**Dampmaskinesimulator**

**Dokumentation**

***Det Gamle Værft 2016/17***

*Jørgen Friis*

Indholdsfortegnelse

Simulator Hardware 4

Forsyningsspænding 4

Lavspænding 7

Servodriver 8

Viserinstrumenter 8

Omstyring 8

Lydanlæg 8

Røganlæg 9

Ventilation 9

Varmluftblæser 9

Flammelys 9

Overtryksventil 10

Maskintelegraf 10

Omstyring 11

Oliepumpe 11

Dampmaskinemodel 11

Skueglas for olie 12

Transportbånd 12

Båndet 12

Vejeceller 12

Frekvensomformer 13

Kedelrør 15

Indfyringslåge 15

Spjæld for primær forbrændingsluft 15

Skueglas for kedelvand 15

Haner for kedelvand 16

Haner for kondensatoren 16

TV skærm i koøjet 16

Damphaner 17

Computer Hardware 17

Raspberry Pi 18

Opsætning af Raspberry Pi 3 computer 18

I2C-bus 22

Arduino Nano 26

Algoritme 26

Designparametre 26

Driftsparametre 26

Lovmæssigheder 26

Computer Software 26

Raspberry Pi 26

Powernet.py 27

Model.py 27

PCA9685.py 27

Servo.py 27

Transport.py 27

Vejecelle.py 28

Arduino 28

Powernet.ino 28

Model\_computer.ino 28

Slave\_transport.ino 28

Vejeceller.ino 28

Programkode 29

Kredsløbsdiagrammer 29

Bilag 30

# Simulator Hardware

De første syv afsnit omhandler de generelle funktioner, der sidder på hovedtavlen og i kontrolrummet bag ved keddelskottet.

Simulatorens hardware er derefter beskrevet i læseretningen, når man står i maskinrummet og ser på kedlen.

## Forsyningsspænding

Dampskibssimulatoren forsynes med 230 V AC med jord fra stikkontakt i svejserummet. Det eneste, der er sluttet direkte til nettet er en 24 V DC strømforsyning, Raspberry Pi og ATX forsyningen.

Med 24 V forsynes en knivafbryder placeret på agterskottet. Knivafbryderen har 4 mulige stillinger, der starter simulatoren i fire forskellige tilstande:

1. Maskinsimulator med realistisk respons
2. Maskinsimulatot + skibssimulator (styrhus) med realistisk respons
3. Maskinsimulator + skibssimulator (styrhus) med pædagogisk respons
4. Maskinsimulator med pædagogisk respons

De fire sluttekontakter går via en diodeseperation til en fælles forsyning af et 24 V DC/230 V AC relæ (#1), der tænder for strømmen til transportbåndet. Endvidere leveres 24 V DC til en glødelampe i maskinrummet, der indikerer, at der er spænding på simulatoren.

Før diodeseparatoren aflæses (Arduino slave 0x43) hvilken stilling knivafbryderen er sluttet i via et "prelfilter" og en spændingsdeler. Arduinoen er placeret inde i knivafbryderen.

Øvrige enheders forsyningsspænding samt aflæsning af knivafbryderen styres af en slave arduino (0x10) på hovedtavlen, som det fremgår af nedenstående tabel:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Enhed** | **Arduino PIN** | **Relæ #** | **Kode for tænd** | **Kode for sluk** |
| Spændingsforsyning | Output |  |  |  |
| Raspberry Pi + ATX | - | - | - | - |
| Transportbånd 230 V AC | - | 1 | - | - |
| Ledig | D2 | 2 | 1 | 2 |
| Røgmaskine 230 V AC | D3 | 3 | 3 | 4 |
| Røgmaskine aktivering | D4 | 9 | 11 | 12 |
| Varmeblæser 230 V AC | D5 | 4 | 7 | 8 |
| Lydanlæg 230 V AC | D6 | 5 | 9 | 10 |
| TV-skærm 230 V AC | D7 | 6 | 5 | 6 |
| Ventilation 230 V AC | D8 | 7 | 19 | 20 |
| Skibsbro 230 V AC | D9 | 8 | 17 | 18 |
| Simulator klar: 24 V DC | D10 | 10 | 13 | 14 |
| Skibsbro 24 + 12 V DC | D11 | 11 | 15 | 16 |

De 10 relæer, der trækkes af Arduinoen er koblet som det ses på næste side. De hedder AR322173, og de kan klare en belastning på 10 A, 250 V AC. De er konfigureret med spolen mellem ben 1 og 6, og en afbryder mellem ben 2 og 3 og en anden afbryder mellem ben 4 og 5.

Relæ-spolens modstand er målt til 137 Ohm, og stømforbruget er målt til 90 mA ved en forsyning på 12 V DC. Beregningsmæssigt er strømmen I =U/R, dvs I = 12/137 = 88 mA, så vi regner med Ic = 90 mA i det følgende.

Som transistor vælges 2N2222, der har en max Ic på 0,6 A, og en hFE (DC Collector/Base Gain hFE Min) på 75, hvilket giver en Ib = Ic/hFE = 0,09/75 = 1,2 mA.

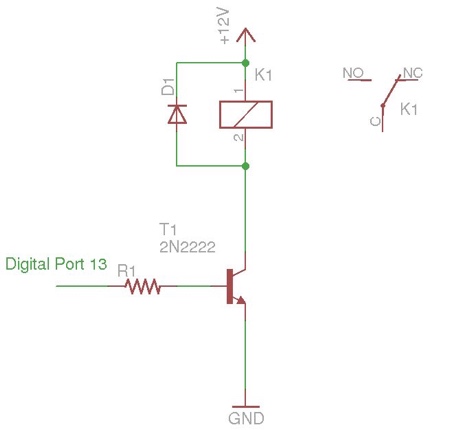
R1 på diagrammet på næste side beregnes som R1 = U/Ib = 5 V/0,0012 A = 4,2 kOhm. Vi bruger en standardmodstand på 3,9 kOhm, hvilket belaster Arduinoen med 1,3 mA, hvilket ikke er noget problem.

Som diode benyttes en 1 A standard ensretter diode, 1N4007.

Der laves en stiktavle med 12 jordforbundne 230 V AC stik på hovedtavlen i 3 rækker à 4 stik.

Stikkene nummereres fortløbende i læseretningen.

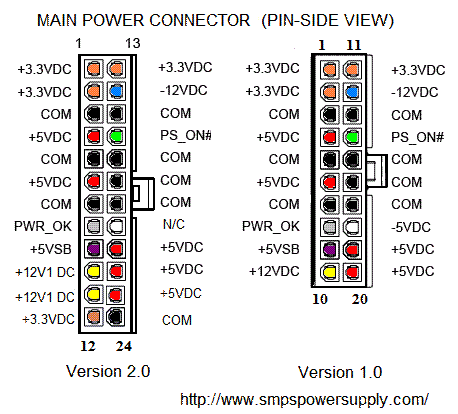
|  |  |  |
| --- | --- | --- |
| **Stik #** | **Anvendelse** | **Input** |
| 1 | Transportbånd | Relæ 1 |
| 2 | Ledig | Relæ 2 |
| 3 | Røgmaskine | Relæ 3 |
| 4 | Varmeblæser | Relæ 4 |
| 5 | Lydanlæg | Relæ 5 |
| 6 | TV-skærm | Relæ 6 |
| 7 | Ventilation | Relæ 7 |
| 8 | Skibsbro | Relæ 8 |
| 9 | Raspberry Pi | NET |
| 10 | ATX strømforsyning | NET |
| 11 | 24 V DC strømforsyning | NET |
| 12 | Ledig | NET |



## Lavspænding

Lavspænding produceres dels af en 24 V DC radiostrømforsyning, og dels af en ATX strømforsyning fra en PC. ATX forsyningen leverer 3,3 V DC (19 A), 5 V DC (14 A) og 12 V DC (7,5 A). Der etableres en fælles 0 (GND). PS\_ON# skal forbindes til COM for at tænde for strømforsyningen. Der laves en fast forbindelse, så strømforsyningen starter når den får netspænding.

ATX strømforsyningen er kablet, som følger:



Strømforsyningerne leverer strømmen ind i hovedtavlen, der distribuerer 24 og 12 V DC til de eksterne enheder. 3,3 V og 5 V anvendes kun internt i hovedtavlen.

De eksterne enheder regulerer selv strømmen ned fra 12 V, til hvad der nu er brug for.

## Servodriver

På hovedtavlen sidder en PCA9685 servodriver med 16 kanaler, som via I2C bussen er forbundet til Raspberry Pi, der styrer servoerne. Manual for driveren findes i bilag 2.

Hver servomotor forsynes med 5 V DC fra ATX strømforsyningen. Ved hver motor placeres en 100 uF elektrolytkondensator for at kompensere for spændingsfaldet ved start af motoren. Dertil kommer en skærmet signalledning til hver servomotor. Kabellængden forventes ikke at udgøre et problem jf. andres erfaringer indhentet på nettet.

### Viserinstrumenter

Der er følgende viserinstrumenter:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Instrument** | **Servo nr.** | **Min. udslag** | **Max. udslag** | **Min. visning** | **Max. visning** |
| Maskintelegraf | 1 |  | | * Astern, Full * Astern, Half * Astern, Slow * Stop * Ahead, Slow * Ahead, Half * Ahead, Full | |
| Røggastemperatur | 2 |  |  | 500C | 6500C |
| Damptryk i kedel | 3 |  |  | 0 bar | 10 bar |
| Vacuum i kondensator | 4 |  |  | 0 bar | 1 bar |

### Omstyring

Omstyringen på dampmaskinemodellen styres af en stor servomotor, som er Servo nr. 5.

## Lydanlæg

Simulatoren er forsynet med et mono lydanlæg placeret under transportbåndet og med en 130 W højttaler i hvert sideskab.

Lydfilerne afspilles af Raspberry Pi´HDMI udgang og leveres til forstærkeren aux indgang via en HDMI splitter placeret på hovedtavlen.

Der er følgende lydfiler til rådighed:

|  |  |  |  |
| --- | --- | --- | --- |
| **Filnavn** | **Lyd** | **Anvendelse** | **Varighed** |
| Dampmaskine.mp3 | Maskinrum på Bjørn | Demo | Ca 20 minutter |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Røganlæg

Der er tikoblet en teaterrøgs generator til manifolden for kedelrørene. Generatoren tændes når indfyringen starter, og røgudviklingen startes når frontlågen på kedlen åbnes.

Det skal besluttes hvor lang tid røgudviklingen skal køre.

## Ventilation

Der etableres udsugning til det fri fra toppen af skorstenen, sådan at den dannede røg kan fjernes fra simulatoren.

Det skal besluttes hvor lang tid der skal gå fra røgudviklingen starter til ventilationen starter, samt hvor lang tid ventilationen skal køre.

Ventilatoren starter ved aktivering af relæ # 7.

Der ventileres XX m3/min.

## Varmluftblæser

Der placeres en varmluftblæser ophængt under kedelrørene med blæseretning mod indfyringsåbningen. Den styres af et selvstændigt 13 A relæ for ikke at overbelaste printet i powernettet. Blæseren startes når indfyringslugen åbnes, hvis der er "ild" i ovnen.

Varmluftsblæseren har en effekt på 2.000 W.

## Flammelys

Der laves en lysflade med 20 RGB powerleds, som placeres på bagvæggen i brandkammeret, så den lyser ud mod indfyringsåbningen.

Power led'ene limes på en 5 mm aluminiumsplade.

LED'erne forbindes så lysstyrke og farve kan reguleres for hele fladen. Slaven genererer selv et varierende lys. Dvs at masteren kun tænder eller slukker for lyset, sådan at masterens regnekraft ikke går til at lave lysshow. Lysdioderne har følgende specifikation:

* **Emitting Color :RGB(Red, Green, Blue)**
* **DC Forward Voltage (VF): Red: 2.0-2.5Vdc, Green: 3.2-3.6Vdc, Blue: 3.2-3.6Vdc,**
* **DC Forward Current (IF): Red: 400MA/ Green: 350MA/ Blue: 350MA**
* **Viewing Angle: 120 Degree**
* **Luminous Intensity: Red: 45-50LM, Green: 65-75LM, Blue: 20-25LM**
* **Wave Length: Red: 620-625nm, Green: 520-530nm, Blue: 460-465nm.**

Med 20 parallelforbundne dioder bliver strømforbruget hhv 8 A på rød og 7 A på blå og grøn.

For at have lidt sikerhedsmargen designer vi lysfladen med 20 dioder. Det giver 8 A på rød og 7 A på blå og grøn. Det giver et effekttab i mosfetten på 24 W, hvilket den kan håndtere.

Det giver dog fortsat behov for en særskilt 5 V stømforsyning, som kan levere de 22 A. Denne strømforsyning kan tændes med samme relæ, som tænder for varmeblæseren. Det kan løses med en ATX 350 W strømforsyning, der kan give 25 A på 5 V.

Farveskanning af fotografier af brændende kul i essen giver følgende typiske farver:

RGB = 248, 209, 166

RGB = 252, 251, 233

RGB = 226, 98, 113

RGB = 239, 145, 93

RGB = 168, 73, 103

RGB = 229, 137, 86

Arduinoen programmeres så den vælger 3 random værdier for farverne og en for varigheden, og så gentager det i det uendelige.

random(R) = 168 til 252

random(G) = 73 til 251

random(B) = 93 til 233

random(tid) = 100 til 5000 millisekunder

Det bør programmeres, så farverne fader over i hinanden i løbet af den randomiserede tidsperiode.

## Overtryksventil

Midt over maskinrummet placeres en vandforstøver, der aktiveres, hvis sikkerhedsventilen lætter.

Det skal besluttes hvor længe den skal blæse.

Vandforstøveren vandforsynes af kommunalt trykvand via en slangebrudsventil, et stykke slange, og en 12 V magnetventil.

## Maskintelegraf

Maskintelegrafen styres på forskellige måder, afhængigt af hvilket program, der afvikles på simulatoren.

Hvis der er bemanding på broen, så styres maskintelegrafen der fra.

Hvis der ikke er bemanding på broen, så anvendes en af to tidskodede tabeller, der angiver hvor lang tid der går mellem hver ordre, og hvilken rækkefølge ordrene kommer i.

For at fastlægge den realistiske tidskode, så gennemspillet et bugseringsforløb på simulatoren. Den pædagogiske tidskode bliver så den samme, blot med reduceret tid mellem hver ordre. Hvor meget tiden skal reduceres må vi prøve os frem til.

## Omstyring

Omstyringshåndtaget har 28 omdrejninger fra fuld kraft bak til fuld kraft frem. Håndtagets stilling aflæses v.h.a. en 10 kOhm skydemodstand, der er mekanisk forbundet til en omløber på spindlen.

Viseren bag ved omstyringshåndtaget styres af en stepmotor bag skottet.

Stepmotoren styres direkte af Arduino slaven, og slaven kommunikerer omstyringshåndtagets stilling til Raspberry Pi masteren via I2C bussen. Der sendes en byte (0-255) til angivelse af omstyringshåndtagets stilling.

Ved initieringen køres viseren først i yderposition, svarende til fuld kraft frem, hvorved der aktiveres en microswitch. Derpå stilles viseren svarende til omstyringshåndtagets stilling.

## Oliepumpe

Simuleringen benytter en håndpumpe, der pumper frostsikret vand fra en beholder gennem et flowmeter og retur til beholderen.

Flowmeteret benytter en hall-sensor til at måle omdrejningerne på skovlhjulet.

Arduinoslaven, der er koblet til flowmeteret returnerer antal liter væske, der er pumpet gennem pumpen siden sidste aflæsning. Det vil sige at registeret nulstilles efter hver aflæsning fra masteren.

## Dampmaskinemodel

Dampmaskinemodellen styres af en Arduino slave anbragt i modellens forlængelse bag ved skottet.

Modellen initieres ved at omstyringen kører ud i de to yderstillinger, hvor den aktiverer to microswtiches, hvorefter omstyringen kører i en midterposition. Derpå kører motoren frem til den har højtryksstemplet i top, hvilket den registrerer ved hjælp af et blændehjul med en IR LED og en IR sensor. Blændehjulet er fastlåst på hovedaksen bag ved skottet.

Fremdriften af modellen leveres af to stepmotorer bag skottet.

Stemplerne belyses fra den side, hvor "dampen" kommer ind.

Modellens omdrejningshastighed bestemmes ved to parametre: den tilførte energi og omstyringens stilling. Disse to værdier modtages fra Raspberry Pi masteren via I2C bussen som 2 bytes.

## Skueglas for olie

Skueglasset er via en peristaltisk pumpe med stepmotor forbundet med et tilsvarende glas på bagsiden af skottet. Pumpen kan pumpe væske mellem de to glas, og derved variere væskestanden i skueglasset.

Den samlede væskemængde passer med at det ene glas viser maksimum når det andet viser minimum.

Skueglasset har en indvendig diameter på 8 mm, hvilket giver et rumfang på ca. 0,5 ml/cm. Pumpen kan levere 160 ml/min svarende til 2,5 cm/sek. Røret kan altså fyldes på ca. 20 sekunder.

For at fastsætte 0-punktet på det synlige skueglas er det skjulte glas forsynet med en flyder, der aktiverer en Micro switch, når 0-punktet er nået i det synlige glas. Denne kalibrering foregår som en del af initieringen af simulatoren.

Pumpen styres af en Arduino slave, der modtager en byte fra Raspberry Pi masteren (0-255), der angiver den ønskede væskehøjde i skueglasset.

## Transportbånd

### Båndet

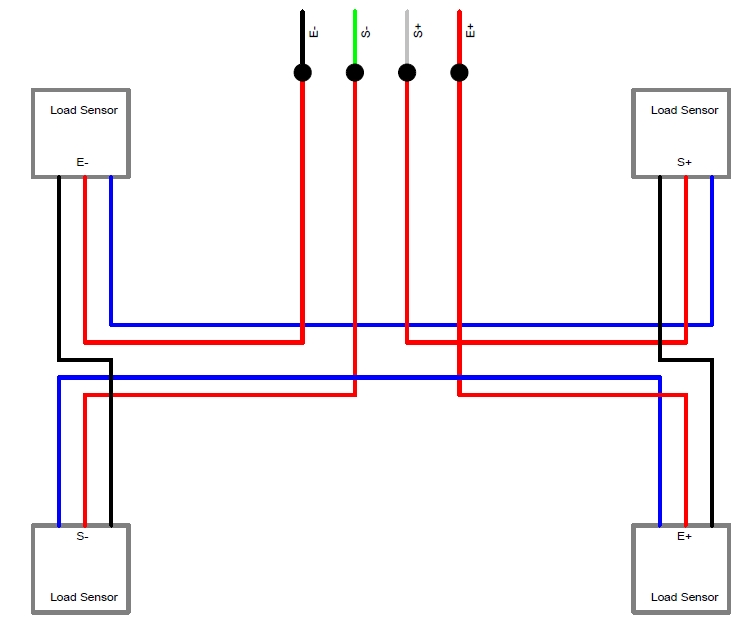
Transportbåndet er leveret af SOCO Systems og har følgende parametre:

* Længde 1500 mm
* Bredde 500 mm
* Understøttelse Ruller
* Egenvægt ca. 40 kg

Motoren er forsynet med temperaturføler. [Temperaturføleren er af typen ... Den aflæses af den arduino, der styrer båndet og stopper båndet i tilfælde af overhedning ...]

### Vejeceller

Båndet står på fire vejeceller, der er koblet i en Wheatstones bro. Signalet her fra behandles i en A/D converter: HX711 fra AVIA Semiconductor. Se bilag 3.



Vejecellerne er koblet som vist ovenfor. De kobles til HX711 som følger:

|  |  |  |
| --- | --- | --- |
|  | På tegning | På HX 711 |
| Forsyning + | E+ | Out + |
| Forsyning - | E- | GND |
| Signal + | S+ | A+ |
| Signal - | S+ | A- |

Vi benytter indgang A, da den kan programmeres til 128 eller 64 gange forstærkning, hvorimod indgang B har en fast forstærkning på 32 gange.

Den anden ende af HX 711 forbindes til 5 V forsyningsspænding og til en Arduino Nano som følger:

|  |  |  |
| --- | --- | --- |
|  | HX 711 | Arduino |
| Forsyning + 5 V DC | VCC | + 5V |
| Jord 0 V | GND | GND |
| Data | DO/RX | A1 |
| Klokkesignal | CK/TX | A0 |

### Frekvensomformer

Motorhastigheden styres af en frekvensomformer, Danfoss VLT 2800.

Frekvensomformeren er opsat med standard fabriksopsætning med følgende ændringer:

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Variabel #** | **Værdi** | **Bemærkninger** |
| Local/remote operation | 002 | 0 | Frekvensomformeren fjernstyres |
| Motor Power [kW] | 102 | 0.12 | Mærkeplade på motor |
| Motor voltage [V] | 103 | 230 | Mærkeplade på motor |
| Motor frequency [Hz] | 104 | 50 | Mærkeplade på motor |
| Motor current [A] | 105 | 0.84 | Mærkeplade på motor |
| Rated motor speed [rpm] | 106 | 1325 | Mærkeplade på motor |
| Digital input | 302 | 7 | Start af motor styret på terminal 18 |
| Digitalt input | 304 | 2 | Coasting stop inverted  (motoren løber selv farten af) |
| Analog input | 308 | 1 | Analog spændingsinput på terminal 53 |

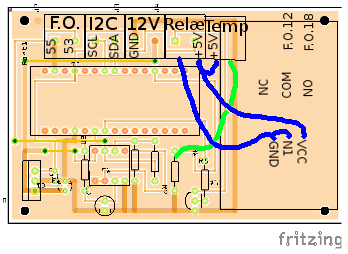
Frekvensomformeren (F.O.) er forsynet med jumper fra terminal 12 (12 V DC) til terminal 27, samt forbundet med µC (Arduino Nano) via relæ til terminal 18 (start/stop) og via spændingsreguleringskreds til terminal 53 (frekvensstyring).

Relæet forbindes sådan, at det kun strømforsynes, når det skal være tændt. Dvs at motoren stopper ved strømsvigt på svagstrømssiden.

Stømforbruget i relækredsen er målt til 80 mA ved en forsyning på 5 V DC.

Som transistor vælges 2N2222, der har en max Ic på 0,6 A, og en hFE (DC Collector/Base Gain hFE Min) på 75, hvilket giver en Ib = Ic/hFE = 0,08/75 = 1,1 mA.

Basismodstanden, R1 beregnes som R1 = U/Ib = 5 V/0,0011 A = 4,7 kOhm.



Via 12 V DC forsyningen får kredsen fælles jord (GRN) med Raspberry Pi.

I2C kabling har sort som SCL og hvid som SDA.

Kredsen er dimensioneret efter en termistor på ca 10 kohm. Hvis termistoren har en anden modstand skal modstanden for enden af den grønne ledning på tegningen skiftes til en med en værdi af samme størrelsesorden som termistoren, da den indgår i en spændingsdeler.

Se i øvrigt bilag 1 side 62.

## Kedelrør

Kedelrørene nummereres fortløbende set fra fyrrumssiden. Rørene nummer 6, 7,11,39 og 42 er forsynet med ultralyds afstandsmålere, så man kan detektere om røret bliver renset med rensekosten. Endvidere detekterer rør 39 om dækslet er åbent.

Forbindelsen til ultralydstransducerne er farvekodet som følger:

|  |  |
| --- | --- |
| **Funktion** | **Farve** |
| Vcc | Rød |
| Trig | Sort |
| Echo | Gul |
| GND | hvid |

## Indfyringslåge

Indfyringslågen skal være lukket når simulatoren initieres. Om lågen er lukket detekteres af en microswitch, der sidder bag et dæksel for oven til venstre for indfyringslågen i forbindelse med lågens hængsel.

Switchen bruges til at nulstille den optiske rotary encoder, der ligeledes sidder på indfyringslågens hængsel.

Microswitch og rotary encoder er forbundet til en Arduino slave. Raspberry Pi masteren aflæser lågens stilling via I2C bussen som en byte (0-255).

## Spjæld for primær forbrændingsluft

Spjældets stilling aflæses v.h.a. en ultralyds transducer koblet til en Arduino slave. Spjældets stilling aflæses af Raspberry Pi masteren via I2C bussen. Der aflæses en byte (0-255).

## Skueglas for kedelvand

Skueglasset er via en peristaltisk pumpe forbundet med et tilsvarende glas på bagsiden af skottet. Pumpen kan pumpe væske mellem de to glas, og derved variere væskestanden i skueglasset.

Den samlede væskemængde passer med at det ene glas viser maximum når det andet viser minimum.

For at fastsætte 0-punktet på det synlige skueglas er det skjulte glas forsynet med en flyder, der aktiverer en microswitch, når 0-punktet er nået i det synlige glas. Denne kalibrering foregår som en del af initieringen af simulatoren.

Pumpen styres af en Arduino slave, der modtager en byte fra Raspberry Pi masteren (0-255), der angiver den ønskede væskehøjde i skueglasset.

## Haner for kedelvand

Der er to haner for regulering af kedelvandet: en påfyldningshane til venstre og en aftapningshane til højre.

Hanerne har 8 omdrejninger fra helt lukket til helt åben. Håndtagets stilling aflæses v.h.a. en 10 kOhm skydemodstand, der er mekanisk forbundet til en omløber på spindlen.

Stillingen af alle 6 haner aflæses af den samme Arduino slave, og gemmes i et register på 6 bytes. Raspberry Pi masteren anmoder om dataene via I2C bussen.

## Haner for kondensatoren

Der er to haner til regulering af havvandet i kondensatoren: en påfyldningshane til venstre og en aftapningshane til højre.

Hanerne har 8 omdrejninger fra helt lukket til helt åben. Håndtagets stilling aflæses v.h.a. en 10 kOhm skydemodstand, der er mekanisk forbundet til en omløber på spindlen.

Stillingen af alle 6 haner aflæses af den samme Arduino slave, og gemmes i et register på 6 bytes. Raspberry Pi masteren anmoder om dataene via I2C bussen.

## TV skærm i koøjet

TV skærmen strømforsynes via relæ nr 6. Når skærmen har forsyningsspænding skal den tændes og der skal vælges visning v.h.a. en IR fjernbetjening. IR signalerne produceres af en Arduino slave med en IR LED, der sender den relevante kode til skærmen.

Opgaven med at vise en videofilm på skærmen kan klares ved anvendelse af fire koder:

1. ON/OFF
2. OK
3. -> (højrepil)
4. Pause/play

Koderne består af et 38 kHz firkantsignal, der er tændt eller slukket i et bestemt tidsmønster ligesom en slags morsekode. En ordre består af 38 pulser og 38 pauser, hvor alle pauserne har samme længde og længden af pulserne varierer mellem korte og lange.

Koden er dechiffreret ved at optage signalet fra den fjernbetjening, der fulgte med fjernsynet.

Arduino slaven styres af Raspberry Pi masteren via I2C bussen. Masteren sender en byte: 1, 2 eller 3, og det får slaven til at sende det tilsvarende signal til fjernsynet.

Der optages en videofilm fra bagbord side af Ærøfærgen på en sejlads fra Ærøskøbing til Svendborg. Den afspilles, hvis der ikke er bemanding på broen.

Hvis der er bemanding på broen, så vises skærmbilledet fra simulatoren på skærmen.

## Damphaner

Den venstre hane regulerer dampmængden til dampmaskinen.

Den højre hane lukker damp ud i det fri.

Hanerne har 8 omdrejninger fra helt lukket til helt åben. Håndtagets stilling aflæses v.h.a. en 10 kOhm skydemodstand, der er mekanisk forbundet til en omløber på spindlen.

Stillingen af alle 6 haner aflæses af den samme Arduino slave, og gemmes i et register på 6 bytes. Raspberry Pi masteren anmoder om dataene via I2C bussen.

# Computer Hardware

Simulatoren drives af en Raspberry Pi 3.0 og et antal Arduino Nano 3.0.

## Raspberry Pi

Valgt computer: Raspberryn Pi 3 Model B V1.2.

User: pi. Pw: AngeloDGV.

### Opsætning af Raspberry Pi 3 computer

Fra Raspberry Pi's website, <http://www.raspberrypi.org/downloads> , downloades det nyeste Raspbian image. Raspbian er operativsystemet til Raspberry Pi. Det er en Linux klon:

***NOOBS***

Offline and network install

Version: **2.1.0**

Release date: **2016-11-29**

*NOOBS INSTALLATION INSTRUCTIONS*

*1. Insert an SD card that is 4GB or greater in size into your computer.*

*2. Format the SD card using the platform-specific instructions below:*

*a. Windows*

*i. Download the SD Association's Formatting Tool from https://www.sdcard.org/downloads/formatter\_4/eula\_windows/*

*ii. Install and run the Formatting Tool on your machine*

*iii. Set "FORMAT SIZE ADJUSTMENT" option to "ON" in the "Options" menu*

*iv. Check that the SD card you inserted matches the one selected by the Tool*

*v. Click the "Format" button*

*b. Mac*

*i. Download the SD Association's Formatting Tool from https://www.sdcard.org/downloads/formatter\_4/eula\_mac/*

*ii. Install and run the Formatting Tool on your machine*

*iii. Select "Overwrite Format"*

*iv. Check that the SD card you inserted matches the one selected by the Tool*

*v. Click the "Format" button*

*c. Linux*

*i. We recommend using gparted (or the command line version parted)*

*ii. Format the entire disk as FAT*

*3. Extract the files contained in this NOOBS zip file.*

*4. Copy the extracted files onto the SD card that you just formatted so that this file is at the root directory of the SD card. Please note that in some cases it may extract the files into a folder, if this is the case then please copy across the files from inside the folder rather than the folder itself.*

*5. Insert the SD card into your Pi and connect the power supply.*

*Your Pi will now boot into NOOBS and should display a list of operating systems that you can choose to install.*

*If your display remains blank, you should select the correct output mode for your display by pressing one of the following number keys on your keyboard:*

*1. HDMI mode - this is the default display mode.*

*2. HDMI safe mode - select this mode if you are using the HDMI connector and cannot see anything on screen when the Pi has booted.*

*3. Composite PAL mode - select either this mode or composite NTSC mode if you are using the composite RCA video connector.*

*4. Composite NTSC mode*

*If you are still having difficulties after following these instructions, then please visit the Raspberry Pi Forums ( http://www.raspberrypi.org/forums/ ) for support.*

Vælg følgende operativsystem til installation:

***Raspbian Jessie with PIXEL***

Image with PIXEL desktop based on Debian Jessie

Version: **November 2016**

Release date: **2016-11-25**

Kernel version: **4.4**

**Når operativsystemet er installeret opsættes landekode, keyboard og mus til danske standarder. Derpå vælger man konfigurationsmenuen til at tillade I2C kommunikation.**

**Herefter går man ind i kommandovinduet for at tilføje de nødvendige moduler til modulfilerne i /etc direktoratet:**

**... $ sudo nano /etc/modules**

**Filen skal om nødvendigt rettes til at have følgende indhold:**

# /etc/modules: kernel modules to load at boot time.

#

# This file contains the names of kernel modules that should be loaded

# at boot time, one per line. Lines beginning with "#" are ignored.

# Parameters can be specified after the module name.

snd-bcm2835

i2c-bcm2708

i2c-dev

**Tjek om modulerne er installeret:**

**...$ sudo apt-get install i2c-tools**

**Før du fortsætter skal du reboote raspberry Pi computeren:**

**...$ sudo shutdown -r now**

**Så er det tid til at teste om Raspberry Pi'en kan udføre I2C kommunikation:**

**...$ sudo i2cdetect -y 1**

**Svaret skulle se nogenlunde sådan ud:**

0 1 2 3 4 5 6 7 8 9 a b c d e f

00: -- -- -- -- -- -- -- -- -- -- -- -- --

10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

70: -- -- -- -- -- -- -- --

**Så er vi kommet til at skulle installere I2C understøttelse for Python 3:**

...$ sudo apt-get install python-smbus

...$ sudo apt-get install libi2c-dev

...$ sudo apt-get install python3-dev

...$ cd /usr/src

...$ sudo wget http://ftp.de.debian.org/debian/pool/main/i/i2c-tools/i2c-tools\_3.1.0.orig.tar.bz2

...$ sudo tar xvf i2c-tools\_3.1.0.orig.tar.bz2

...$ cd i2c-tools-3.1.0/py-smbus

...$ sudo mv smbusmodule.c smbusmodule.c.old

...$ sudo wget https://gist.githubusercontent.com/sebastianludwig/c648a9e06c0dc2264fbd/raw/2b74f9e72bbdffe298ce02214be8ea1c20aa290f/smbusmodule.c

**Hvis det hele virkede som det skulle, så er følgende ting sket:**

1. **installering af libi2c-development biblioteker**
2. **installering af python3-dev biblioteker**
3. **download af den aktuelle version af i2c-tools, for tiden er det version 3.1.0**
4. **udpakning og verificering af indholdet af i2c-tools arkivet**
5. **skift til kilde kode direktoratet for py-smbus**
6. **omdøbe smbusmodule.c kildekode fil til noget andet**
7. **hente den nyeste version af den kode der virker med Python version 3-**

**Hurra for Sebastian Ludwig, der har skrevet kildekoden :-)**

**Nu har vi fået fat i kildekoden, så skal den blot kompileres og lægges det rigtige sted, så Python 3 kan få fat i den:**

...$ cd /usr/src/i2c-tools-3.1.0/py-smbus

...$ sudo python3 setup.py build

**Det skulle gerne give et svar a´la :**

running build

running build\_ext

building 'smbus' extension

gcc -pthread -DNDEBUG -g -fwrapv -O2 -Wall -Wstrict-prototypes -D\_FORTIFY\_SOURCE=2 -g -fstack-protector --param=ssp-buffer-size=4 -Wformat -Werror=format-security -fPIC -I/usr/include/python3.2mu -c smbusmodule.c -o build/temp.linux-armv6l-3.2/smbusmodule.o

creating build/lib.linux-armv6l-3.2

gcc -pthread -shared -Wl,-O1 -Wl,-Bsymbolic-functions -Wl,-z,relro build/temp.linux-armv6l-3.2/smbusmodule.o -o build/lib.linux-armv6l-3.2/smbus.cpython-32mu.so

**Og så skal den installeres:**

...$ sudo python3 setup.py install

running install

running build

running build\_ext

running install\_lib

copying build/lib.linux-armv6l-3.2/smbus.cpython-32mu.so -> /usr/local/lib/python3.2/dist-packages

running install\_egg\_info

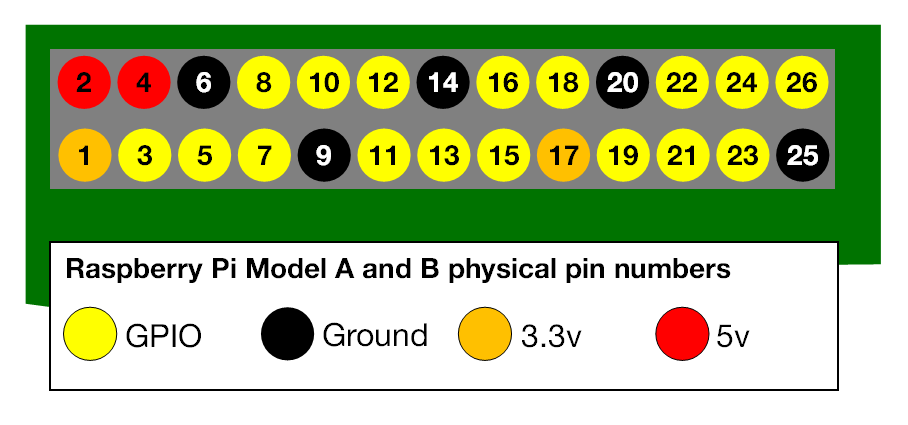
Writing /usr/local/lib/python3.2/dist-packages/smbus-1.1.egg-info

**Så er Python 3 klar til at kunne kommunikere med I2C modulerne.**

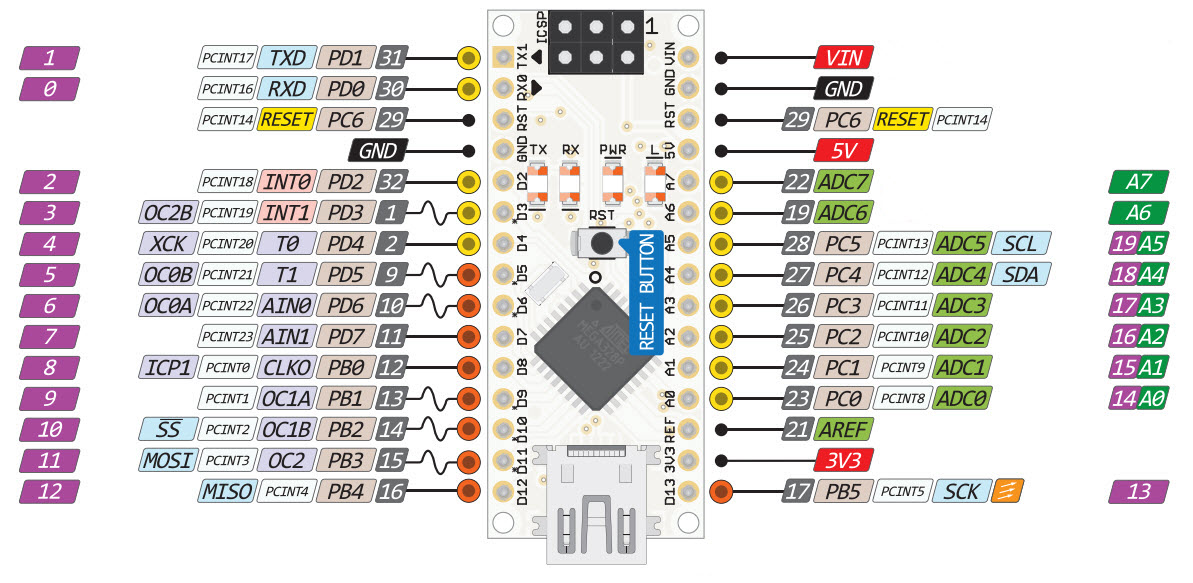
**Tak til** [**http://jtecheng.com/?p=959#comment-925**](http://jtecheng.com/?p=959#comment-925)

### **I2C-bus**

**På Rasberry Pi's GIOP (General Input Output Pins) findes pin 3, 5 og 6:**



På Arduino Nano findes pin A4, A5 og GND:



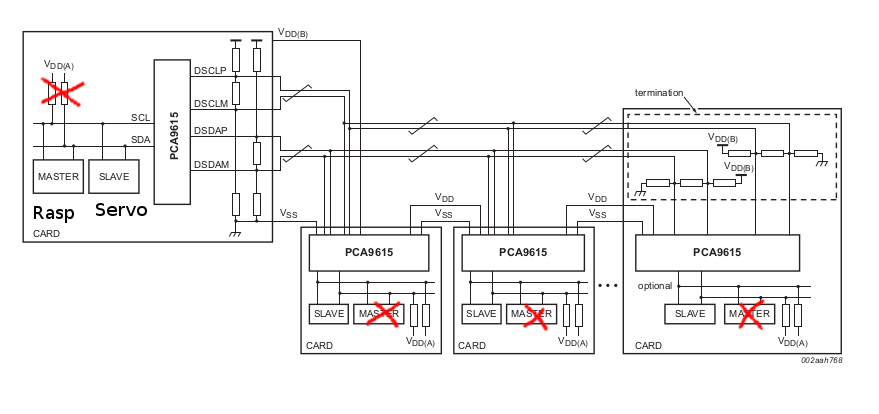
**og de forbindes som følger:**

|  |  |  |
| --- | --- | --- |
|  | Raspberry Pi 3 | Arduino Nano |
| I2C SDA | 3 | A4 |
| I2C SCL | 5 | A5 |
| GND | 6 | GND |

Da I2C bussen ikke er designet til at fungere over lange afstande (over nogle få cm) er vi nødt til at lave et kunstgreb: I2C signalet oversættes til et differentielt signal, DI2C, der transmitteres i parsnoede ledninger, hvorved det bliver væsentligt mindre følsomt overfor ekstern elektromagnetisk støj, som der er meget af i et maskinværksted som på Det Gamle Værft.

Der benyttes en PCA9615DPJ kreds. Se PCA9615 datablad.

Kablingen er som følger:



Der anvendes ikke pull-up modstande før den første DI2C-kreds, da de er indbygget i Rasberry Pi.

Der kables mellem RaspberryPi og Arduinoslaver med cat5e spundet kabel. Farvekoderne fremgår af nedenstående tabel:

|  |  |
| --- | --- |
| **Funktion** | **Wirefarve** |
| D-SCL-P | Orange |
| D-SCL-M | Orange/hvid |
| D-SDA-P | Grøn |
| D-SDA-M | Grøn/hvid |
| VDD(B) = +5V DC | Brun + Brun/hvid |
| VSS = GND | Blå + Blå/hvid |

Slaverne kobles på én bus i følgende rækkefølge:

|  |  |  |
| --- | --- | --- |
| **Busplacering** | **Funktion** | **Slave nummer** |
| **Hovedtavle** | | |
| 0 | Hovedcomputer + servostyring | Master + 0x40 + 0x70 |
| 0 | Styring af 240 V AC forsyningsspænding mv. | 0x10 |
| **Bag styrbord skot** | | |
| 1 | Dampmaskinemodel | 0x52 |
| 2 | Omstyring | 0x50 |
| 3 | Oliepumpe | 0x51 |
| **Bag ovnskot** | | |
| 4 | Vejeceller | 0x31 |
| 5 | Skueglas for smøreolie | 0x38 |
| 6 | Indfyringslåge | 0x33 |
| 7 | Luftspjæld | 0x34 |
| 8 | Røggasrør | 0x35 |
| 9 | Transportbånd | 0x32 |
| 10 | Skueglas for kedelvand | 0x37 |
| 11 | Dampventiler, Søvandsventiler og Kedelvandsventiler | 0x36 |
| 12 | Overtryksventil | 0x39 |
| **Bag bagbord skot** | | |
| 13 | TVskærm fjernbetjening | 0x42 |
| 14 | Valg af simulatorfunktion | 0x41 |
| **Kommandobro** | | |
| 15 | Maskintelegraf afsender | 0x20 |

I Python programmeringen tages der højde for at der kan opstå I/O-fejl hvis bussen er optaget eller hvis den Arduino, som man henvender sig til er ved at lave noget andet, der sætter interruptet ud af drift. Kommer der en I/O-fejl skal programmet forsøge et par gange mere, før det resulterer i et stop.

## Arduino Nano

De enkelte Arduino mikrocomputere er programmeret i Arduino-C. Som indføring i Arduino-C kan anbefales: Beginning C for Arduino af Jack Purdum (ISBN-13: 978-1-4842-0941-7).

# Algoritme

I de følgende afsnit beskrives de algoritmer, der benyttes til at simulere dampmaskinens drift.

## Designparametre

Her beskrives de designparametre, der er anvendt i hhv den realistiske udgave af algoritmen og den pædagogiske udgave.

De realistiske designparametre er som vi forventer at de vil blive om bord på Angelo, når den er forsynet med en 150 hk compound dampmaskine.

De pædagogiske designparametre er tilpasset, sådan at brugeren af simulatoren oplever en hurtigere reaktion på sine handlinger. Det gøres fx ved at reducere vandmængden i kedlen.

## Driftsparametre

Her beskrives de variable, som skyldes maskinens slitage: Olieforbrug, damptab mv.

De realistiske driftsparametre er som vi forventer at de vil blive om bord på Angelo, når den er forsynet med en 150 hk compound dampmaskine.

De pædagogiske driftsparametre er tilpasset, sådan at brugeren af simulatoren oplever en hurtigere reaktion på sine handlinger. Det gøres fx ved at øge mængden af flyveaske der dannes pr time, øge behovet for smøring, og større tab af damp grundet utætheder etc.

## Lovmæssigheder

Her beskrives de temodynamiske sammenhænge mellem indfyret mængde kul og den dannede mængde damp.

# Computer Software

## Raspberry Pi

Styresystemet i Raspberry Pi er Raspbian Linux:

***Raspbian Jessie with PIXEL***

Image with PIXEL desktop based on Debian Jessie

Version: **November 2016**

Release date: **2016-11-25**

Kernel version: **4.4**

En generel indføring i Linux kan findes i: Beginning Linux Programming af Neil Matthew og Richard Stones. (ISBN 978-0-470-14762-7).

Raspberry Pi afvikler et programkompleks skrevet i Python 3. Version 3.5. Som indføring i Python 3 kan anbefales: The Coder's Apprentice af Pieter Spronck. Bogen kan downloades fra <http://www.spronck.net/pythonbook>.

Alle Python programkoderne, der er skrevet til simulatoren, ligger i /home/pi/Python\_sketches.

Der er 4 hovedprogrammer:

* **simulatorA.py** (Maskinsimulator med realistisk respons)
* **simulatorB.py** (Maskinsimulator + skibssimulator (styrhus) med realistisk respons)
* **simulatorC.py** (Maskinsimulator + skibssimulator (styrhus) med pædagogisk respons)
* **simulatorD.py** (Maskinsimulator med pædagogisk respons)

Computerens valg af hovedprogram afhænger af operatørens valg på knivafbryderen i maskinrummet.

Programmerne er modulært opbygget med et modul til betjeningen af hver Arduino i simulatoren.

### Powernet.py

Modul, der styrer den Arduino, der styrer relæerne, der tænder og slukker for 230 V AC til simulatorens forskellige dele.

### Model.py

Modul, der styrer dampmaskinemodellen.

### PCA9685.py

Modul, der muliggør kommunikation mellem Raspberry Pi og servocontrolleren PCA9685 via I2C bussen.

### Servo.py

Modul, der styrer instrumentvisningen på maskintelegraf, røggastermometer, kandensator vakuummeter og kedel manometer. Endvidere styrer den omstyringen på dampmaskinemodellen.

### Transport.py

Modul, der styrer transportbåndet, via en Arduino Nano, der styrer frekvensomformeren.

### Vejecelle.py

Modul, der styrer den Arduino, der aflæser og resetter vejecellerne under transportbåndet.

## Arduino

### Powernet.ino

1. Arduino I2C slave til styring af strømforsyningstavlen. Der modtages et byte med tænd eller sluksignal for et enkelt relæ.

### Model\_computer.ino

Arduini I2C slave til styring af model-dampmaskinen. Der modtages 2 bytes:

1. byte er et tal mellem 0 og 255, der angiver hvor meget energi motoren tilføres

2. byte er et tal mellem 0 og 255, der angiver omstyringens stilling: 0 er fuld kraft bak.

### Slave\_transport.ino

Arduino I2C slave til transportbånd styring. Styring af frekvensomformer vha af 0-10 V spænding

produceret vha PWM på Arduinoens ben D3. Kredsen måler også motortemperaturen vhja A0 og slukker for motoren, hvis temperaturen bliver for høj. Endvidere styres et relæ, der tænder og slukker for motoren.

Der modtages en byte med en værdi mellem 0 og 255 til styring af frekvensen (motorhastigheden),

og en byte: 0 eller 1 til styring af sluk/tænd relæet.

### Vejeceller.ino

Vejecellerne aflæses af en A/D converter, HX 711, som er forbundet med en Arduino Nano.

Programmet benytter biblioteket HX711, der er skrevet af "bogde": <https://github.com/bogde/HX711> .

# Programkode

Al programkode kan findes i <https://github.com/joergenfriis/Steam-Engine-Simulator>

# Kredsløbsdiagrammer

Alle kredsløbsdiagrammer kan findes i <https://github.com/joergenfriis/Steam-Engine-Simulator> .

# Bilag

1. Danfoss VLT 2800 Manual
2. Servo Driver manual
3. HX 711 datablad
4. PCA9615 datablad