus, Danmark: Systime, 1930. Sider benyttet: 129.
- [16] HyperPhysics. *Photodiodes.* URL: <http://hyperphysics.phy-astr.gsu.edu/hbase/Electronic/photdet.html> (sidst set 14. maj 2017).
- [17] Kingbright. *T-1 (3mm) INFRARED EMITTING DIODE.* URL: [http://www.mouser.com/ds/2/216/L-34F3BT\(Ver.14A\)-795135.pdf](http://www.mouser.com/ds/2/216/L-34F3BT(Ver.14A)-795135.pdf) (sidst set 1. maj 2017).
- [18] LEGO System A/S. *75826 King Pig's Castle.* URL: <https://www.lego.com/en-us/angrybirdsmovie/products/king-pig-castle-75826> (sidst set 13. maj 2017). Kildekritik: ***.
- [19] LEGO System A/S. *LEGO® Power Functions XL-Motor.* URL: <https://shop.lego.com/en-DK/LEGO-Power-Functions-XL-Motor-8882> (sidst set 13. maj 2017). Kildekritik: ***.
- [20] Mehdi Sadaghdar. *Just Try and Make Your Own Coil-Gun (Accelerator).* URL: https://youtu.be/mdZo_keUoEs (sidst set 27. marts 2017). Effekt estimering baseret ud fra de givet værdier ud fra el-diagrammet på 2:31.
- [21] Microchip. *ATmega328.* URL: <http://www.microchip.com/wwwproducts/en/ATmega328> (sidst set 13. maj 2017). Kildekritik: ***.
- [22] Microchip. *"MCP601".* URL: <http://www.microchip.com/wwwproducts/en/MCP601> (sidst set 18. januar 2017). Kildekritik: ***.
- [23] Microchip. *Peakdetector OPAMP rectifier.* URL: <http://ww1.microchip.com/downloads/en/AppNotes/01353A.pdf> (sidst set 28. april 2017). Relevante sider: 8 og 9.
- [24] Microchip Technology Inc. *2.7V to 6.0V Single Supply CMOS Op Amps - MCP601/1R/2/3/4.* URL: <http://ww1.microchip.com/downloads/en/DeviceDoc/21314g.pdf> (sidst set 12. maj 2017).
- [25] multi-cb. *"PCB price calculator".* URL: <https://portal.multi-circuit-boards.eu/Customer/Index/1990438>. Kildekritik: Det er en estimering lavet af firmaet multi-cb så det er meget muligt at den estimerings-program de har benyttet er partisk i forhold til deres kommercielle interesser.
- [26] Nexperia. *HEF4001B.* URL: <http://assets.nexperia.com/documents/datasheet/HEF4001B.pdf> (sidst set 8. maj 2017).
- [27] RS Components. *Genuino Uno Rev3.* URL: <http://uk.rs-online.com/web/p/processor-microcontroller-development-kits/1240656/> (sidst set 13. maj 2017). Kildekritik: ***.

- [28] RS Components. *Hybrid stepper motors*. URL: <http://docs-europe.electrocomponents.com/webdocs/12bb/0900766b812bb588.pdf> (sidst set 28. april 2017).
- [29] RS Components. *RS Pro Hybrid Stepper Motor 1.8°, 260mNm, 5 V dc, 1 A, 8 Wires*. URL: <http://uk.rs-online.com/web/p/stepper-motors/1918328/> (sidst set 13. maj 2017). Kildekritik: ***.
- [30] RS Components. *Silicon PIN Photodiode with Daylight Filter*. URL: <http://dk.rs-online.com/web/p/fotodioder/6547921/> (sidst set 1. maj 2017).
- [31] RS components. *"Panasonic 10nF Polyester Capacitor PET 250 V dc ±5% ECQE(F) Series Through Hole"*. URL: <http://uk.rs-online.com/web/p/polyester-film-capacitors/6224634/?searchTerm=622-4634&relevancy-data=636F3D3126696E3D4931384E5> (sidst set 13. maj 2017). Kildekritik: Sandsynligvis er denne pris ikke inkluderet moms og fragt. Da RS components er en forhandler af elektroniske varer.
- [32] Texas Instruments. *H bridge*. URL: https://en.wikipedia.org/wiki/H_bridge (sidst set 24. april 2017).
- [33] Texas Instruments. *xx555 Precision Timers*. URL: <http://www.ti.com/lit/gpn/ne555> (sidst set 8. maj 2017).
- [34] Verical. *BPW21R*. URL: https://www.verical.com/pd/vishay-photodiode-BPW21R-234268?utm_source=FindChips&utm_medium=invListing&utm_campaign=FC2015 (sidst set 13. maj 2017). Kildekritik: ***.
- [35] Wikipedia. *Gausskanon*. URL: <https://da.wikipedia.org/wiki/Gausskanon> (sidst set 27. marts 2017).
- [36] Wikipedia. *L293D - Texas Instruments*. URL: <http://www.ti.com/lit/ds/symlink/l293.pdf> (sidst set 24. april 2017).

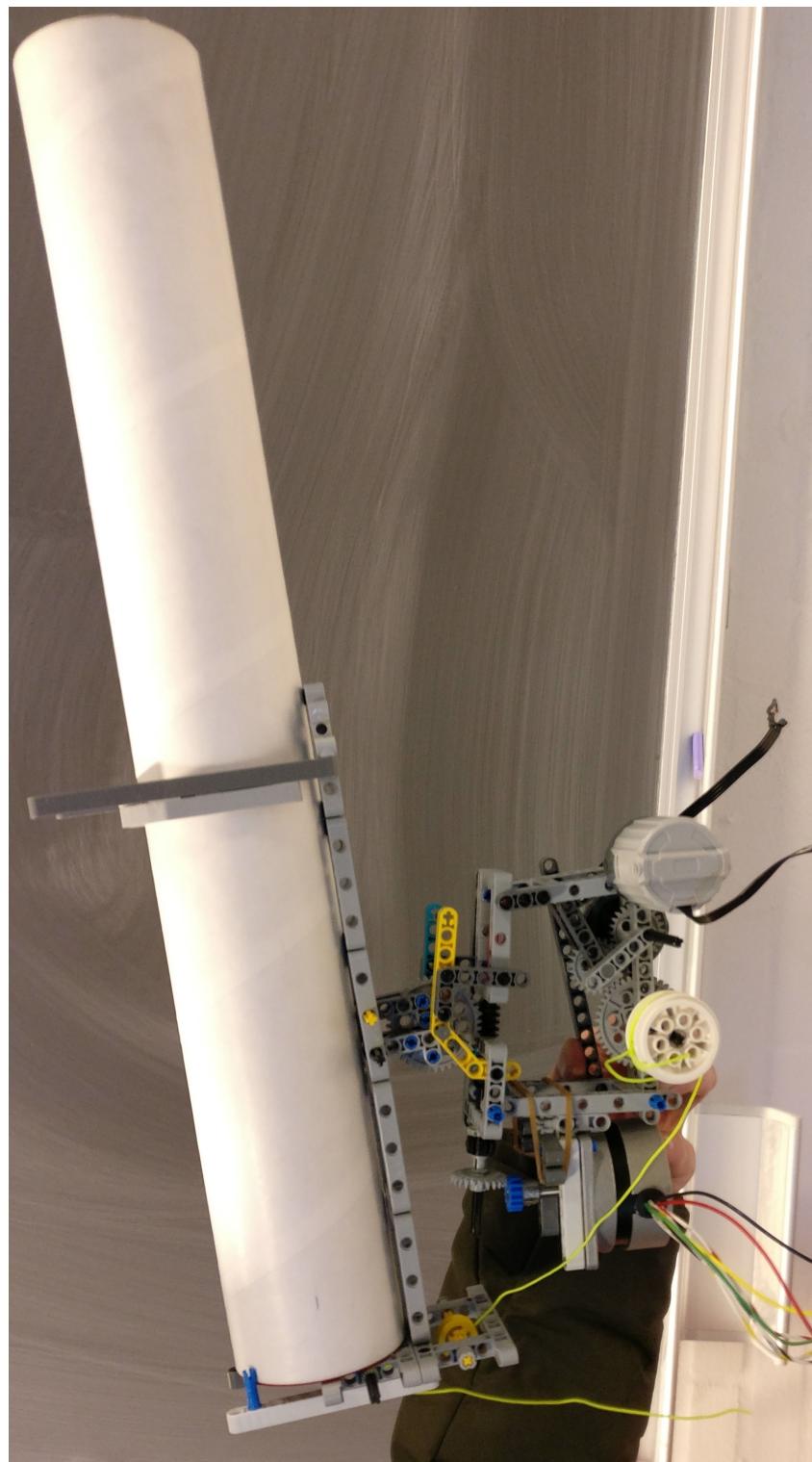
A Samlet kredsløb



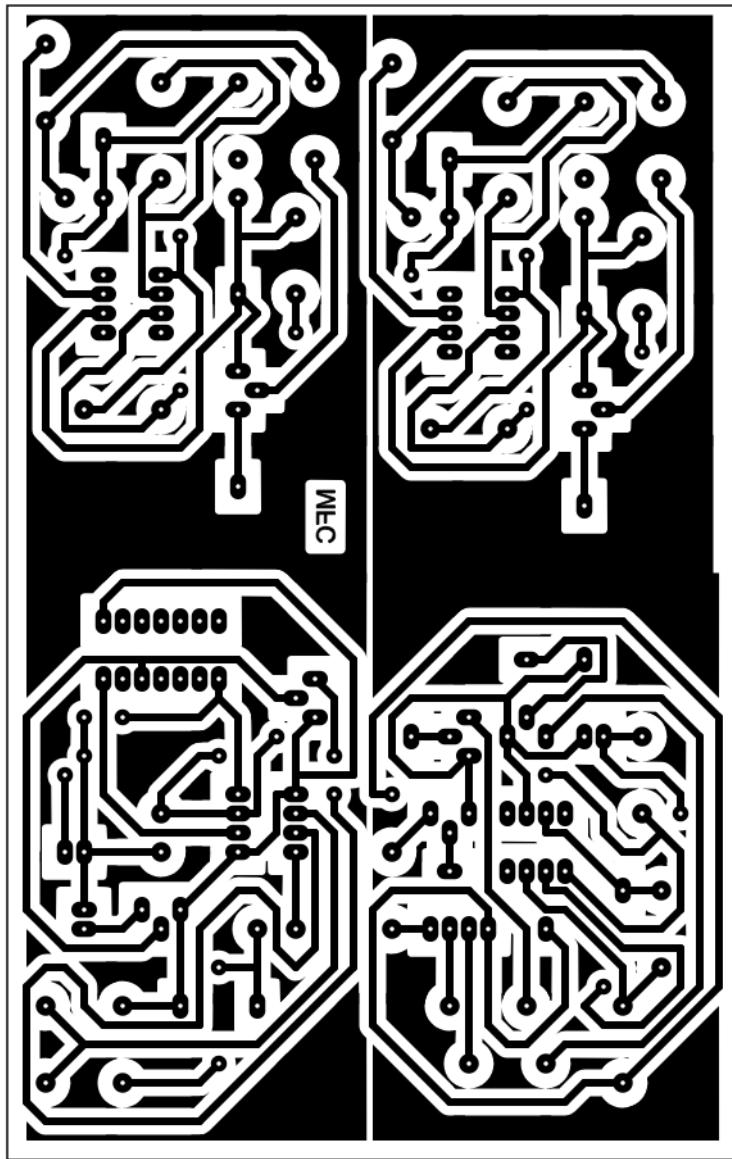
B Endelig prototype - kredsløb



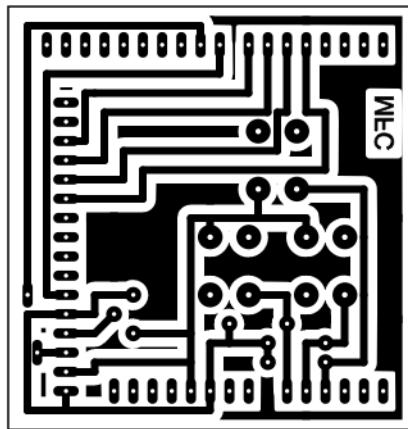
C Endelig prototype - kanonkonstruktion



D PCB artwork til hastighedsensor

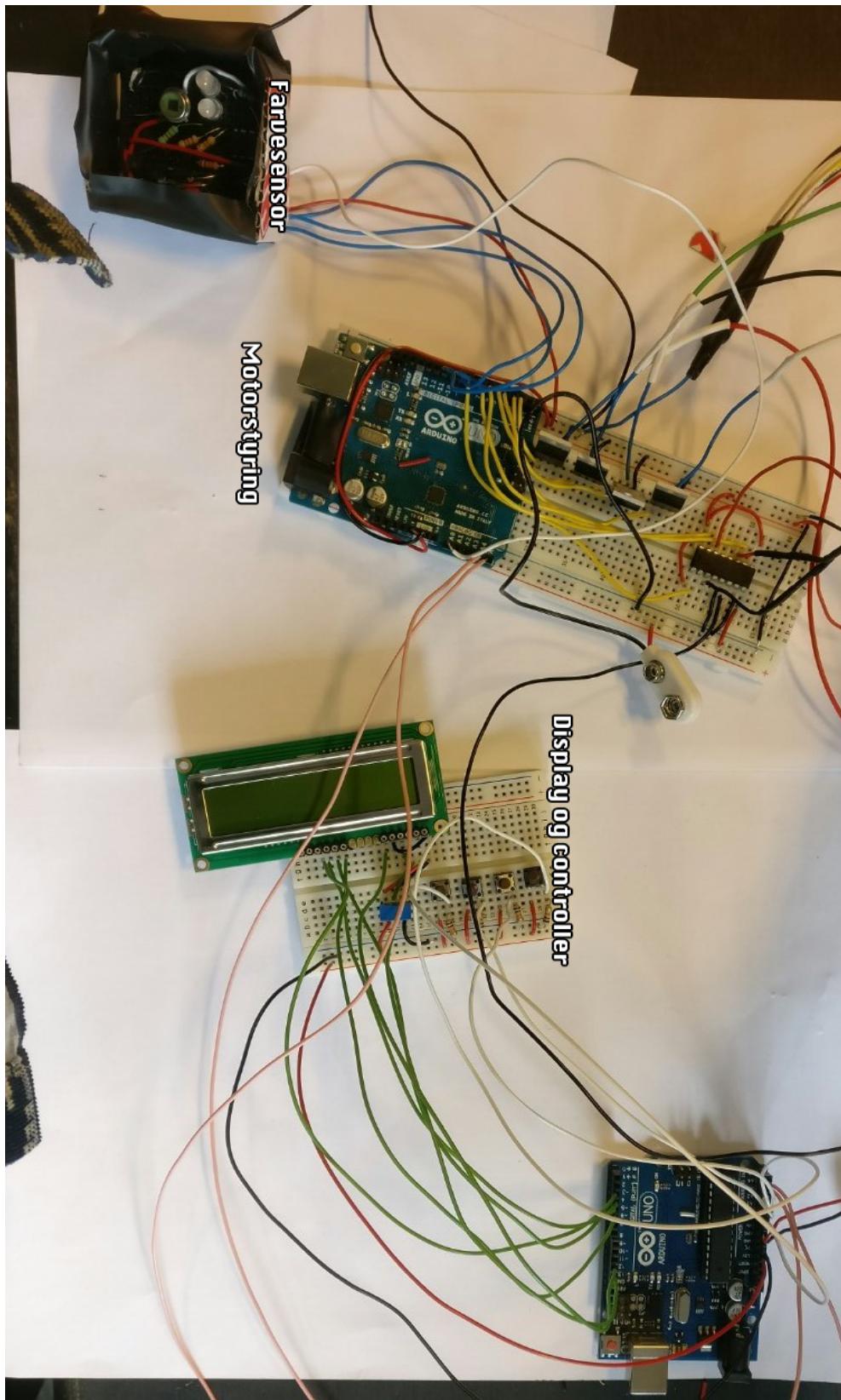


E PCB artwork til arduino shield



Måske
skul-
le vi
over-
veje at
lave
et nyt
billede
med
sam-
lede
kabler

F Fumlebrætmodel anden del



G Program til Master Arduino

```
1 #include <Wire.h>
2 #include <Stepper.h>
3
4 //Debug funktioner - Hvis debug er defineret vil der komme debug output til
5 //serial porten
6 #define DEBUG //Hvis debug er defineret. Den første funktion på hver linje
7 //bliver erstattet med den anden funktion på linjen, når programmet bliver
8 //kompileret
9 #ifdef DEBUG
10 #define DEBUG_PRINTLN(x) Serial.println(x)
11 #define DEBUG_PRINTLNF(x,y) Serial.println(x,y)
12 #define DEBUG_PRINT(x) Serial.print(x)
13 #else //Hvis debug ikke er defineret. Her gælder det samme, som betyder at
14 //funktionerne bliver fjernet
15 #define DEBUG_PRINTLN(x)
16 #define DEBUG_PRINTLNF(x,y)
17 #define DEBUG_PRINT(x)
18 #endif
19
20 //Wire - I2C protokol
21 int ADDRESS = 9; // Dens egen adresse. Ikke nødvendig, da dette er masteren
22 //af netværket, men blev sat alligevel
23 int SLAVEADD = 8; //Addressen til controlleren
24 char separatorChar = 254; //ASCII tegnet som bruges til at adskille linjerne
25 //i dataen
26 char modeChar = 50; //ASCII tegnet som indikere at det en farve der skal på
27 //displayet
28
29 //LCD variabler
30 char mode = 0; // Den sidste mode
31 String colorBuffer; //Den sidste sendte farve
32 String speedBuffer = "-1m/s"; //Den sidste sendte hastighed
33 String angleBuffer; //Den sidste sendte vinkel
34
35 //Retningsregulerende kreds
36 int stepperrev = 400; // Antal skridt per omgang
37 Stepper retning(stepperrev, 2, 3, 4, 5); //vores stepper motor objekt, med de
38 //tilsluttede ben og
```

```
31 int gearing = 40; //forholdet mellem hvor meget kannonen rygger sig og
   stepper motoren drejer
32 int stepsize = stepperrev * gearing / 180; // Skridtstoerrelsen for naar
   jystere, er lavet til at dreje 2 grad for hvert tryk
33 int currangle = 0; //Den nuvaerende vinkel i grader
34
35 //Controller
36 byte states; // det modtaget Port C input registre fra controlleren. Viser
   state af de forskellige ben
37 int upbit = 0; //Angiver hvilket bit i byten der tilhøere op knappen. analog
   A0
38 int downbit = 1; //Angiver hvilket bit i byten der tilhøere ned knappen.
   analog A1
39 int firebit = 2; //Angiver hvilket bit i byten der tilhøere skyd knappen.
   analog A2
40
41 //Hastighed
42 String Speed = "-1m/s"; //Den beregnede fart som string
43 int hastpin1 = A1; // Den ene hastighedsensor input ben
44 int hastpin2 = A2; // Den anden hastighedssensor input ben
45 float lengthtube = 1; //Laengden imellem de to hastighedssensor i meter
46
47 //Farver
48 boolean newball = true; //Angiver om der er kommet en ny bold i roeret, som
   betyder at farven skal tjekkes
49 int confirm; // Antallet af gange som sensoren har bestemt den samme farve
50 //Dette bliver brugt saa farven bliver bestemt flere gange i traek, og hvis
   det samme resultat kommer nok gange, antager vi at det den rigtige farve
51 int confirmAmount = 5; // Antallet af gange den skal faa den samme farve
52 int pinRGB[] = {12, 11, 13}; //benene til de forskellige LED'er
53 int pinColorSensor = A0; //Benet hvor farvesensoren er tilsluttet
54 int lightval[4]; //Et array til at gemme vaerdierne af farvesensoren naar den
   bliver belyst med forskellig farve. 0 er Roed, 1 er groen, 2 er blaa og 3
   er hvid
55 int fireval = 0; //Angiver hvor haardt bolden skal skydes
56 String light[] = {"RED", "GREEN", "BLUE", "WHITE"}; //Angiver de forskellige
   farvers navne
57 String color; // Den sidste besluttede farve
58
59 //Affyringsmekanisme
60 int gearpinlock = 10; //Benet som vores gear laases paa
```

```
61 int gearpinunlock = 9; //Benet som vores gear laases op paa
62 int trekpin = 6; //Benet hvor vores traekke motor er paa. Skal kun kunne
63   koere en vej
64
65 void setup() {
66   //ben tilstanden saettes for vores input og output
67   pinMode(gearpinlock, OUTPUT);
68   pinMode(gearpinunlock, OUTPUT);
69   pinMode(trekpin, OUTPUT);
70   pinMode(pinRGB[0], OUTPUT);
71   pinMode(pinRGB[1], OUTPUT);
72   pinMode(pinRGB[2], OUTPUT);
73   pinMode(pinColorSensor, INPUT);
74   pinMode(hastpin1, INPUT);
75   pinMode(hastpin2, INPUT);
76
77   //Da vi benytter NPN transistor, skal benene saettes hoej for at slukke
78   //LED'erne
79   digitalWrite(pinRGB[0], HIGH);
80   digitalWrite(pinRGB[1], HIGH);
81   digitalWrite(pinRGB[2], HIGH);
82
83   //Kommunikationen med I2C starter, med angivet addresse
84   Wire.begin(ADDRESS);
85
86   //Serial kommunikation med computeren hvis DEBUG
87 #ifdef DEBUG
88   Serial.begin(9600);
89 #endif
90
91   //Hastigheden af vores retningsregulerende motor saettes (stepper motoren)
92   retning.setSpeed(25);
93 }
94
95 void loop() {
96   //Ser om den foerste hastighedssensor er blokeret, hvilket betyder at der
97   //er en bold i roeret.
98   //Hvis der er det, saa bestem dens farve
99   if ((analogRead(hastpin1)) <= 250)determinecolor();
100  else {
101    //Ellers nulstil vaerdier til naeste bold
```

```
99     DEBUG_PRINTLN("WAITING FOR BALL");
100    newball = true;
101    confirm = 0;
102 }
103 getinput(); //Faa input fra controlleren
104 lcddisplay(); //Hvis der er noget nyt der skal paa displayet, saa skriv det
105     til controlleren
106 }
107
108 //Denne funktion modtager en byte fra controlleren, hvori at tilstanden af
109 //op, ned og skyd benet er i. Denne funktion finder ud af hvad der skal ske
110 //naar de forskellige er trykket
111 void getinput() {
112     Wire.requestFrom(SLAVEADD, 1); //Anmoder om 1 byte data fra controlleren
113
114     //Hvis noget data bliver sendt, saa set det i states
115     if (Wire.available()) states = Wire.read();
116     else return;
117
118     //Udskriver hvad vi har modtaget binaert
119     DEBUG_PRINT("Recieved: ");
120     DEBUG_PRINTLNF(states, BIN);
121
122     //Ser om op knappen er trykket, og ned knappen ikke er trykket
123     if (bitRead(states, upbit) && !bitRead(states, downbit)) {
124         //Hvis den er det, saa skal retningsmotoren koere 1 grad op
125         DEBUG_PRINT("Updated angle from ");
126         DEBUG_PRINT(angle()); //Funktion til at udskrive vinklen som en string
127         DEBUG_PRINT(" to ");
128         currangle += 2; //Aendre den nuvaerende vinkel med 2 grad
129         DEBUG_PRINT(angle()); //Udskriver den nye vinkel
130         DEBUG_PRINTLN();
131         retning.step(stepsizes); //Saetter retningsmotoren til at koere 2 grad op
132     }
133     //Ser om op knappen er trykket, og ned knappen ikke er trykket
134     if (bitRead(states, downbit) && !bitRead(states, upbit)) {
135         //Hvis den er det, saa skal retningsmotoren koere 1 grad ned
136         DEBUG_PRINT("Updated angle from ");
137         DEBUG_PRINT(angle());
138         DEBUG_PRINT(" to ");
```

```
137     currangle -= 2; //Aendre den nuvaerende vinkel med -2 grad
138     DEBUG_PRINT(angle());
139     DEBUG_PRINTLN();
140     retning.step(-stepsize); //Saetter retningsmotoren til at koere 2 grad ned
141 }
142 if (bitRead(states, firebit))fire(); //Kalder affyringsfunktionen hvis skyd
143 knappen er trykket
144 }
145 // Funktionen som skyder bolden afsted
146 void fire() {
147     DEBUG_PRINTLN("Firing");
148     if (!(analogRead(hastpin1) <= 250)) { //Hvis den ene hastighedssensor er
149         hoej, saa er der ikke en bold i roeret, som betyder at affyringen skal
150         stoppes
151         DEBUG_PRINTLN("NOT LOADED");
152         return;
153     }
154     if (newball) { //Hvis boldens farve stadig ikke er bekraeftet, saa stop
155         affyringen
156         DEBUG_PRINTLN("COLOR NOT VERIFIED");
157         return;
158     }
159     //Skriver indstillingen (farven) til Serialporten
160     DEBUG_PRINT("Setting: ");
161     DEBUG_PRINTLN(color);
162
163     long time = millis(); //den nuvaerende tid i millis
164     digitalWrite(trekpin, HIGH); //Saetter traek bennet hoej, saa elastiken
165     bliver strukket ud
166     digitalWrite(gearpinlock, 130); //Saetter den i gear
167     while (millis() - time <= fireval * 1000); // Treaker i det stykke tid der
168     svare til indstillingen
169     // Der laases nu op, ved at saette den i frigear
170     DEBUG_PRINTLN("UNLOCKING");
171     digitalWrite(gearpinlock, LOW);
172     digitalWrite(gearpinunlock, 130); //traekker gearet ud i frigear
173     digitalWrite(trekpin, LOW);
174
175     time = millis(); //Vi opdatere tiden
176     determinespeed(); //Funktionen der bestemmer boldens hastighed
```

```
172     while (millis() - time <= 2000); //Saa laenge der ikke er gaaet 2 sekunder,  
173         saa vent, Dette er for at soerge for at gearet bliver holdt i frigear i  
174             lang nok tid  
175             digitalWrite(gearpinunlock, LOW); //Stopper med at holde gearet fast i  
176                 frigear  
177 }  
178  
179  
180  
181 String angle() { //Returnere vinklen som en string  
182     return String(currangle);  
183 }  
184  
185 void determinecolor() { // Bestemmer farven af bolden i roeret  
186     if (!newball) return; // Hvis farven allerede er bestem og bekraeftet, saa  
187         lad vaere med at tjekke farven  
188         DEBUG_PRINTLN("DETERMINE COLOR");  
189         //Taender for alle farverne  
190         for (int i = 0; i < 3; i++) {  
191             digitalWrite(pinRGB[i], LOW);  
192         }  
193         delay(10);  
194         lightval[3] = analogRead(pinColorSensor); //gemmer farvesensorens vaerdi i  
195             lightval[3]  
196             //slukker alle LED'erne igen  
197             for (int i = 0; i < 3; i++) {  
198                 digitalWrite(pinRGB[i], HIGH);  
199             }  
200             if (lightval[3] > 770) { //Hvis spaendingen er højere end 3.8 V er bolden  
201                 hvid  
202                 setColor(light[3], 1); //Saetter farven til hvid  
203                 return;  
204             }  
205             for (int i = 0; i < 3; i++) { //Gaar igennem alle LEDerne og taender og  
206                 slukker dem en efter en, og gemmer farvesensor vaerdien  
207                 digitalWrite(pinRGB[i], LOW); //Taender for LEDen  
208                 delay(10);  
209                 lightval[i] = analogRead(pinColorSensor); //gemmer vaerdien i lightval  
210                 DEBUG_PRINT(light[i]);  
211                 DEBUG_PRINT(" Value: ");
```

```
206     DEBUG_PRINTLN(lightval[i]);
207     digitalWrite(pinRGB[i], HIGH); //slukker LEDen igen
208 }
209
210 if (lightval[0] * 1.2 > lightval[1] && lightval[0] * 1.2 > lightval[2]) {
211     //Tjekker om farven er roed
212     setColor(light[0], 2); //Saetter farven til roed og skydevardien til 2
213     return;
214 }
215 if (lightval[1] > lightval[2] && lightval[1] > lightval[0]) { //Tjekker om
216     farven er groen
217     setColor(light[1], 3); //Saetter farven til roed og skydevardien til 3
218     return;
219 }
220 if (lightval[2] > lightval[0] && lightval[2] > lightval[1]) { //Tjeker om
221     farven er blaau
222     setColor(light[2], 4); //Saetter farven til roed og skydevardien til 4
223     return;
224 }
225 void setColor(String newColor, int newFireval) { // Denne funktion saetter
226     farven, og tjekker bekraeftelse
227     if (color == newColor) { //Hvis den forrige bold var samme farve, saa skal
228         confirm blive 1 hoejere
229         confirm += 1;
230         if (confirm == confirmAmount) newball = false; //Hvis antallet af gange
231             boldens farve er blevet bekraeftet, saa saet newball falsk
232     } else { // Ellers genstart bekraeftelses processen og saet farven til
233         newColor
234         confirm = 0;
235         color = newColor;
236     }
237     fireval = newFireval; //Tiden som elastikken skal blive traekket i bliver
238     sat til newFireval
239     DEBUG_PRINT("COLOR DETECTED : ");
240     DEBUG_PRINTLN(color); //Skriver den fundede farve til Serialporten
241 }
242
243 void determinespeed() { // Bestemmer hastigheden af bolden
244     DEBUG_PRINTLN("DETERMING SPEED");
```

```
239     unsigned long time = millis(); //den nuvaerende tid saettes i time
240     long calc; //en vaerdi til at se hvornaar den foerste hastighedssensor kan
241         //se bolden
242     while (millis() - time <= 3000 && !(analogRead(hastpin1) <= 250)); //Venter
243         til den foerste hastighedssensor bliver blokeret eller der er gaeet 3
244         sekunder
245     calc = millis(); //saetter calc til den nuvaerende tid
246     while (millis() - time <= 3000 && !digitalRead(hastpin2)); //Venter til den
247         anden hastighedssensor bliver blokeret eller der er gaet 3 sekunder fra
248         funktionen startede
249     if (millis() - time >= 3000) { //tjekker om der er gaaet 3 sekunder, som
250         betyder at der er TIMEOUT
251         return;
252     }
253     Speed = String(lenghttube / (((float)(millis() - calc)) / 1000.0), 1);
254         //Beregner hastigheden og laver det om til en String med 1 decimal
255         precision
256     Speed += "m/s";
257     DEBUG_PRINTLN("Updated speed: " + Speed);
258 }
259
260 void lcddisplay() { //Opdatere displayet
261     String data; //Stringen som skal sendes. Denne bliver bygget paa igennem
262         funktionen
263     boolean changed = false; //Angiver om noget har aendret sig paa displayet
264         siden den sidst blev opdateret
265         //Ser om farven har aendret sig
266         if (colorBuffer != color) {
267             mode = modeChar; //tilstanden bliver sat til farv
268             data += mode; //saetter moden paa data stringen
269             colorBuffer = color; //laver farve bufferen om til den nuvaerende farve
270             data += colorBuffer; //saetter farvebufferen paa data stringen
271             changed = true; //Angiver at noget har aendret sig
272         }
273         //Ser om hastigheden har aendret sig
274         else if (speedBuffer != Speed) {
275             mode = modeChar + 1; //tilstanden bliver sat til hastighed (noget andet
276             end modeChar)
277             data += mode; //Saetter moden paa data stringen
278             speedBuffer = Speed; //Laver hastighedsbufferen om til den nuvaerende
279             hastighed
```

```
268     data += speedBuffer; //Saetter hastighedsbufferen paa data stringen
269     changed = true; //angiver at noget har aendret sig
270 } else { //Ellers fyldes data stringen op med de sidst gemte vaerdier.
271     Dette er vist anden linje skal opdateres
272     data += (char)mode;
273     if (mode == modeChar) data += colorBuffer;
274     else data += speedBuffer;
275 }
276     data += seperatorChar; //Saetter en seperator paa enden af data stringen
277     if (angleBuffer != angle() || changed) {// Hvis vinklen ikke er den samme
278         eller hvis foerste linje har andret sig
279         angleBuffer = angle(); //Saetter vinkelbufferen til den nuvaerende vinkel
280         data += angleBuffer; //Saetter vinkelbufferen paa data stringen
281         changed = true; //Angiver at noget har aendret sig
282     }
283     data += seperatorChar;//Saetter endnu et seperationstegn
284     if (changed) { //Hvis noget har aendret sig
285         DEBUG_PRINT("TRANSMITTING: ");
286         DEBUG_PRINTLN(data);
287
288         //Laver data stringen om til et array af char
289         char dataBuffer[data.length()];
290         data.toCharArray(dataBuffer, data.length());
291
292         //Begynder kommunikation til controlleren, og overfoere dataen som et char
293         array
294         Wire.beginTransmission(SLAVEADD);
295         Wire.write(dataBuffer);
296         Wire.endTransmission();
297     }
298     delay(100); //Venter i 0.1 sekund for at forhindre overbelastning af I2C
299     bufferen
300 }
```

H Program til Controller Arduino

```
1 #include <LiquidCrystal.h>
2 #include <Wire.h>
3
```

```
4
5 //Debug funktioner - Hvis debug er defineret vil der komme debug output til
6   serial porten
7 #define DEBUG
8 #ifdef DEBUG //Hvis debug er defineret. Den foerste funktion paa hver linje
9   bliver erstattet med den anden funktion paa linjen, naar programmet bliver
10  kompileret
11 #define DEBUG_PRINTLN(x) Serial.println(x)
12 #define DEBUG_PRINTLNF(x,y) Serial.println(x,y)
13 #define DEBUG_PRINT(x) Serial.print(x)
14 #else //Hvis debug ikke er defineret. Her gaelder det samme, som betyder at
15   funktionerne bliver fjernet
16 #define DEBUG_PRINTLN(x)
17 #define DEBUG_PRINTLNF(x,y)
18 #define DEBUG_PRINT(x)
19 #endif
20
21
22 //Wire - I2C protokol
23 int ADDRESS = 8; // Dens egen adresse. Denne skal bruges naar master vil
24   kontakte controlleren
25 char separatorChar = 254; // ASCII tegnet som bruges til at adskille linjerne
26   i dataen
27 char modeChar = 50; //ASCII tegnet som indikere at det en farve der skal paa
28   displayet
29
30 //knapper til controlleren. Benene bliver kun brugt til at angive at det er
31   input, ellers benyttes port manipulation (PINC) til at sende benenes
32   tilstande som en byte.
33 int uppin = A0;
34 int downpin = A1;
35 int skydpin = A2;
36
37 //LCD variabler
38 LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //vores LCD objekt, med benene som den
39   er koblet paa
40 String line1_const1 = "Farve: "; //Den ene konstant paa linje 1. Skal have et
41   mellemrum efter, da .length bliver brugt
42 String line1_const2 = "Fart: "; //Den anden konstant paa linje 1. Der skiftes
43   mellem disse med mode i modtager funktionen
44 String line2_const = "Vinkel: "; //Den ene konstant paa linje 2.
```

```
33
34 void setup() {
35     //ben tilstand for knapperne saettes som input
36     pinMode(uppin, INPUT);
37     pinMode(downpin, INPUT);
38     pinMode(skydpin, INPUT);
39
40     //Starter I2C kommunikation
41     Wire.begin(ADDRESS); //Angiver adresse og begynder
42     Wire.onRequest(requestEvent); //Angiver funktion der skal kaldes naar
        controlleren bliver bedt om at sende
43     Wire.onReceive(receiveEvent); //Angiver funktion der skal kaldes naar
        controlleren skal modtage
44
45     //Starter LCD
46     lcd.begin(16, 2); //Begynder LCD'en med 2 linjer og 16 tegn per linje
47     lcd.print(line1_const1); //Skriver 1. linje konstant paa LCD
48     lcd.setCursor(0, 1); //Skifter position til 2. linje
49     lcd.print(line2_const); //Skriver 2. linje konstant paa LCD
50
51     //Starter serialkommunikation hvis DEBUG er defineret
52 #ifdef DEBUG
53     Serial.begin(9600);
54 #endif
55 }
56 // Loop funktionen, controlleren skal ikke gøre noget hvis den ikke bliver
    bedt om det
57 void loop() {
58 }
59
60 //Naar controlleren skal sende data
61 void requestEvent() {
62     byte data = PINC; //Port C input registret
63     //Skriver den sendte data binaert i serialkommunikationen hvis DEBUG er
        defineret
64     DEBUG_PRINT("TRANSMITTING DATA: ");
65     DEBUG_PRINTLN(data, BIN);
66     DEBUG_PRINTLN();
67
68     Wire.write(data); //Sender dataen over I2C
69 }
```

```
70
71 //Naar controlleren modtager data
72 void receiveEvent(int datalength) {
73     DEBUG_PRINTLN("RECIEVING DATA: ");
74
75     boolean mode = (Wire.read() == modeChar); //Den foerste byte indikere
76         tilstand, hvor at den enden kan vaere 0 eller hoejere end 0
77
78     //Stringene for den 1. og 2. linje
79     String firstLineBuffer = getLine(); //Faar den 1. linje fra I2C bufferen
80     String secondLineBuffer = getLine(); //Faar den 2. linje fra I2C bufferen
81
82     setDisplay(mode, firstLineBuffer, secondLineBuffer); // Det modtaget data
83         bliver sendt til funktionen setDisplay, som skriver dataen til displayet
84 }
85
86 //Denne funktion laeser fra I2C bufferen indtil at den er tom eller at ny
87     linje tegnet dukker op. Dette bliver sat i en String som bliver returneret
88 String getLine() {
89     String output;
90     while (Wire.available()) { //Saa laenge der er noget i I2C bufferen
91         char c = Wire.read(); //laes det naeste tegn
92         if (c == separatorChar) break; //Hvis tegnet er seperationstegnet, stop
93             med at laese fra bufferen
94         DEBUG_PRINT(c); //Skriv til Serial hvis DEBUG
95         output += c; //Ellers skal tegnet tilfoejes til output
96     }
97     DEBUG_PRINTLN();
98     return output; //returner outputtet
99 }
100
101 //opdatere displayet
102 void setDisplay(boolean mode, String line1, String line2) {
103
104     //Debug udskrivning. Skriver hvad displayet bliver opdateret til
105     DEBUG_PRINT("Updating display: ");
106     DEBUG_PRINT("Line 1: ");
107     DEBUG_PRINT(line1);
108     DEBUG_PRINT("  ");
109     DEBUG_PRINT("Line 2: ");
110     DEBUG_PRINTLN(line2);
```

```
107 lcd.clear(); //Fjerner alt fra displayet
108 lcd.setCursor(0, 0); //Saetter position til starten af displayet paa 1.
109     linje
110     int leng; //laengde paa 1. linje. Dette bruges til at goere resten af
111         felterne tomme paa linjen tomme
112
113 // Skriver konstanten paa 1. linje, alt efter hvilken mode det er
114     (hastighed/farve)
115     if (mode) {
116         lcd.print(line1_const1);
117         leng += line1_const1.length();
118     }
119     else {
120         lcd.print(line1_const2);
121         leng += line1_const2.length();
122     }
123
124     lcd.print(line1); // Skriver dataen til linje 1 paa linje 1.
125     lcd.setCursor(0, 1); // Saetter positionen til 2. linje efter konstanten
126     lcd.print(line2_const);
127     lcd.print(line2); //Skriver dataen til linje 2 paa linje 2.
128 }
```

I Logbog

Uge 10

I dag har vi researched en del med hensyn til modellering af gausskanoner.
Generelt beregninger.

Vi har fået lavet vores projektbeskrivelse.

Vigtig info

Vi fandt en gangske fin model udformet af Don Pettibone: <http://www.madsci.org/posts/archives/1998-08/897010420.Ph.r.html>. Den skal tilpasses magneten i stedet for jern og tilpasses til at vi bruger flere modeller. En beregnet udgave ligger under misc - beregninger.

Beregnsel af inductance: <http://electronbunker.ca/eb/InductanceCalc.html>

Mangler

Forfinet modellen, givet af Don Pettibone.

Uge 11

I dag har vi fået opstil en model til at bestemme temperaturen af spolen som funktion af strømstyrken og tiden den er tændt.

Vi fik ikke lavet videre på modellen givet af Don Pettibone, da modellen virkede optimistisk med, hensyn til kraften som metalkuglen bliver påvirket. Dette ledte os til at opstillet nogle simple forsøg med en spole med 200 vindinger. Ud fra disse forsøg har det vist sig at det er tilnærmelsesvis umuligt at lave et kraftigt nok magnetfelt inde i spolen, til at skubbe/trække i en metalkugle på 3 mm i radius. Dette forsøg var lavet med en spole på en ydre diameter på 32 mm. Dette har ledt til overvejelser om at placere en spole med jernkerne bag kanonrøret, og skubbe en magnet ud af røret i stedet for at accelerere et normalt metalstykke.

Vigtig info

Magnetfeltet af en normal spole er ikke kraftig nok, og derfor skal der laves forsøg med spole der har en jernkerne, i stedet for en normal spole. Dette betyder at projektillet skal være magnetisk.

Mangler

Forfinet modellen, givet af Don Pettibone. Forsøg med jernkerne i spole, og magnet som projektil.

Uge 12

Vi har besluttet at skrætte gausskanonen og indføre elastikkanon i stedet. Vi skal således få el ind i vores projekt ved at gøre hastighedssensoren mere kompleks, ved at gøre brug af 38 kHz sensor kreds. Hvor vi gøre brug af en pingpong kugle som projektil i stedet for en magnet eller metalstykke.

Vigtig info

Vi skal lave en kompleks bandpass baseret IR måler. Gausskanonen skråttet til fordel for elastikanonen.

Mangler

Uge 13

Referent

Christoffer

Tilstedeværende

Alle

Dagens arbejde og beslutninger

Rumle: Skrev et program og satte kredsløbet op for steppermotoren og har lavet controlleren.

Daniel: Vi har bygget kanonen og skydemekanismen

Christoffer: Vi har fået skrevet i rapporten med hensyn til overordnede løsninger.

Vigtig info

Næste gang skal vi genoverveje vores konstruktionsvalg og retningsregulerende kreds, idet det er lidt buggy.

Mangler

Elastik kanon-afsnittet skal skrives færdigt næste gang.

Number 2

Referent

Christoffer

Tilstedeværende

Alle

Dagens arbejde og beslutninger

Vi er påbegyndt hastighedssensor. Overordnet Løsningsforslag er omrent færdigt. Kredsløbet i virkeligheden for controlleren, triggermekanismen og motorstyringen.

Vigtig info

Der er beregninger og dokumentation af 555 timer beregninger i misc. Kredsløbet for timeren for vores hastighedssensor er lavet i LiveWire (Daniel har det).

Mangler

Vi skal få set på resten af hastighedssensoren.

Dato	3. april
Tilstede	Alle
Referent	Christoffer
Dagens arbejde og beslutninger	<p>Vi har besluttet at opbygningen på rapporten skal være for hver blok</p> <ul style="list-style-type: none">- Beskrivelse af kreds og indgående led- Beskrivelse af hver komponent ifølge datasheet- Teori for hver komponent- Test og beregning af hver komponent <p>Vi har lavet en peakdetector</p> <p>Vi har testet infrarøde komponenter og har set hvor godt de virker.</p> <p>Der er lavet tre livewire drawings af retnings, hastigheds og kontroller kredsløb.</p>
Laves inden næste gang	
Laves næste gang	Vi skal huske kilder på vores datasheets
Spørgsmål til vejleder	

Dato	20. april
Tilstede	Alle
Referent	Christoffer
Dagens arbejde og beslutninger	Der er blevet klarlagt vores todo list. Simon er begyndt på farvesensor, han er også som konsekvens af dette blevet blind. 555-timer er der blevet beregnet mål og er blevet bygget (se beregninger under misc) for at fundet datasheet for vores LED og vi har af Lone på

	lectio fået IR modtager Datasheet. Dagen blev gjort kort pga. SO prøvemappe
Laves inden næste gang	
Laves næste gang	
Spørgsmål til vejleder	

Dato	24. april
Tilstede	Alle
Referent	Christoffer
Dagens arbejde og beslutninger	Mål: <ol style="list-style-type: none">1) Daniel 555 timer skal færdiggøres2) Fortsætte på mål3) Christoffer Rapport <p>Nået:</p> <ol style="list-style-type: none">1) Vi fik færdiggjort vores IR sensor og vi skal færdiggøre vores modtager. Vi skal også færdiggøre vores RGB sensor2) Vi har store problemer med RGB sensoren.3) Der er lavet opdateret målinger i maple af modstandende der virker.4) Der er påbegyndt på test og beregninger i hastighedssensor-afsnittet.
Laves inden næste gang	
Laves næste gang	
Spørgsmål til vejleder	

Dato	26. april
Tilstede	Alle
Referent	Christoffer
Dagens arbejde og beslutninger	I går: vi lavet tests for farvesensoren og IRsensor. Vi er begyndt på display. Vi er gået i gang med at få lavet forstærker til farvesensoren. I dag: Vi skal have fået et display til at virke, vi skal have lavet et livewirediagram af alle vores kredse og få lavet PCB - templates til i morgen
Laves inden næste gang	
Laves næste gang	
Spørgsmål til vejleder	

Dato	27. april
Tilstede	Daniel, Christoffer
Referent	Christoffer
Dagens arbejde og beslutninger	Vi har fået lavet alle kredsløb på fumlebræt og vi har lavet et print, der under fremstillingen mislykkedes.
Laves inden næste gang	
Laves næste gang	PRINT og færdiggørelse af produkt. DOKUMENTATION - Målinger, beregninger, kredse i rapport
Spørgsmål til vejleder	

--	--

Dato	28. april
Tilstede	ALLE
Referent	Christoffer
Dagens arbejde og beslutninger	<p>Vi har lavet to print, ingen af dem virker. Alle delkredsløb virker på fumlebræt. Vi mangler et ekstra kredsløb til hastighedsmåleren.</p> <p>MORALE STATUS: Qutie low.</p> <p>NUVÆRENDE FOKUSOMRÅDE: Rapport udkast med tre hovedpunkter 1. Målinger, 2. Beregninger, 3. Kredsløb.</p>
Laves inden næste gang	
Laves næste gang	
Spørgsmål til vejleder	

Dato	8. maj
Tilstede	ALLE
Referent	Christoffer
Dagens arbejde og beslutninger	<p>Vi har forsøgt at få vores fumlebræt implementeret på kanonen. Desværre virker det ikke. Sandsynligvis pga. formen og størrelsen på røret, så farver ikke kan registreres.</p> <p>Vi har også fået skrevet en del rapport, men vi har problemer med GitHub og onedrive.</p>

Laves inden næste gang	
Laves næste gang	
Spørgsmål til vejleder	