

# Indholdsfortegnelse

---

<b>Kapitel 1</b>	<b>Ordliste</b>	<b>3</b>
<b>Kapitel 2</b>	<b>Kravspecifikation</b>	<b>5</b>
2.1	Indledning . . . . .	5
2.2	Aktører . . . . .	7
2.3	Usecases . . . . .	8
2.3.1	UC1: Tilføj/fjern enhed . . . . .	9
2.3.2	UC2: Konfig . . . . .	10
2.3.3	UC3: Aktiver/deaktiver . . . . .	11
2.3.4	UC4: Databehandling . . . . .	11
2.3.5	UC5: Tjek status . . . . .	12
2.3.6	UC6: Udskriv log . . . . .	12
2.4	Ikke-funktionelle krav . . . . .	13
2.5	Grafiske brugerflade-skitser (BS) . . . . .	14
<b>Kapitel 3</b>	<b>Forundersøgelse</b>	<b>19</b>
3.1	Sprinkler . . . . .	19
3.1.1	Sprinkler valg . . . . .	19
3.2	Temperatursensor (SK) . . . . .	20
3.3	Bevægelsessensor (SK) . . . . .	21
3.3.1	Bevægelsessensor valg . . . . .	21
3.4	Fugtsensor (SK og LB) . . . . .	22
3.4.1	Fugtsensor valg . . . . .	23
<b>Kapitel 4</b>	<b>Systemarkitektur</b>	<b>25</b>
4.1	Domænemodel (JC) . . . . .	25
4.2	Hardware beskrivelse . . . . .	26
4.2.1	BDD (SK) . . . . .	26
4.2.2	BDD Master (MK) . . . . .	26
4.2.3	BDD Enhed (MK) . . . . .	27
4.2.4	IBD (SK) . . . . .	27
4.2.5	Grænseflade (HW) . . . . .	29
4.2.6	Signal beskrivelse (HW) . . . . .	30
4.3	Software beskrivelse . . . . .	31
4.3.1	Logical View (JC) . . . . .	31
4.3.2	Deployment View (JC) . . . . .	33
4.3.3	Implementation View (BS) . . . . .	34
4.3.4	Data View (BS) . . . . .	35
4.3.5	Fejl-håndtering (JC) . . . . .	36
4.3.6	Kommunikationsprotokol (BS) . . . . .	36
4.3.7	Kommunikationsprotokol (MIPO) . . . . .	38

<b>Kapitel 5 Hardwaredesign</b>	<b>41</b>
5.1 Prototypemodel . . . . .	41
5.2 Temperatur- og fugtsensor . . . . .	41
5.2.1 Foranliggende filter . . . . .	41
5.2.2 Præcisions- og støjhåndtering . . . . .	43
5.3 PIR-sensor . . . . .	44
5.4 Sprinkler-pumpe system . . . . .	44
5.4.1 230V/5V relæ . . . . .	44
5.4.2 Alpha 2 pumpen . . . . .	45
5.4.3 Relæ styring . . . . .	45
5.4.4 Driver . . . . .	46
5.5 Strømforsyning og tilslutnings-print . . . . .	47
5.5.1 Strømforsyning . . . . .	47
5.5.2 Tilslutningsprint . . . . .	49
5.6 SPI . . . . .	50
5.6.1 Driver . . . . .	51
5.7 Hardware forbindelser . . . . .	52
5.7.1 Enhed (HW) . . . . .	52
5.7.2 Master (HW) . . . . .	53
<b>Kapitel 6 Softwaredesign</b>	<b>55</b>
6.1 Dataprotokol (BS) . . . . .	55
6.2 Applikationsmodeller (JC BS) . . . . .	56
6.2.1 Master . . . . .	56
6.2.2 Enhed . . . . .	61
6.3 Klassebeskrivelser . . . . .	61
6.3.1 Master . . . . .	61
<b>Kapitel 7 Modultest</b>	<b>69</b>
<b>Kapitel 8 Accepttestspezifikation</b>	<b>71</b>
<b>Kapitel 9 Bilag (CD-indhold)</b>	<b>79</b>

# Ordliste 1

---

**AASH** Antal af samtidige hændelser

**Bunker** En fordybning på et golfhul fyldt med fint sand

**Devkit8000** Udviklingsboard fra Embest baseret på Linux (Beagleboard)

**Enhed** Autonomt undersystem som håndterer sensorer mv. baseret på PSoC-boardet

**Enhedstimer** En timer i Enhed som er defineret i ikke-funktionelle-krav

**ETX** End og TeXt ifm. seriel SPI kommunikation

**FT-sensor** Fugt og Temperatur sensor

**Golfbane** En golfbane består af 18 golfhuller

**Golfhul** Et golfhul består af teested, fairway, rough, green og hazards

**GUI** Graphical User Interface (Grafisk brugergrænseflade)

**IDLE tid** Up-time, oppetid eller standby tid

**KP** Komponentpakke. Del af systemet som indeholder en temperatur- og fugtsensor samt en sprinkler

**Master** Hovedenhed i systemet baseret på Devkit8000

**MTBF** Mean Time Between Failure

**PIR-sensor** Passive Infrared sensor

**PSoC** Udviklingsboard fra Cypress

**STX** Start of TeXt ifm. seriel SPI kommunikation

**Teested** Tee eller Teested er starten af et golfhul, hvor man slår sit første slag

**UI** User Interface (brugergrænseflade)

**Vandingsstatus** Indikerer hvorvidt vandingssprinkler er aktiv eller ej (1/0)



# Kravspecifikation 2

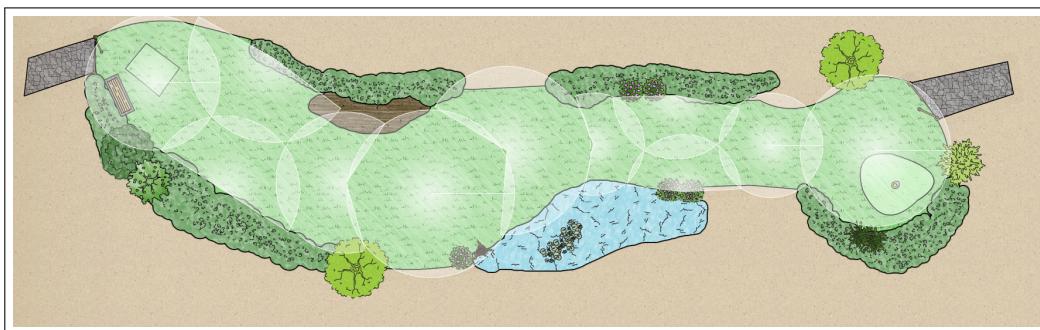
---

## 2.1 Indledning

Gruppen vil udvikle et system til vanding af golfbaner. I forbindelse med stigende temperaturer bliver det mere kritisk, at vandingen af golfbaner sker på de rigtige tidspunkter, for at holde banen spilbar. For at spare på ressourcerne er det også kritisk, ikke at spilde store mængder vand, på vanding af områder som ikke trænger til det.

Med et system af intelligente enheder der arbejder autonomt, men som modtager indstillinger for vandingen fra en masterenhed styres vandingen på hele Golfbanen centralt. På denne måde sparar man arbejdstid for greenkeeperen og vandingen sker kun når det er nødvendigt. Systemet kaldes for EasyWater8000.

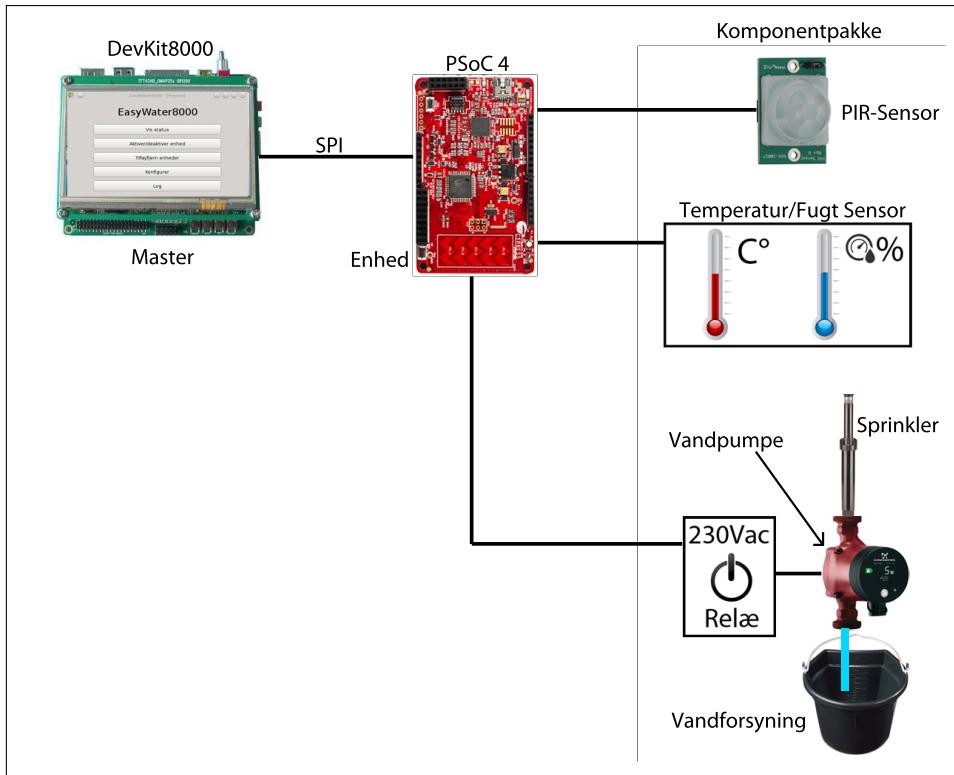
Normalt vil man placere en af disse enheder ved hvert hul på golfbanen og lave et netværk af sensorer lokalt til denne enhed. Enheden kan herved overvåge området og vande hvis nødvendigt. Alle enhederne forbinder til et netværk, som er koblet sammen med en Master. Grænseværdierne, for f.eks. jordfugtigheden, der afgør hvornår enheden skal påbegynde vanding kommer fra Master og styres igennem denne af greenkeeperen.



*Figur 2.1.* Overblik af golfhul med system

Figur 2.1 illustrerer et hul på en given golfbane. Dette hul er inddelt i 11 områder, med hver sin sprinkler. Spinklere er markeret ved halv til helcirkler på figuren. Til venstre på figuren ses tee-stedet med tilhørende indgang. Ved denne indgang er der placeret en bevægelsessensor, som sørger for at der ikke kan vandes på banen, når der er spillere.

Oversigt over blokkene i systemet kan ses i figur 2.2. De tre hovedbegreber for systemet er Master, som er hovedenheden baseret på Devkit8000. Enhed som er de autonome undersystemer på hvert hul, hvor hovedenheden er PSoC. Og Komponentpakker som er et begreb for pakken med temperatur- og fugtsensor samt sprinkler. Disse er distribueret ud på selve golfhullet.



**Figur 2.2.** Systemoverblik

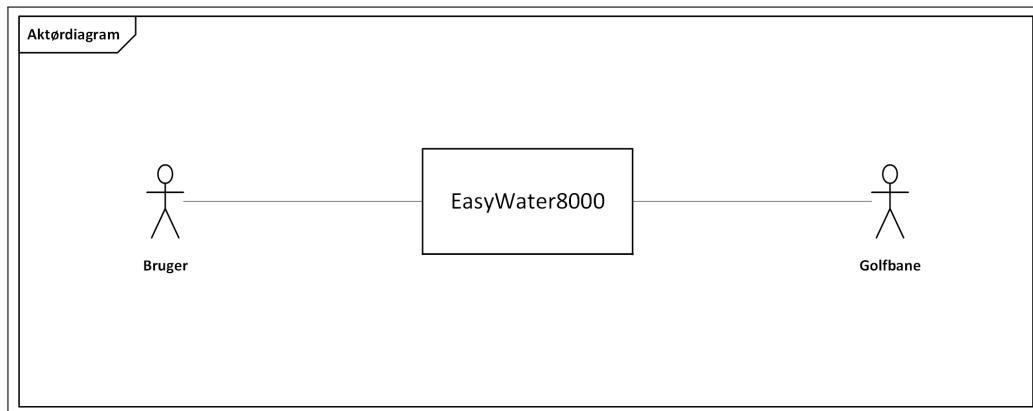
En Enhed består altså af:

1. Fugtighedssensor(er)
2. Temperatursensor(er)
3. Bevægelsessensor(er)
4. Sprinkler(e)
5. PSoC controller board

Fugt- og temperatursensorerne registrerer banens behov for vanding. Bevægelsessensoren registrerer om der er spillere på det pågældende hul. Sprinkleren sørger for vandingen når der skal vandes. Denne skjules nede i græsset og kommer op når vandingen skal være aktiv. PSoC controller-boardet bliver hjernen i Enheden. Denne styrer kommunikationen til Master og holder styr på de generelle funktioner for Enheden så som udlæsning af data, aktivering af sprinkler mv.

## 2.2 Aktører

Her beskrives systemets aktører. Disse vil blive refereret til, i de efterfølgende usecase-beskrivelser.

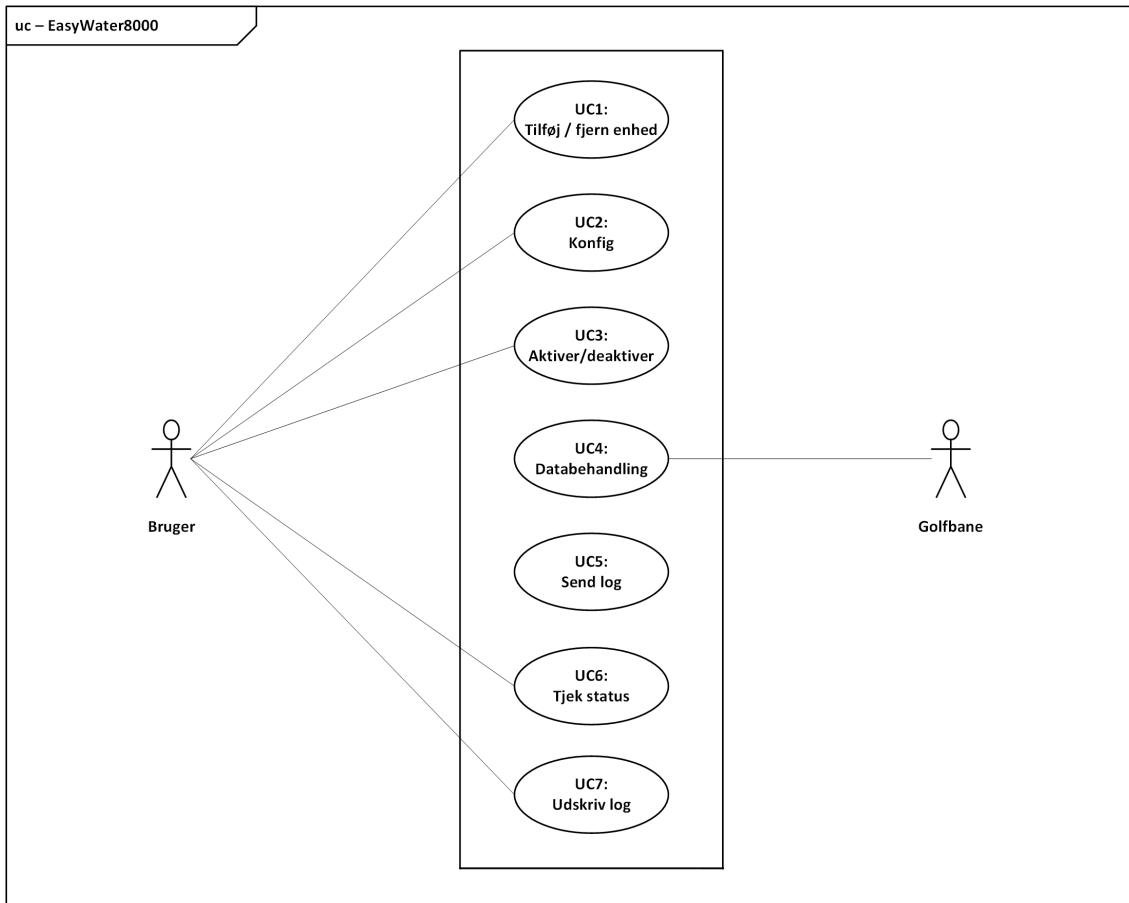


**Figur 2.3.** Kontekst diagram

Aktør navn	Beskrivelse
Bruger	Bruger vil normalt være greenkeeperen. Det er vedkommende som kontrollerer og betjener systemet. (Primær)
Golfbane	De almene omgivelser på golfbanen, som har indflydelse på systemets sensorer. Det indebærer temperatur, fugtighed og bevægelse i området omkring systemet. (Sekundær)

## 2.3 Usecases

Her følger en dybere beskrivelse af systemets opbygning og måde at virke på. Dette gøres med fulde usecase-beskrivelser hvor systemets virkning er beskrevet i detaljer.



*Figur 2.4.* Usecase diagram

### 2.3.1 UC1: Tilføj/fjern enhed

<b>UC1: Tilføj/fjern enhed</b>	
<b>Mål</b>	Bruger kan tilføje eller fjerne enheder fra systemet
<b>Initialisering</b>	Bruger
<b>Aktører og Stakeholders</b>	Bruger(Primær)
<b>Referencer</b>	Ingen
<b>AASH</b>	1
<b>Efterfølgende tilstand</b>	Ønsket Enhed er tilføjet eller fjernet fra systemet.
<b>Hovedforløb</b>	<ol style="list-style-type: none"> <li>1. Bruger vælger "Tilføj/fjern enhed" i hovedmenu</li> <li>2. Bruger vælger "Tilføj enhed"</li> <li>3. En liste af opsatte enheder præsenteres på skærmen             <ol style="list-style-type: none"> <li>a) Master beder bruger om at indtaste informationer</li> <li>b) Bruger indtaster hul-nr. og adresse på Enhed</li> </ol> <p><b>[Undtagelse 3b.a]</b> Indtastede værdier er ikke gyldige</p> </li> <li>c) Master tilføjer Enhed til systemet med default parametre</li> <li>d) Enhed forbindes til kommunikationsnetværket</li> <li>e) Master verificerer forbindelsen til Enhed og sender dato og tidspunkt</li> <p><b>[Undtagelse 3e.a]</b> Enheden kan ikke verificeres</p> <li>f) Enhed gemmer dato og tidspunkt</li> </ol> <ol style="list-style-type: none"> <li>4. Bruger vælger "Fjern enhed"</li> <li>5. En liste af opsatte enheder præsenteres på skærmen             <ol style="list-style-type: none"> <li>a) Bruger indtaster adresse på Enhed</li> <li>b) Master deaktivere Enhed</li> <li>c) Master sletter Enhed fra systemet</li> </ol> </li> <li>6. Master opdaterer liste med opsatte enheder</li> <li>7. Bruger kan returnere til hovedmenu eller opsætte ny enhed (Gå til UC1.2)</li> </ol>

...fortsat fra forrige side

<b>UC1: Tilføj/fjern enhed</b>	
<b>Undtagelser</b>	<p>3a.a Master viser fejlbesked omkring ugyldige værdier Gå til UC1.3a</p> <p>3e.a Master viser fejlbesked angående verificering af enheden</p> <p>3e.a Bruger kan forsøge igen (Gå til UC1.3e eller afbryde)</p>

### 2.3.2 UC2: Konfig

<b>UC2: Konfig</b>	
<b>Mål</b>	Ændre default parametre på en Enhed
<b>Initialisering</b>	Bruger
<b>Aktører og Stakeholders</b>	Bruger(Primær)
<b>Referencer</b>	Ingen
<b>AASH</b>	1
<b>Efterfølgende tilstand</b>	Ingen
<b>Hovedforløb</b>	<ol style="list-style-type: none"> <li>1. Bruger vælger ”Konfig” i hovedmenu</li> <li>2. Bruger vælger den Enhed, der ønskes omkonfigureret.</li> </ol> <p><b>[Undtagelse 2.a]</b> Bruger vælger afbryd</p> <ol style="list-style-type: none"> <li>3. Bruger ændrer parametre på den valgte Enhed.</li> </ol> <p><b>[Undtagelse 3.a]</b> Der indtastes ugyldige værdier</p> <ol style="list-style-type: none"> <li>4. Bruger vælger ”Gem”</li> </ol> <p><b>[Undtagelse 4.a]</b> Bruger vælger afbryd</p> <ol style="list-style-type: none"> <li>5. Master sender de nye indstillinger til den valgte Enhed.</li> </ol>
<b>Undtagelser</b>	<p>2.a Bruger vælger ”afbryd” Skærm viser hovedmenu</p> <p>3.a Der indtastes ugyldige værdier Skærm viser fejlbesked</p> <p>4.a Bruger vælger ”afbryd” Skærm viser Konfig-menu</p>

### 2.3.3 UC3: Aktiver/deaktiver

<b>UC3: Aktiver/deaktiver</b>	
<b>Mål</b>	At aktivere / deaktivere Enhed
<b>Initialisering</b>	Bruger
<b>Aktører og Stakeholders</b>	Bruger(Primær)
<b>Referencer</b>	Ingen
<b>AASH</b>	1
<b>Efterfølgende tilstand</b>	Enhedstilstand ændres aktiv/deaktiv
<b>Hovedforløb</b>	<ol style="list-style-type: none"> <li>1. Bruger vælger ”Aktiver/Deaktiver” i hovedmenu</li> <li>2. Bruger markerer Enhed ud fra liste af opsatte Enheder</li> <li>3. Bruger vælger ”Aktiver” eller ”Deaktiver” for at ændre Enheden [Undtagelse 3.a] Bruger vælger ”Afbryd”</li> <li>4. Master udskriver på skærmen, at ønsket Enhed er aktiveret eller deaktiveret</li> <li>5. Bruger vælger ”Afslut”</li> <li>6. Master viser hovedmenu</li> </ol>
<b>Undtagelser</b>	<p>3.a Bruger vælger ”Afbryd” Master viser hovedmenu</p>

### 2.3.4 UC4: Databehandling

<b>UC4: Databehandling</b>	
<b>Mål</b>	Indsamle data fra Golfbane og vande
<b>Initialisering</b>	System
<b>Aktører og Stakeholders</b>	Golfbane(Primær)
<b>Referencer</b>	Ingen
<b>AASH</b>	1 per Enhed
<b>Efterfølgende tilstand</b>	Data gemt i Enhed og logget i Master + evt. vanding

...fortsat fra forrige side

<b>UC4: Databehandling</b>	
<b>Hovedforløb</b>	<p>I et fast interval udføres nedenstående forløb på Enhed for hver tilkoblede Komponentpakke:</p> <ol style="list-style-type: none"> <li>1. Sensorernes data for temperatur og fugtighed udlæses fra Golfbanen</li> <li>2. Vanding startes på områder med for lave værdier [Undtagelse 2.a] Bevægelse registreret</li> <li>3. Enhed gemmer data i en buffer til senere udlæsning fra Master</li> </ol> <p>I et andet fast interval udføres nedenstående forløb på Master for hver opsatte Enhed:</p> <ol style="list-style-type: none"> <li>4. Master henter information fra bufferen i Enheden</li> <li>5. Master gemmer data i log</li> </ol> <p>Bevægelsessensoren aktiverer følgende forløb:</p> <ol style="list-style-type: none"> <li>6. Enhed modtager besked ved bevægelse</li> <li>7. Vanding deaktiveres 30 min</li> <li>8. Enhed gemmer hændelse i en buffer til senere udlæsning fra Master</li> </ol>
<b>Undtagelser</b>	2.a. Se punkt 6 og punkt 7 i hovedforløbet

### 2.3.5 UC5: Tjek status

<b>UC5: Tjek status</b>	
<b>Mål</b>	At se status på systemet
<b>Initialisering</b>	Bruger
<b>Aktører og Stakeholders</b>	Bruger(Primær)
<b>Referencer</b>	Ingen
<b>AASH</b>	1
<b>Efterfølgende tilstand</b>	Hovedmenuen vises på Master
<b>Hovedforløb</b>	<ol style="list-style-type: none"> <li>1. Bruger vælger ”Tjek status”</li> <li>2. Aktuel status vises på Master ud fra sidste logføring fra UC4</li> <li>3. Bruger vælger ”Tilbage”</li> </ol>

### 2.3.6 UC6: Udskriv log

<b>UC6: Udskriv log</b>	
<b>Mål</b>	Log udskrives på Masters skærm

...fortsat fra forrige side

<b>UC6: Udskriv log</b>	
<b>Initialisering</b>	Bruger
<b>Aktører og Stakeholders</b>	Bruger(Primær)
<b>Referencer</b>	Ingen
<b>AASH</b>	1
<b>Forudsætning</b>	Aktiv Master
<b>Efterfølgende tilstand</b>	Hovedmenuen vises på Master
<b>Hovedforløb</b>	<ol style="list-style-type: none"> <li>1. Bruger vælger ”Udskriv log” i hovedmenuen</li> <li>2. Log udskrives på Master</li> <li>3. Bruger vælger ”Tilbage”</li> </ol>

## 2.4 Ikke-funktionelle krav

### Brugbarhed

1. Opsætningen skal ske af autoriseret personale
2. UI skal kunne benyttes af bruger efter gennemlæst manual

### Pålidelighed

3. Mean Time Between Failure (MTBF): 2 år
4. Software IDLE tid: minimum 1 måned uden restart

### Ydeevne

5. Master skal kunne håndtere op til 18 enheder
6. Sprinkler skal kunne vande et areal af 360 grader med radius på 3.5 m
7. Der skal kunne tilføjes yderligere Enheder efter opsætning af systemet
8. Master skal hente data fra tilkoblede Enheder hvert 15. minut
9. Enhed skal hente data fra sensorer konstant

### Vedligeholdelse

10. Enheder skal være udskiftelige uden at det er nødvendigt at tilgå sensorer eller sprinkler
11. Sprinklere skal være let tilgængelige for personale

### Enheder

12. Enhed skal kunne operere autonomt efter denne er sat op fra Master
13. Enheder skal gemme log-information ifm. vanding og målinger, indtil Master udlæser denne

## Begrænsninger

- Der udarbejdes en skaleret prototype, da vi ikke har adgang til en hel golfbane
- Grundet tidsbegrænsninger udvikles der ikke et fuldt system
- Som minimum skal der produceres funktionel Master og én komplet Enhed
- Systemet udvikles således, at der til hver sprinkler tilhører en pumpe. Denne løsning vil ikke vælges i en produktionsudgave, her vil der være én vandforsyning, med et tryk stort nok, til at tilføre samtlige sprinklere den nødvendige mængde vand.

## 2.5 Grafiske brugerflade-skitser (BS)

Inden det endelige GUI udvikles laves nogle skitser som udviklingen læner sig op af. Ud fra UC beskrivelserne findes de skærmbilleder som skal vises på Master og skitseres.

Her følger korte beskrivelser af hver skitse samt skitsen selv.

### Startmenu

Figur 2.5 er den første menu brugeren kommer til. Her er valgmulighederne præsenteret for brugeren.



*Figur 2.5.* Skitse af startmenu på GUI

### Vis Status

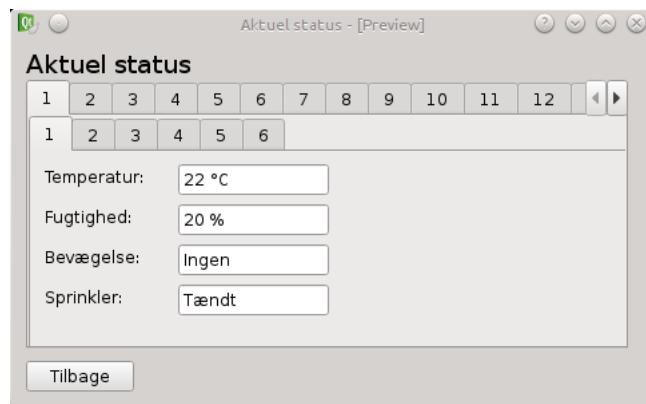
På figur 2.6 vises den aktuelle status af systemets enheder. Den øverste række tal er tilkoblede Enheder. Den anden række tal er komponentpakker tilkoblet den enkelte enhed.

### Aktiver og deaktiver enhed

Skitsen på figur 2.7 viser brugerens mulighed for at aktivere og deaktivere enkelte enheder i systemet.

### Tilføj/fjern enheder

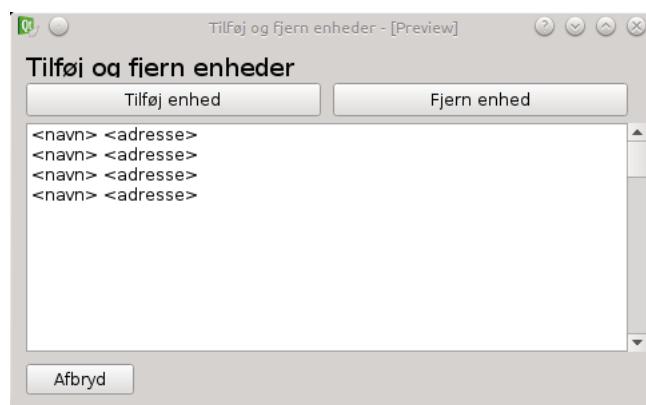
De præsenterede muligheder i forbindelse med at fjerne og tilføje enheder til systemet er vist på figur 2.8.



**Figur 2.6.** Skitse af ”Vis status” på GUI



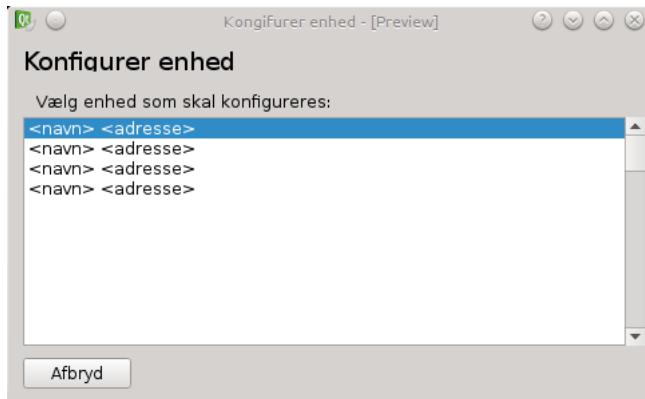
**Figur 2.7.** Skitse af ”Aktiver og deaktiver enhed” på GUI



**Figur 2.8.** Skitse af ”Tilføj/fjern enheder” på GUI

## Konfigurer

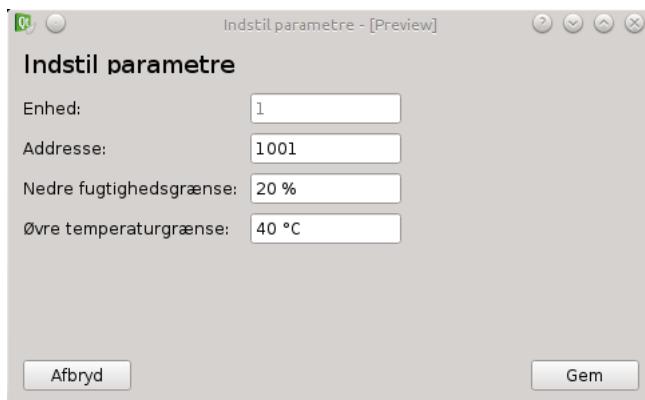
Når brugeren skal konfigurere en enhed bruges skitsen på figur 2.9. Her vælger brugeren hvilken enhed der skal konfigureres.



**Figur 2.9.** Skitse af "Konfigurer" på GUI

## Indstil parametre

Når brugeren har valgt en enhed som skal konfigureres vises skærmen på figur 2.10. Her kan brugeren indtaste parametrene for enheden.



**Figur 2.10.** Skitse af "Indstil parametre" på GUI

## Log

Loggen præsenterer de indsamlede data fra alle enhederne. Det er muligt at se alle data på en gang, som vist på figur 2.11, hvor den øverste fanerække er numre på opsatte enheder i systemet, eller man kan vælge de enkelte enheder og se deres forskellige komponentpakker, som vist på figur 2.12, hvor den anden fanerække er komponentpakker koblet op på systemet.

The screenshot shows a software window titled 'Log - [Preview]'. At the top, there is a toolbar with icons for help, zoom, and close. Below the title is a header 'Log' with tabs for 'Alle', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', and '11'. The 'Alle' tab is selected. The main area is a table with columns: Dato, Enhed, KP, Temp, Fugt, Bevægelse, and Sprinkler. The data rows are:

Dato	Enhed	KP	Temp	Fugt	Bevægelse	Sprinkler
08-10-14 12:15	1	1	25	30%	Ingen	Tændt
08-10-14 12:15	1	2	31	10%	Ingen	Tændt
08-10-14 12:15	2	1	26	40%	Ingen	Slukket
08-10-14 12:30	3	1	29	60%	Registreret	Slukket
08-10-14 12:30	4	1	23	40%	Ingen	Tændt

At the bottom are two buttons: 'Afbryd' (Cancel) and 'Generer graf' (Generate graph).

Figur 2.11. Skitse af "Log" for alle enheder på GUI

The screenshot shows a software window titled 'Log - [Preview]'. At the top, there is a toolbar with icons for help, zoom, and close. Below the title is a header 'Log' with tabs for 'Alle', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', and '11'. The '1' tab is selected. The main area is a table with columns: Enhed, KP, Temp, Fugt, Bevægelse, and Sprinkler. The data rows are:

Enhed	KP	Temp	Fugt	Bevægelse	Sprinkler
1	25	30%	Ingen	Tændt	
1	31	10%	Ingen	Tændt	
1	26	40%	Ingen	Slukket	
1	29	60%	Registreret	Slukket	

At the bottom are two buttons: 'Afbryd' (Cancel) and 'Generer graf' (Generate graph).

Figur 2.12. Skitse af "Log" for enkelt enhed på GUI



# Forundersøgelse 3

## 3.1 Sprinkler

<b>Løsning</b>	1/2" Hunter PS
<b>Producent</b>	Hunter Industries / Agrometer A/S
<b>Tilslutning</b>	1/2" gevindmuffe
<b>Beskrivelse</b>	Simpel pop-up sprinkler som gemmes i græsset og kommer op når der påtrykkes vand. Kan indstilles til at vande i 45-360 grader eller kvadratisk.
<b>Krav</b>	Ekstern vandpumpe
<b>Fordele</b>	Simpel i opsætning og nem kaskadekobling af flere sprinklere hvor kun pumpens kapacitet skal udvides.
<b>Ulemper</b>	-
<b>Pris</b>	Cirkulær: 66,25 kr. Rektangulær: 77,50 kr.
<b>Link</b>	<a href="http://goo.gl/bzS1By">http://goo.gl/bzS1By</a>



### 3.1.1 Sprinkler valg

Hunter PS sprinkleren er en simpel løsning som nemt skjules på banen og er meget fleksibel når den skal tilpasses de enkelte golfhuller.

### 3.2 Temperatursensor (SK)

<b>Løsning</b>	LM75, Digital temperatursensor med two-wire interface.
<b>Producent</b>	National semiconductors
<b>Tilslutning</b>	I2C
<b>Beskrivelse</b>	Hardware modul der kan tilkobles enheden via I2C
<b>Krav</b>	Indgående kendskab til I2C og PSoC creator
<b>Fordele</b>	Der er blevet lavet en øvelse i GFV, der omhandler denne type sensor.
<b>Ulemper</b>	Den kommunikerer over I2C som er mere besværligt at arbejde med. Der skal konstrueres en "beholder" til sensoren så den kan tåle at komme ned i jorden.
<b>Pris</b>	Hentet fra værkstedet på IHA
<b>Link</b>	<a href="http://datasheets.maximintegrated.com/en/ds/LM75.pdf">http://datasheets.maximintegrated.com/en/ds/LM75.pdf</a>



<b>Løsning</b>	DS18B20 temperatursensor.
<b>Producent</b>	Dallas
<b>Tilslutning</b>	-
<b>Beskrivelse</b>	Vandtæt temperaturprobe der kan tilkobles enhver microcontroller med en enkelt digital benforbindelse
<b>Krav</b>	Indgående kendskab PSoC creator.
<b>Fordele</b>	Der er mulighed for at tilslutte flere af dem til den samme forbindelse, den kræver ingen ekstra HW.
<b>Ulemper</b>	Dallas 1-Wire-protokollen er lidt kompliceret, og kræver en del kode for at trække kommunikationen ud
<b>Pris</b>	89 kr
<b>Link</b>	<a href="https://elextre.dk/main.aspx?page=article&amp;artno=H15942">https://elextre.dk/main.aspx?page=article&amp;artno=H15942</a>



### 3.3 Bevægelsessensor (SK)

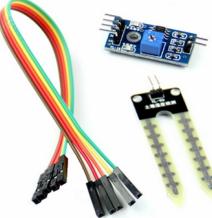
<b>Løsning</b>	HC-SR501, PIR-bevægelsessensor
<b>Producent</b>	Vides ikke
<b>Tilslutning</b>	-
<b>Beskrivelse</b>	En PIR-sensor, når bevægelse er registeret gives et højt eller lavt output(High 3.3 V /Low 0V), alt efter hvad jumperen er indstillet til
<b>Krav</b>	Indgående kendskab til PSoC creator
<b>Fordele</b>	Denne PIR-sensor kræver ingen ekstra HW. Ju-sterbar følsomhed og forsinkelse via potentiome-ter
<b>Ulemper</b>	-
<b>Pris</b>	Hentet fra værkstedet på IHA
<b>Link</b>	<a href="http://goo.gl/c6GyTL">http://goo.gl/c6GyTL</a>



#### 3.3.1 Bevægelsessensor valg

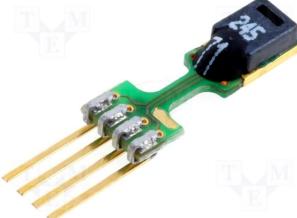
HC-SR501, PIR-bevægelsessensor. Opfylder alle de behov der er brug for, den er lille og kompakt dertil kan følsomhed/forsinkelse let indstilles.

### 3.4 Fugtsensor (SK og LB)

<b>Løsning</b>	Jord-hygrometer modul til Arduino
<b>Producent</b>	Vides ikke
<b>Tilslutning</b>	-
<b>Beskrivelse</b>	En jord-fugtighedssensor, når fugtigheden er høj giver output modulet en høj værdi og omvendt ved lav fugtighed
<b>Krav</b>	Indgående kendskab til PSoC creator
<b>Fordele</b>	Denne fugtighedssensor er klar til at blive sat i jorden, kræver ingen ekstra HW. Justerbar følsomhed via potentiometer
<b>Ulemper</b>	-
<b>Pris</b>	79 kr
<b>Link</b>	<a href="http://eleextra.dk/main.aspx?page=article&amp;artno=H11086">http://eleextra.dk/main.aspx?page=article&amp;artno=H11086</a>
	

<b>Løsning</b>	SHT21P - Fugtighed- og temperatursensor IC
<b>Producent</b>	Sensirion
<b>Tilslutning</b>	Analog
<b>Beskrivelse</b>	Hardware modul der giver et analog output (PWM)
<b>Krav</b>	Indgående kendskab til analog signaler og PSoC creator
<b>Fordele</b>	Fugtighed- og temperatursensor samlet i en lille kompakt enhed.
<b>Ulemper</b>	Der skal konstrueres noget HW og en "beholder" til sensoren så den kan tåle at komme ned i jorden
<b>Pris</b>	27.55
<b>Link</b>	<a href="http://www.farnell.com/datasheets/1847262.pdf">http://www.farnell.com/datasheets/1847262.pdf</a>
	

<b>Løsning</b>	SHT71P - Fugtighed- og temperatursensor IC
<b>Producent</b>	Sensirion
<b>Tilslutning</b>	I2C Variation
<b>Beskrivelse</b>	Hardware modul der kommunikeres med via en I2C variation
<b>Krav</b>	Indgående kendskab til I2C kommunikation og PSoC creator
<b>Fordele</b>	Fugtighed- og temperatursensor samlet i en lille kompakt enhed.
<b>Ulemper</b>	Der skal konstrueres noget HW og en "beholder" til sensoren så den kan tåle at komme ned i jorden
<b>Pris</b>	Hentes i værkstedet
<b>Link</b>	<a href="http://www.sensirion.com/fileadmin/user_upload/customers/sensirion/Dokumente/Humidity/Sensirion_Humidity_SHT7x_Datasheet_V5.pdf">http://www.sensirion.com/fileadmin/user_upload/customers/sensirion/Dokumente/Humidity/Sensirion_Humidity_SHT7x_Datasheet_V5.pdf</a>



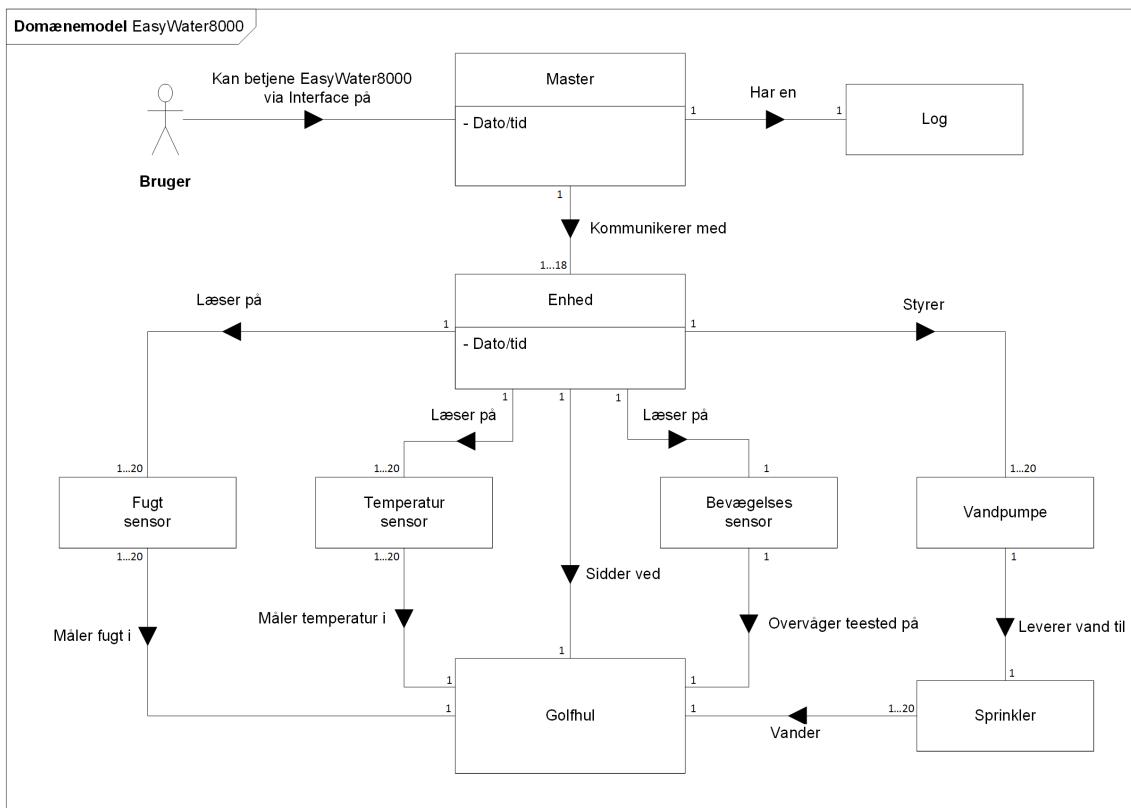
### 3.4.1 Fugtsensor valg

SHT21p. Har den store fordel at både fugt og temperatur er samlet i et, samt at den leverer et analog signal som er nemt at aflæse.



# Systemarkitektur 4

## 4.1 Domænemodel (JC)

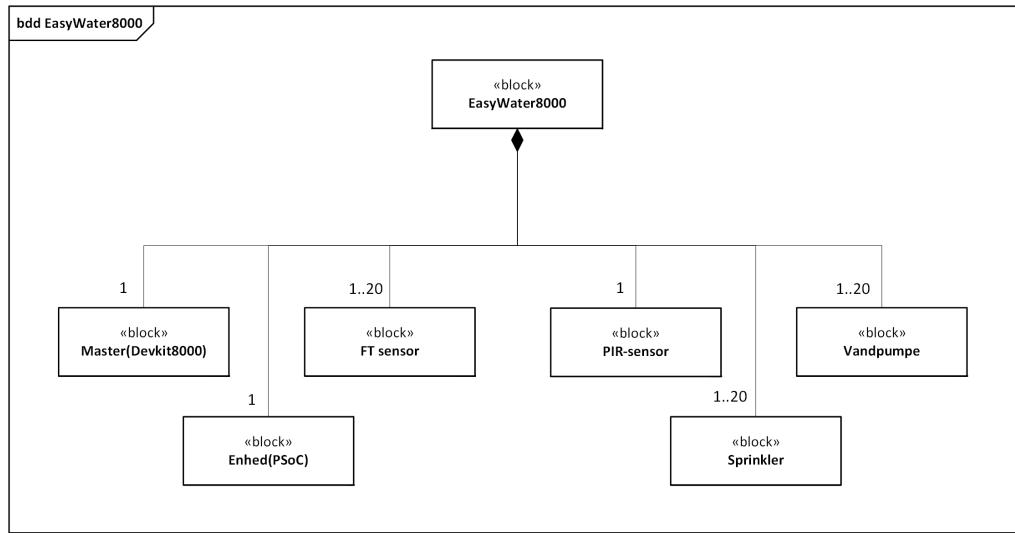


*Figur 4.1.* Domænemodel

Domænemodellen i figur 4.1 beskriver EasyWater8000 systemets funktionalitet og de forskellige deles indbyrdes sammenhæng.

## 4.2 Hardware beskrivelse

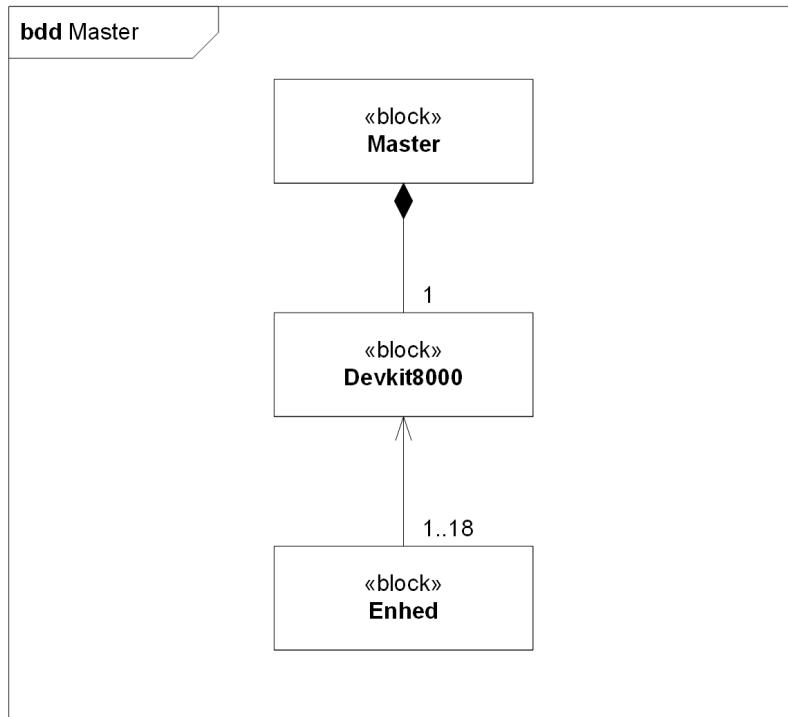
### 4.2.1 BDD (SK)



*Figur 4.2.* BDD

BDD diagrammet giver et overblik over hvad det samlede EasyWater8000 system består af, samt indbyrdes multiplicitet. FT sensoren er en kombineret fugt- og temperatursensor.

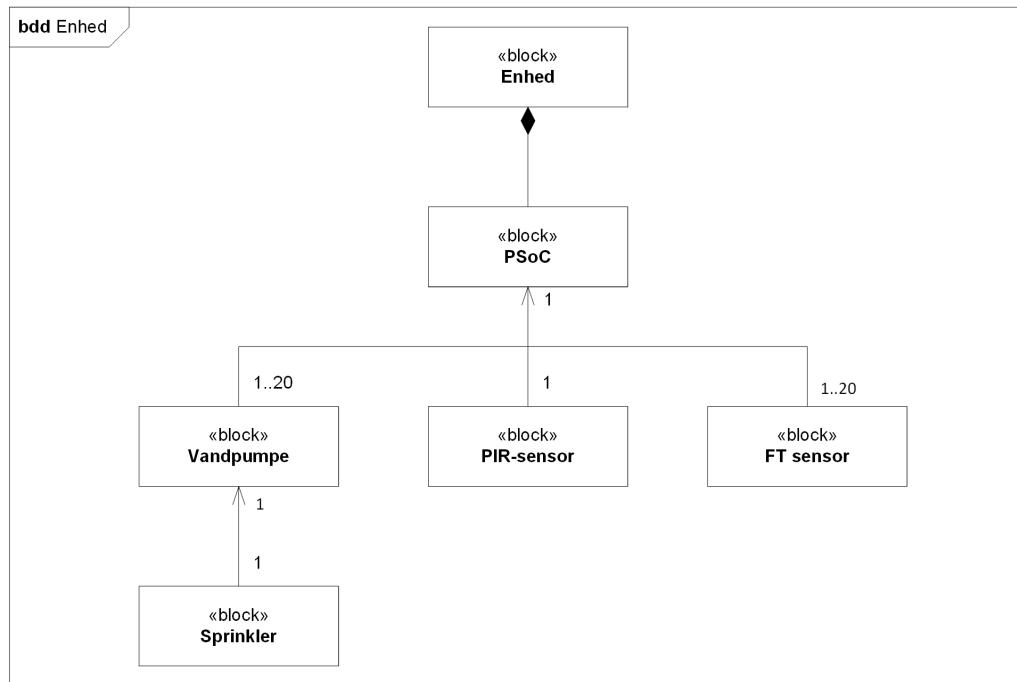
### 4.2.2 BDD Master (MK)



*Figur 4.3.* BDD Master

BDD diagrammet for Master, viser at Master består af ét Devkit8000, som kobles op med en til 18 Enheder.

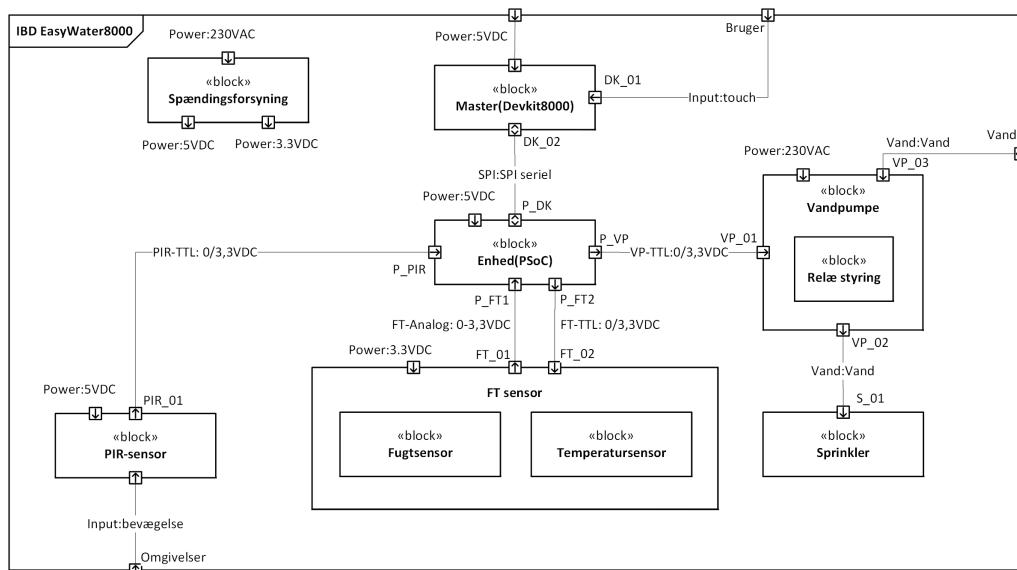
#### 4.2.3 BDD Enhed (MK)



Figur 4.4. BDD Enhed

BDD diagrammet for Enhed, viser at én Enhed består af én PSoC som kan kobles sammen med op til 20 Vandpumper, 20 FT sensorer (Fugt- og Temperatursensorer) samt 1 PIR-sensor. Sprinkler bliver i systemet koblet sammen med én Vandpumpe.

#### 4.2.4 IBD (SK)



Figur 4.5. IBD

IBD diagrammet giver et internt overblik over hvordan hele systemet er forbundet. Der ses hvilke typer signaler der bliver sendt imellem de forskellige blokke.

**Spændingsforsyning:** Spændingsforsyningens forbindelser er ikke påtegnet da dette ville give et uoverskueligt diagram, de er i stedet beskrevet med standardflowports. Spændingsforsyningen forsyner enhed (PSoC), FT sensor, PIR-sensor samt relæet i vandpumpen. Master(devkit8000) har sin egen spændingsforsyning.

**FT sensor:** Blokken beskriver at fugt- og temperatursensor er integreret i en chip.

**Relæ styring:** Blokken beskriver at vandpumpen består af et relæ, der står for at tænde/slukke for den.

### 4.2.5 Grænseflade (HW)

For at opnå forståelse for signalerne mellem blokkene laves en grænseflade beskrivelse, der beskriver de enkelte blokkes porte og hvilke signaler der løber imellem disse.

#### Blok beskrivelse (HW)

Til at beskrive blokkene nærmere anvendes tabeller som ses herunder. Her er hvert signal i en respektiv blok kommenteret og blokkens funktion er kort beskrevet.

**Tabel 4.1.** Tabel med beskrivelse af respektive blokke

Bloknavn	Funktion	Signaler	Kommentar
Master(Devkit8000)	Styreenhed der modtager input fra touchskærm og kommunikere serielt med Enhed(er)	Input:Touch	Touch
		SPI:SPI seriel	Seriel data kommunikation
Enhed(PSoC)	Enhed er grænsefladen til den fysiske verden igennem sensorer	SPI:SPI seriel	Seriel data kommunikation
		PIR-TTL	Signal fra PIR-sensor
		VP-TTL	Signal til vandpumpe
		FT-TTL	Signal til FT-sensor
		FT-Analog	Signal fra FT-sensor
FT-sensor	Måler temperatur og fugt	FT-Analog	Analog signal til Enhed
		FT-TTL	Signal fra Enhed
PIR-sensor	Detektere bevægelse	PIR-TTL	Signal til Enhed
Vandpumpe	Forsyner sprinkler med vand	Vand:Vand	Vand til Sprinkler
Sprinkler	Fordeler vand på golfbane	Vand:Vand	Vand til Golfbane
Spændingsforsyning	Forsyner EasyWater8000, Master har sin egen spændingsforsyning	Power:3,3VDC	Forsyning til FT-sensor
		Power:5VDC	Forsyning til PIR-sensor, Enhed

#### 4.2.6 Signal beskrivelse (HW)

For at fuldende beskrivelsen af grænsefladen er der lavet en signaltabel som kan ses herunder. Hvert signal er beskrevet. Området et signal er defineret under, er også beskrevet. Blok og terminal indgår også.

**Tabel 4.2.** Tabel over signaler med terminaler

Signal-navn	Funktion	Område	Port 1	Port 2	Kommentar
SPI:SPI seriel	Seriel kommunikation		Master (DK_02)	Enhed (P_DK)	
PIR-TTL	TTL	0/3,3V	PIR-sensor (PIR_01)	Enhed (P_PIR)	
VP-TTL	TTL	0/3,3V	Enhed (P_VP)	Vandpumpe (VP_01)	
FT-TTL	TTL	0/3,3V	Enhed (P_FT2)	FT-sensor (FT_02)	
FT-Analog	Analog	0-3,3V +/-10%	FT-sensor (FT_01)	Enhed (P_FT1)	

## 4.3 Software beskrivelse

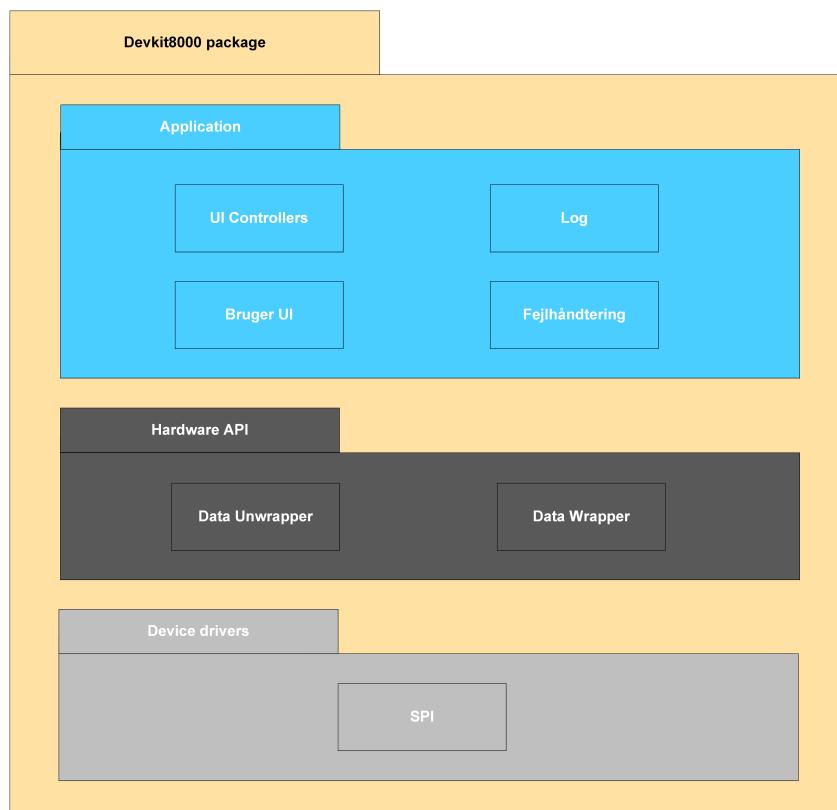
For at danne overblik over software-udviklingen inden det egentlige design, laves bruges N+1 modellen<sup>1</sup>. Denne beskriver fire faser som tager hånd om de overordnede ting inden for software, alt sammen med use-cases som den røde tråd. De fire faser er:

1. Logical View
2. Deployment View
3. Implementation View
4. Data View

Ud over disse punkter tænkes der også en overordnet fejl-håndtering ind i projektet. Disse punkter er beskrevet i detaljer herefter.

### 4.3.1 Logical View (JC)

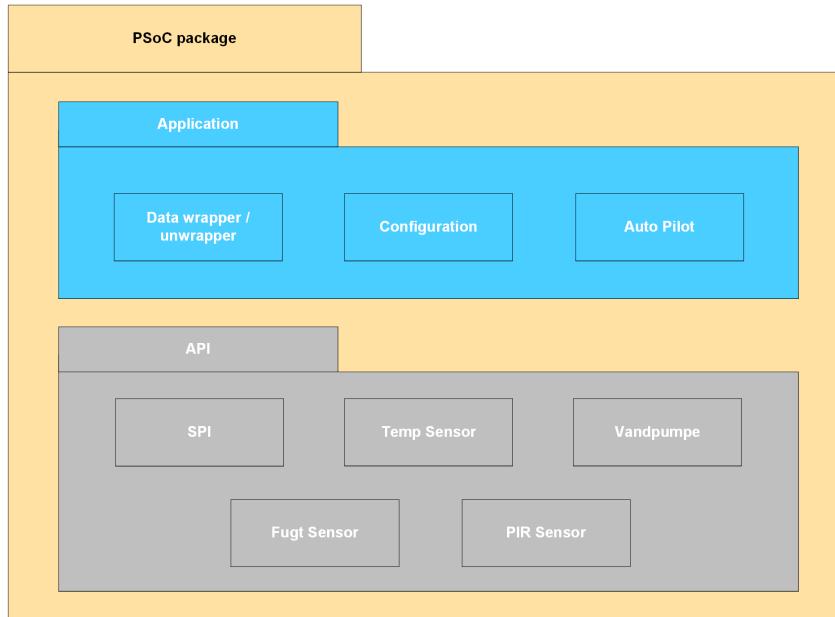
Logical View skal danne et overblik over hvilke softwarepakker der befinner sig på vores platforme. Blokkene inde i de respektive pakker kan sammen med domænemodellen hjælpe med at give et overblik over hvilke klasser og kernemoduler der skal bruges.



**Figur 4.6.** Logical view for Devkit8000 illustrerer hvilke softwarepakker der befinner sig på Masteren

Figur 4.6 illustrerer hvilke softwarepakker der ligger på Masteren. I bunden er *Device drivers*-pakken som håndterer SPI kommunikationen imellem Devkit8000 og PSoC. I midten ligger *Hardware API*-pakken som håndterer protokol-vedtægter ifm. kommunikationen. *Application*-pakken tager sig af alt UI samt log og fejlhåndtering.

<sup>1</sup><http://goo.gl/kU89oe> [2014-11-02]

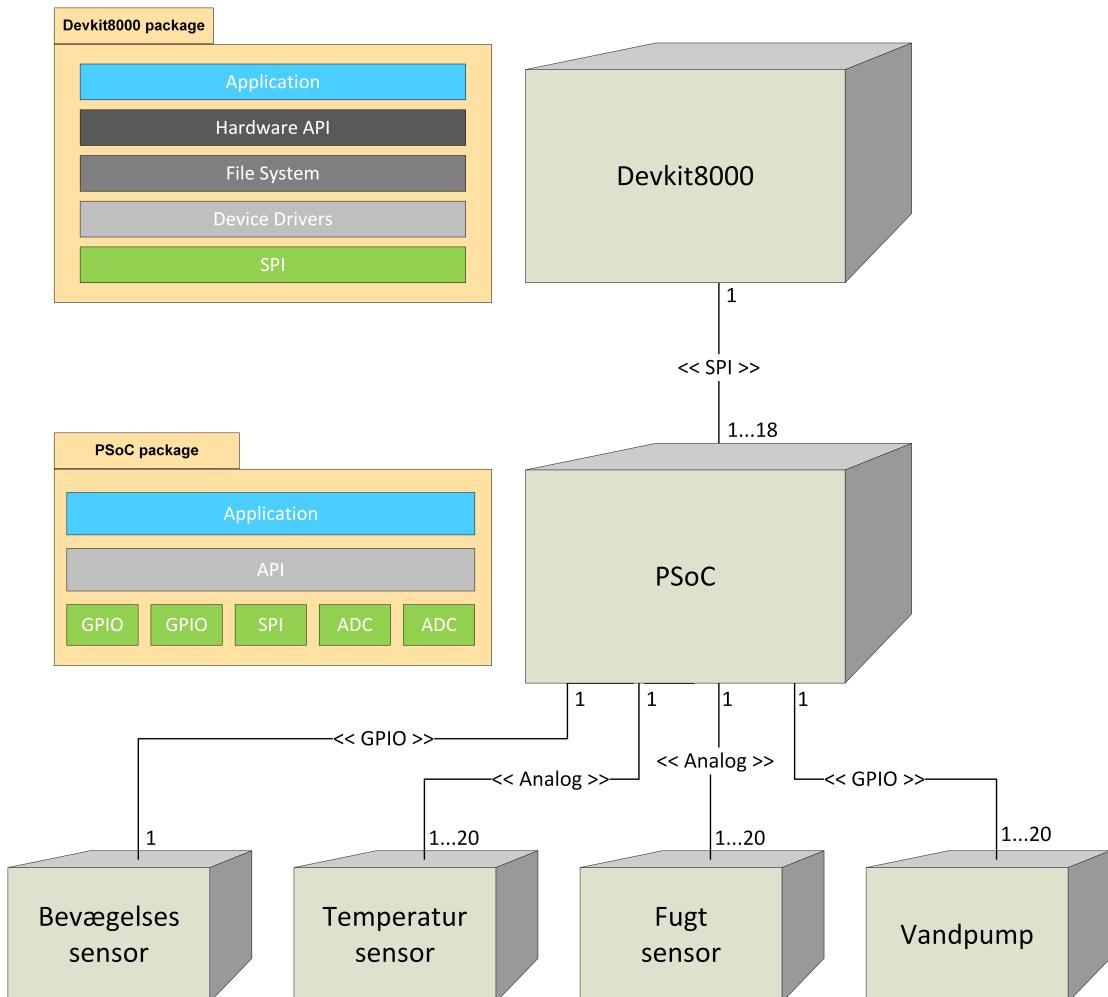


**Figur 4.7.** Logical view for PSoC illustrerer hvilke software pakker der befinder sig på enhederne

Figur 4.7 illustrerer hvilke softwarepakker der ligger på Enhederne. *API*-pakken består af den software som håndterer hardwaren, dvs. den tager imod input og får formateret det til noget brugbart for *Application*-pakken. *Application*-pakken håndterer den indsamlede data som den får fra sensorene igennem *API*-pakken. Denne data sammensættes iht. protokollen og sendes til *API* pakken som får det sendt til Devkit8000. Pakken skal også håndterer data fra Devkit8000 til at konfigurerer de parametre der styrer automatiseringen af vandingen.

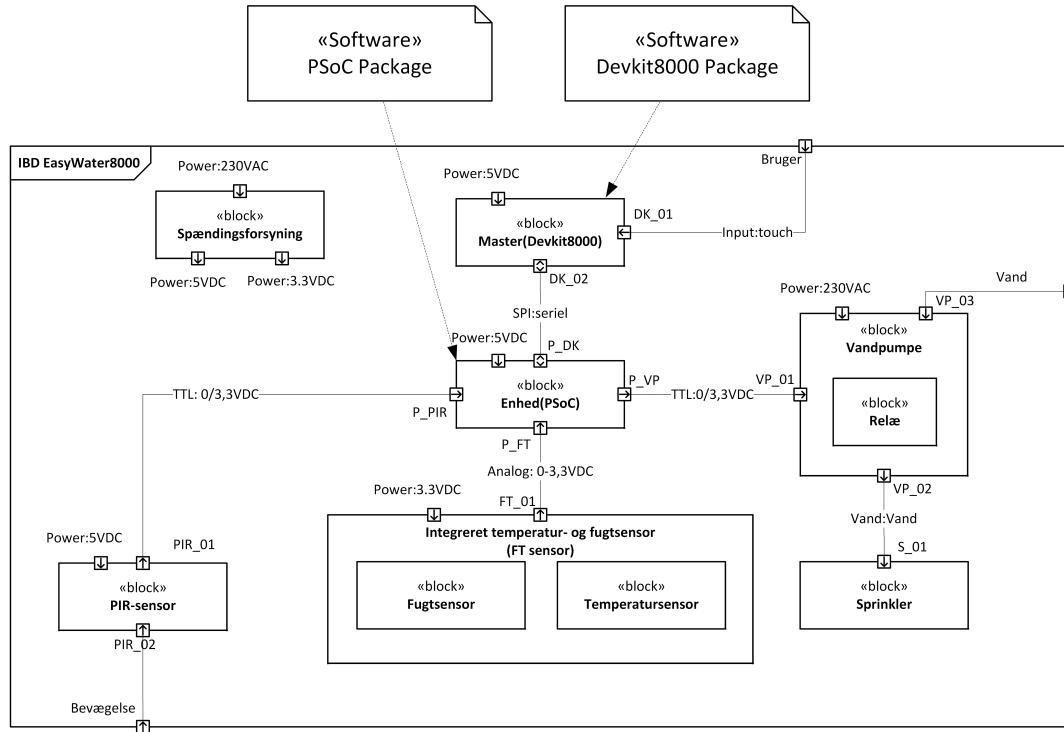
### 4.3.2 Deployment View (JC)

Deployment View skal illustrerer hvor hvilke lag af software der ligger på vores platforme. Devkit8000 kører Linux og har derfor flere softwarelag end PSoC'en.



**Figur 4.8.** Deployment model illustrerer de forskellige software og hardware(grønne) lag

Figur 4.8 viser hvilke lag der er i de respektive pakker og hvor pakkerne hører til. *File system* er en del af Linux systemet og ikke noget der skal implementeres.

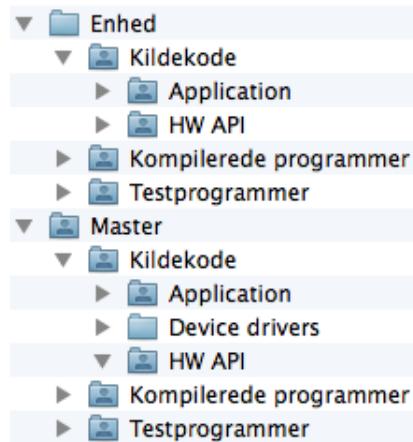
**Figur 4.9.** IBD med software packages

Figur 4.9 viser pakkernes placering på det interne blokdiagram, så man kan se præcist hvor softwaren kommer til at ligge på hardwaren.

### 4.3.3 Implementation View (BS)

Inden programmerne designes, fastlægges en struktur for kildekoden. På den måde er det nemmere for flere programmører at arbejde med delene i programmet samtidigt.

Strukturen skal være som vist i figur 4.10. Under mappen ”Kildekode” skal hver klasse have en mappe med der til hørende filer. Ligeledes med ”Testprogrammer” mappen, som indholder testprogrammer som verificerer funktionaliteten af de enkelte moduler. Mappen ”Kompilerede programmer” er til de endelige programmer.

**Figur 4.10.** Mappestruktur for software

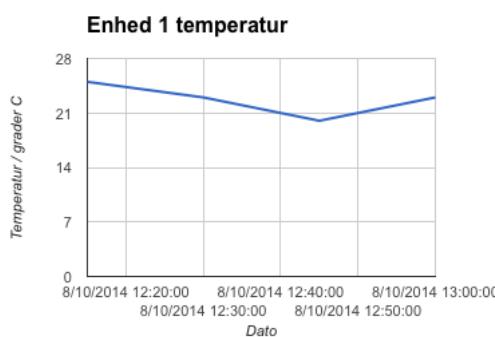
#### 4.3.4 Data View (BS)

I forbindelse med EasyWater8000s log skal der gemmes data på en nem og håndterbar måde. Det skal være muligt at gemme følgende data:

1. Tidsstempel
2. Temperatur
3. Fugtighed
4. Bevægelse
5. Vanding

Ydermere skal disse informationer gemmes for hver enhed. Så hvis der er 18 huller med i alt 18 enheder, skal ovenstående gemmes for alle 18 enheder.

Når informationen skal præsenteres for bruger skal det ske i en tabel som vist på figur 2.11 i afsnit 2.5, data for en enkelt enhed som på figur 2.12 i afsnit 2.5 eller på en graf så man kan se ændringer over tid som vist på figur 4.11.



**Figur 4.11.** Graf for temperatur for én enhed

Tabellen skal have en fane for hver opkoblet enhed. Her kan man se informationer fra hver enkelt enhed. Det bør også være muligt at se alle enheders informationer samtidigt.

Dataen struktureres i semikolon-separerede filer (.csv) på Master. Hver enhed har en fil hvor alle data er samlet med udgangspunkt i strukturen vist i liste 4.1.

**Listing 4.1.** Semikolon-separeret datafil til log af enheder

```

1 <enheds-nr>;
2 <KP-nr>; <dato>; <temperatur>; <fugtighed>; <bevaeglse>; <vanding>;
3 <KP-nr>; <dato>; <temperatur>; <fugtighed>; <bevaeglse>; <vanding>;
4 ...
5 <KP-nr>; <dato>; <temperatur>; <fugtighed>; <bevaeglse>; <vanding>;

```

Dette resulterer i en filstruktur som vist på liste 4.2 hvis der er koblet 18 enheder op på Master.

**Listing 4.2.** Filstruktur for logfiler på Master

```

1 <log>/
2   <enheds-nr1>.csv
3   <enheds-nr2>.csv
4   ...
5   <enheds-nr17>.csv
6   <enheds-nr18>.csv

```

Hæufigheden for målingerne og logningen er beskrevet i de ikke-funktionelle krav, 2.4.

### 4.3.5 Fejl-håndtering (JC)

Systemet kan håndterer fejl og disse vil blive gemt i en fejllog som bliver gemt på Master.

Fejlhåndteringen bliver klaret af en klasse på Devkit8000 som håndterer at skrive det rigtige fejl ud i en .txt-fil. Klassen vil blive kaldt med en fejlkode hver gang fejl opstår. Klassen forstår så at skrive den rigtige fejl ind i txt filen ud fra den pågældende fejlkode den har modtaget som attribut.

Alle funktioner vil returnere et negativt heltal som repræsenterer en fejlkoden som klassen kan tolke på.

```
Fejl -32 registreret. Temperatur uden for range,
Fejl -12 registreret. Kunne ikke åbne log fil til enhed 2.
```

**Figur 4.12.** udsnit af Error log

Billedet på figur 4.12 viser hvordan 2 fejl i error-loggen kunne se ud.

### 4.3.6 Kommunikationsprotokol (BS)

En del data skal flyttes mellem Master og de tilkoblede Enheder. Her følger beskrivelsen af hvordan data pakkes mellem de to dele. Den elektriske protokol som anvendes er SPI.

Opsætningen er som følger: *Hastighed SPI mode Antal bits*

Ud fra UC-beskrivelserne er der identificeret følgende scenarier hvor der sendes data mellem Master og Enhed.

1. Master kontrollerer om Enhed er koblet til systemet (UC1)
2. Master sender parametre til Enhed (UC2)
3. Enhed aktiveres eller deaktiveres af Master (UC3)
4. Master beder om data fra Enhed (UC4)
5. Enhed sender data til Master (UC4)

Fælles for alle kommandoer er start- og stopkarakteren. Her vælges karakterene vist i tabel 4.3.

**Tabel 4.3.** Start- og stopbytes for SPI-kommunikationen

	ASCII	Hex
<b>STX</b>	'S' / 's'	0x53 / 0x73
<b>ETX</b>	'\r'	0x0D

Dataen til og fra enhederne pakkes i nogle frames som beskrevet i tabel 4.10. Først sendes STX, der efter kommer længden af det efterfølgende data. Her efter kommer kommandoen og evt. data i tilfælde af UC2 og UC4, og der afsluttes med ETX.

**Tabel 4.4.** Data formatering for SPI-kommunikation

Byte	0	1	2	3..X	X + 1
Indhold	STX	<Længde>	<Kommando>	<Data>	ETX

### Blokken <Længde>

Længde-byten bestemmer hvor meget data der sendes med den pågældende kommando. Hvis der ikke sendes andet end selve kommandoen er den 0. Bemærk at dette skal fortolkes som et heltal og ikke en karakter! Der skal sendes *4* ikke '*4*'.

### Blokken <Kommando>

Alle kommandoer er én byte lang og kommandoerne i tabel 4.12 er de tilgængelige kommandoer. Hvis ikke kommandoen genkendes er der intet svar.

*Tabel 4.5.* Kommandoer for SPI-kommunikation

ASCII	HEX	Funktion	Afsender
'A' / 'a'	0x41 / 0x61	Aktiver Enhed	
'D' / 'd'	0x44 / 0x64	Deaktiver Enhed	
'V' / 'v'	0x56 / 0x76	Verifier Enhed i systemet	
'P' / 'p'	0x50 / 0x70	Parametre sendes til Enhed	
'L' / 'l'	0x4c / 0x6c	Forespørg logdata fra Enhed	
'M' / 'm'	0x4d / 0x6d	Returner antal af bytes i buffer	

### Blokken <Data>

Data-blokken bruges til at sende data mellem enhederne. Her følger beskrivelsen for hvordan denne formateres. Bemærk den kun bruges til kommandoen '*P*' (UC2) og '*L*' (UC4).

#### Send parametre

Her skal sendes følgende parametre til Enheden.

1. Nedre fugtighedsgrænse
2. Øvre temperaturgrænse

Disse består af tocifrede tal i hhv. procent og grader Celsius. Parametrene skal sendes i ovenstående rækkefølge og resulterer altså i en kommando som vist i tabel 4.13.

*Tabel 4.6.* Data-formatering for parameter-kommando

Byte	0	1	2	3..4	5..6	7
Indhold	STX	4	P	<Fugt>	<Temp>	ETX

#### Send log

Når masteren skal udhente afventende data i Enheden ved denne ikke på forhånd hvor meget data der skal modtages. Da Masteren kun kan starte kommunikation i SPI startes med at sende en '*L*' kommando og efterfølgende læses én byte som returneres af Enheden. Her i står antallet af bytes som skal læses fra Enheden. Masteren påbegynder der efter en række læsninger svarende til antallet af bytes.

Et forløb kunne være som følgende. Master sender strengen "*SOL|r*" og læser her efter bytes indtil ETX modtages. Svaret fra Enheden kan være på to former. Den ene er retursvar for logning af KP-data. Denne er vist i tabel 4.7. Det kan også være en bevægelse der er registreret. Denne formateres som vist i tabel 4.8.

**Tabel 4.7.** Data-formatering for KP-returværdi på Hent log kommando

Byte	0	1	2..3	4..5	6
Indhold	<KP-nr>	<Tid>	<Fugt>	<Temp>	<Sprinkler>

**Tabel 4.8.** Data-formatering for Bevægelses-returværdi på Hent log kommando

Byte	0	1
Indhold	<Bevægelse>	<Tid>

### 4.3.7 Kommunikationsprotokol (MIPO)

En del data skal flyttes mellem Master og de tilkoblede Enheder. Her følger beskrivelsen af hvordan data pakkes mellem de to dele. Den elektriske protokol som anvendes er SPI.

Opsætningen er som følger: *Hastighed SPI mode Antal bits*

Ud fra UC-beskrivelserne er der identificeret følgende scenarier hvor der sendes data mellem Master og Enhed.

1. Master kontrollerer om Enhed er koblet til systemet (UC1)
2. Master sender parametre til Enhed (UC2)
3. Enhed aktiveres eller deaktiveres af Master (UC3)
4. Master beder om data fra Enhed (UC4)
5. Enhed sender data til Master (UC4)

SPI kommunikationen skal bestå af 2 metoder. En write funktion som står for at skrive til Enheden. Og en read funktion som står for at udlæse data fra Enheden.

**Tabel 4.9.** De to metoder for SPI-kommunikationen

	ASCII	Hex
WRITE	'W' / 'w'	0x57 / 0x77
READ	'R' / 'r'	0x52 / 0x72

Dataen til og fra enhederne pakkes i nogle frames som beskrevet i tabel 4.10. Først sendes write/read metoden. Herefter kommer kommandoen. Og data i det tilfælde at parametrene på Enheden skal ændres

**Tabel 4.10.** Data formatering for SPI-kommunikation

Byte	0	1	2	3
Indhold	W/R	<Kommando>	<Data>	<Data>

#### Blokken <Kommando>

Alle kommandoer er én byte lang og kommandoerne i tabel 4.12 er de tilgængelige kommandoer. Hvis ikke kommandoen genkendes er der intet svar.

**Tabel 4.11.** Retur data for SPI-kommunikation

Byte	0	1	2	3
Indhold	<Data>	<Data>	<Data>	<Data>

**Tabel 4.12.** Kommandoer for SPI-kommunikation

ASCII	HEX	Funktion	Metode
'A' / 'a'	0x41 / 0x61	Aktiver Enhed	Write
'D' / 'd'	0x44 / 0x64	Deaktiver Enhed	Write
'P' / 'p'	0x50 / 0x70	Parametre sendes til Enhed	Write
'V' / 'v'	0x56 / 0x76	Verifier Enhed i systemet	Read
'L' / 'l'	0x4c / 0x6c	Forespørg logdata fra Enhed	Read

### Blokken <Data>

Data-blokken bruges til at sende data mellem enhederne. Her følger beskrivelsen for hvordan denne formateres. Ved Aktiver og Deaktiver sendes ingen <data>.

#### Send parametre

Når parametrene på Enheden skal ændres sendes <data>.

Her skal sendes følgende parametre til Enheden.

1. Nedre fugtighedsgrænse
2. Øvre temperaturgrænse

Figur 4.13 viser formateringen for write metoden, når parametrene skal ændres på Enheden. Den nedre fugtighedsgrænse skal bestå af en to cifret procent sats. Den øvre temperaturgrænse består af en to cifret temperatur i °C.

**Tabel 4.13.** Data-formatering for parameter-kommando

Byte	0	1	2	3
Indhold	W	P	<Fugt>	<Temp>

#### Send log

Masteren beder Enheden om at sende log data. Dette gøres ved at benytte read metoden. Tabel 4.14 viser indholdet af metodekaldet.

**Tabel 4.14.** Metode kald for send log-kommando

Byte	0	1	2	3
Indhold	R	L	NULL	<KP-nummer>

Når ovenstående modtages af Enheden, retunereres log data fra Enheden iht. tabel 4.7.

**Tabel 4.15.** Data-formatering for KP-returværdi på Send log kommando

Byte	0	1	2	3
Indhold	<KP-nummer>	<Fugt>	<Temp>	<Sprinkler><Bevægelse>



# Hardwaredesign

5

## 5.1 Prototypemodel

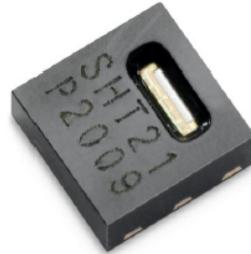
Hardwaredesignet tager udgangspunkt i den ønskede prototype, der beskrives herunder.

Prototypen skal bestå af en Master (DevKit8000) der via SPI kommunikerer med én Enhed (PSoC4). Denne enhed har koblet én komponentpakke, bestående af én PIR-sensor (HC-SR501), én kombineret fugt-, og temperatursensor (SHT21P), én 230V/5V relæstyring, én vandpumpe (Alpha2) samt én sprinkler (Hunter PS).

Prototypen sætter herved begrænsninger i forhold til ovenstående beskrivelser af systemarkitekturen.

De enkelte dele af prototypen designes herunder.

## 5.2 Temperatur- og fugtsensor

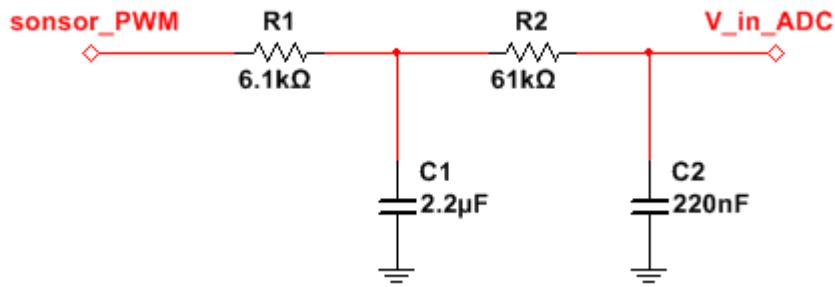


*Figur 5.1.* Fysisk afbildning af SHT21P

Til indsamling af temperatur- og fugtdata for golfhuller anvendes den kombineret temperatur og fugtsensor SHT21P. Denne er valgt ud fra at der således ikke behøves en sensor for hver af de to målinger, samt at denne giver et analog signal med i designet. SHT21P sender en PWM ud som midles til en DC-spænding med et 2. ordens lavpasfilter. Denne spændingen sendes ind i en A/D-convertor i PSoCen hvor denne behandles i en funktion, for så at returnere en temperatur eller fugt. Et select ben på SHT21P bestemmer hvorvidt denne mäter temp. eller fugt. SCL HIGH(1) giver fugt output, SCL LOW(0) giver temperatur output.

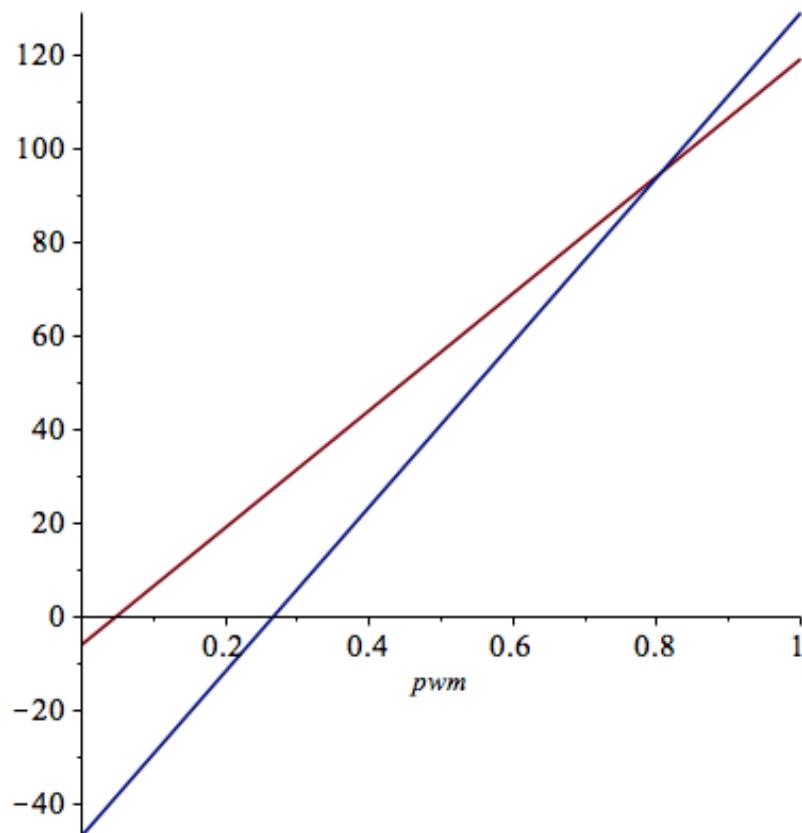
### 5.2.1 Foranliggende filter

Filterets knækfrekvens ( $f_c$ ) er designet ud fra frekvensen på PWM signalet. Denne er opgivet til 120 Hz i databladet. Ved at designe filteret med en  $f_c$  der ligger væsentligt under



**Figur 5.2.** Multisim tegninger af 2. ordens filter

de 120 Hz, vil der opnås en stor dæmpning på amplituden således at signalet tilnærmer sig en fast DC-spænding svarende til middelværdien af PWM. PWM'en oscillerer i området VSS-VDD. Sensoren er forsynet med 3,3 VDC og VSS er forbundet til GND, hvilket giver en PWM i området 0-3,3 VDC. DC-spændingen efter filteret er testet ved 10 % dutycycle og 90 % dutycycle hvilket gav henholdsvis 211 mV og 2610 mV efter filteret. Ved et plot af temperatur og fugt som funktion af PWM kan det ses at de to er lineære. Funktionerne herfor er opgivet i databladet for SHT21P.

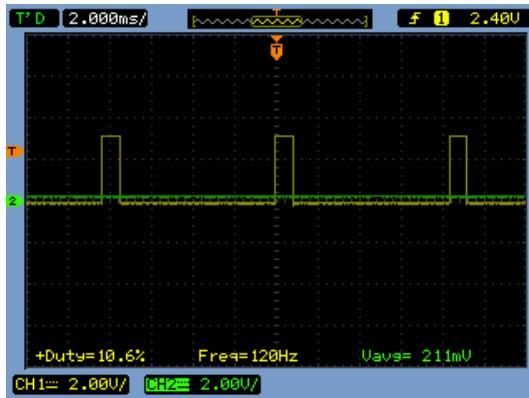


**Figur 5.3.** Plot af fugt(rød kurve) og temperatur(blå kurve) som funktion af pwm

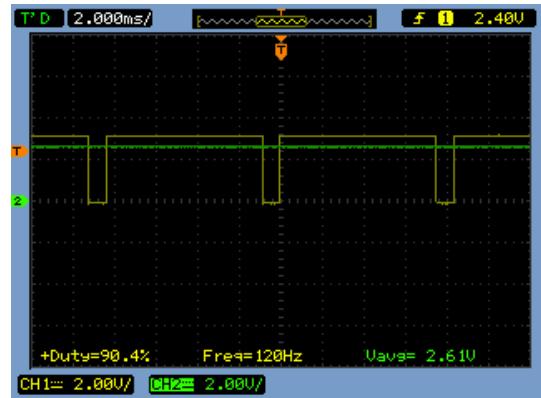
### 5.2.2 Præcisions- og støjhåndtering

En måledifferens for 10 % - 90 % duty cycle ender ud i 2400 mV og en temperaturdifferens på 155 °C og da de er lineære kan et step på 15 mV pr. °C beregnes:

$$step = \frac{2400}{155} = 15,48 = 15 \frac{mV}{^{\circ}C} \quad (5.1)$$



*Figur 5.4.* 10 % duty cycle



*Figur 5.5.* 90 % duty cycle

$$f_c = \frac{1}{2\pi * R1 * C1} = \frac{1}{2\pi * 6,1 * 10^3 * 2,2 * 10^{-6}} = 11,86Hz \quad (5.2)$$

$f_c$  ligger således en dekade under de 120 Hz og da filteret er designet som et 2. ordens filter, vil det dæmpe 40 dB pr. dekade. Amplituden vil da være dæmpet 100 gange og tilbage er den tilnærmet DC værdi.

$$Gain = 20 * \log_{10}(x) \Leftrightarrow 40dB = 20 * \log_{10}(x) \Leftrightarrow x = 100 \quad (5.3)$$

Da det er en dæmpning der er tale om, vil gain være lig med -40 dB hvilket vil give en x-værdi på 0,01 som er det samme som 100 gange dæmpning.

Grundet at værdien kun er tilnærmet skyldes at der stadig vil være en lille smule oscillation fra PWM'en svarende til 33 mV peak-peak. Der vil altså være en usikkerhed i DC-værdien svarende til 2 °C alene fra filteret.

Dette kan løses i softwaren ved evt. at tage et antal samples og så midle de værdier der er målt med simple gennemsnitsberegning.

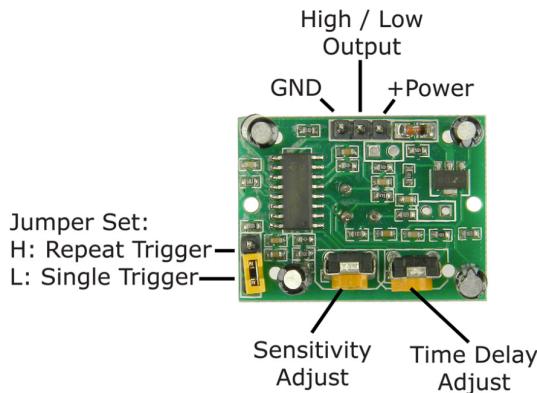
$$V_{inADC_{avg}} = \frac{V_{in1} + V_{in2} + \dots + V_{inN}}{N} \quad (5.4)$$

hvor  $N$  = antal samples

### 5.3 PIR-sensor

Den PIR-sensor(HC-SR501) der er valgt til projektet har 3 ben, +POWER, OUTPUT og GND, 2 potmetre til at indstille sensitiviteten og forsinkelse samt en jumper til at indstille triggeren. Forsyningsspændingen skal ligge mellem 5V - 20V. Forsinkelse kan justeres til mellem 0.3 - 5min.

Output fra PIR-sensoren afgiver et TTL signal på 3.3V ved bevægelse og 0V ved stilstand.



**Figur 5.6.** Billede af selve printet og dets indstillingsmuligheder

### Driver

Driveren der skal drive PIR sensoren på Enhed skal kunne registrere bevægelse via en get-metode der returnere 1(høj) ved bevægelse og 0(lav) ved stilstand.

### Pseudokode

```

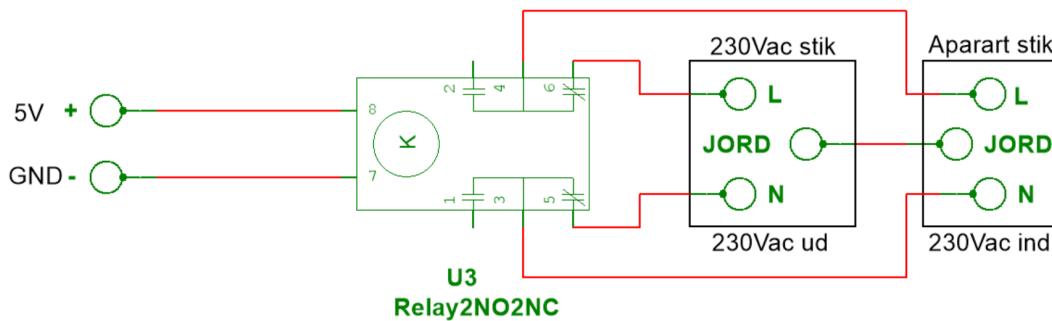
1 int get_pir_status(){
2 Read TTL signal on P_PIR pin
3 Return 1 on movement and 0 on no movement
4 }
```

### 5.4 Sprinkler-pumpe system

Sprinkler-pumpe systemet består af en Hunter PS Pop-up sprinkler og en Alpha 2 pumpe fra Grundfoss, disse forbindes med passende slanger og fittings. For at styre sprinkleren, skal Alpha 2 pumpen kunne tændes og afbrydes. Dette gøres via et 230V/5V relæ. Dette relæ styres via én pin Enheden.

#### 5.4.1 230V/5V relæ

For at 230V/5V relæet kan blive en realitet, er det pålagt, at dette bygges i en lukket kasse og godkendes af en elektronikværksteds-ansvarlig. Kassen som bygges har et 230Vac apparatstik ind og et 230Vac stikkontakt ud. For at skabe forbindelse/afbrydelse af denne 230Vac strøm benyttes et relæ, dette relæ styrers af 5V. Når 5V tilføjes relæets spole, klikker relæet og der skabes gennemgang fra apparatstik til stikkontakt. Dette lukkede kredsløb bygges som sagt ind i en lukke kasse med gennemsigtigt låg. Relæet der benyttes er et Finder 40.52s. Figuren 5.7 viser kredsløbet for dette lukkede kredsløb.

**Figur 5.7.** 230V/5V relæ

#### 5.4.2 Alpha 2 pumpen

Alpha 2 pumpen skal blot tilsluttes 230Vac + jord. Der medfølger et special stik til selve pumpen, dette forbindes til en 230V stikprop, med 3-ledet kabel (Leder, Nul og Jord). Stikproppen kan nu sættes i 230V/5V relæets 230V stikkontakt.

**Figur 5.8.** Alpha2 Cirkulationspumpe fra Grundfoss

#### 5.4.3 Relæ styring

I følge databladet for Finder 40.52s (figur 5.9), kræver relæet 100mA ved 5V forsyning, som databladet

**DC coil data - 0.5 W sensitive (types 40.31/51/52/61)**

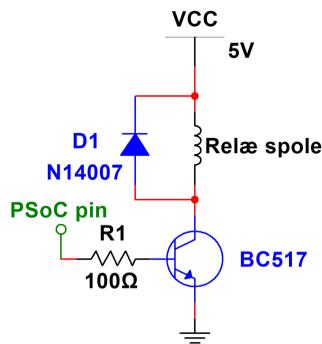
Nominal voltage $U_N$ V	Coil code 7.005	Operating range		Resistance R $\Omega$	Rated coil consumption I at $U_N$ mA
5		$U_{min}^*$ V	$U_{max}^{**}$ V	50	100

**Figur 5.9.** Datablad for Finder 40.52s - 100mA

PSoC'ens udgang giver hverken strøm eller spænding nok til at trække relæet. PSoC'en afgiver 3,3V og relæet kræver 5V. For at opnå dette benyttes en transistor til styringen af relæet. Relæet kræver 5V og 100mA, derfor skal transistoren opfylde dette krav, det gør BC517 den kan leverer 500 mA.

R1(I<sub>base</sub>) modstanden er indsat for at begrænse den strøm der går til transistorens base ben, ifølge databladet for transistoren må I<sub>base</sub> strømmen max være 100 mA. Modstanden er beregnet udfra ohmslov og der er valgt en modstand på 100 ohm, der begrænser strømmen til 33 mA. Dioden er indsat for at beskytte transistoren.

$$I_{base} = \frac{3,3V}{100\Omega} = 0,033A = 33mA \quad (5.5)$$



**Figur 5.10.** BC517 opsætning

Når PSoC'ens pin til BC517 transistoren går høj (3,3V), så skabes der forbindelse mellem collector og emittier på transistoren, herved er der 5V over relæets spole og relæets kontaktsæt klikker herved.

#### 5.4.4 Driver

Driveren der skal håndtere Sprinkler-pumpe systemet, skal kunne aktivere/deaktivere det forudbestemte sprinklerrelæ. Metoden modtager en adresse parameter samt en on/off parameter. Herefter sætter metoden en af den forudbestemte pin på Enheden høj/lav (3,3V/0V). Herefter søger relæstyringen og 230V/5V relæet for hhv. tænde/slukke for sprinkleren.

#### Pseudokode

```

1 void controlSprinkler(int address, bool on_off){
2     Manage Sprinkler address
3     Activate/deactivate corresponding Sprinkler
4 }
```

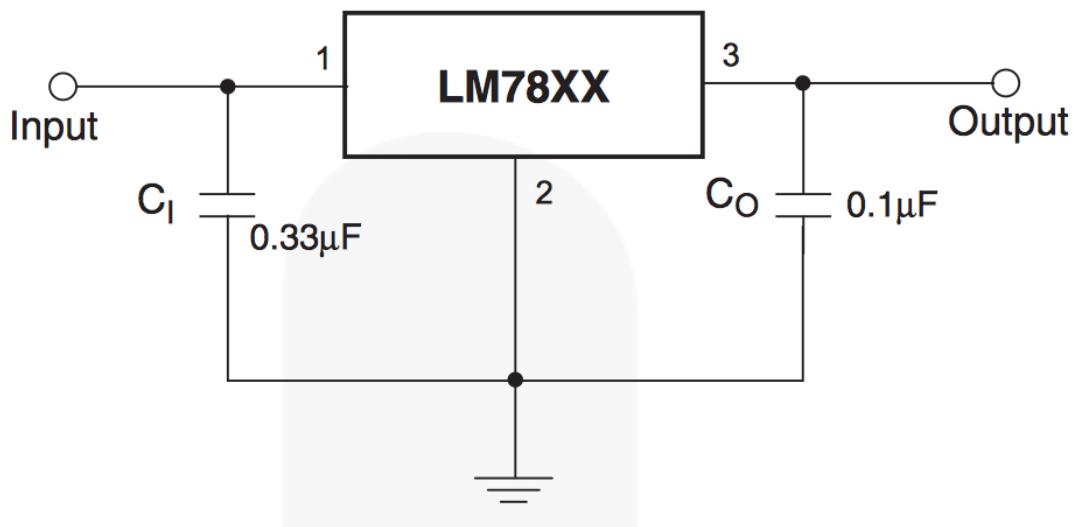
## 5.5 Strømforsyning og tilslutnings-print

### 5.5.1 Strømforsyning

For at frigøre sig fra laboratoriets strømforsyninger udarbejdes egen strømforsyning. Vha. en 230Vac/12Vdc transformerer og egnede spændings-regulatorer designes en strømforsyning til netop dette projekt. Masteren (DevKit8000) har sin egen strømforsyning og denne ændres ikke. Enheden (PSoC4) skal forsynes med 5V dc via USB. Relæstyringen skal forsynes med 5V dc. PIR sensoren skal forsynes med 5-20 V dc. Temperatur/fugt sensoren skal forsynes med 3,3V dc.

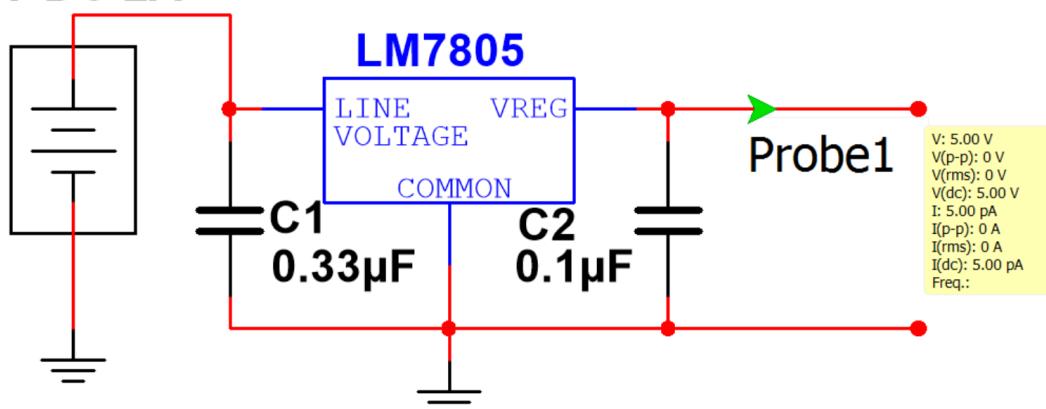
230Vac/12Vdc transformeren er fra en gammel computer. Denne kan give 2 amperre og er forsynet med et DC stik (12V) og en stikprop (230Vac). Denne transformator bruges som den er.

For at regulere fra de 12 V til 5 V benyttes en spændingsregulator. LM7805 er beregnet netop til dette formål. Ud fra databladet 5.11 ses forbindelsen af LM7805'ern. Databladet foreskriver også at regulatoren kan givet et output på max 1A ved korrekt køling.



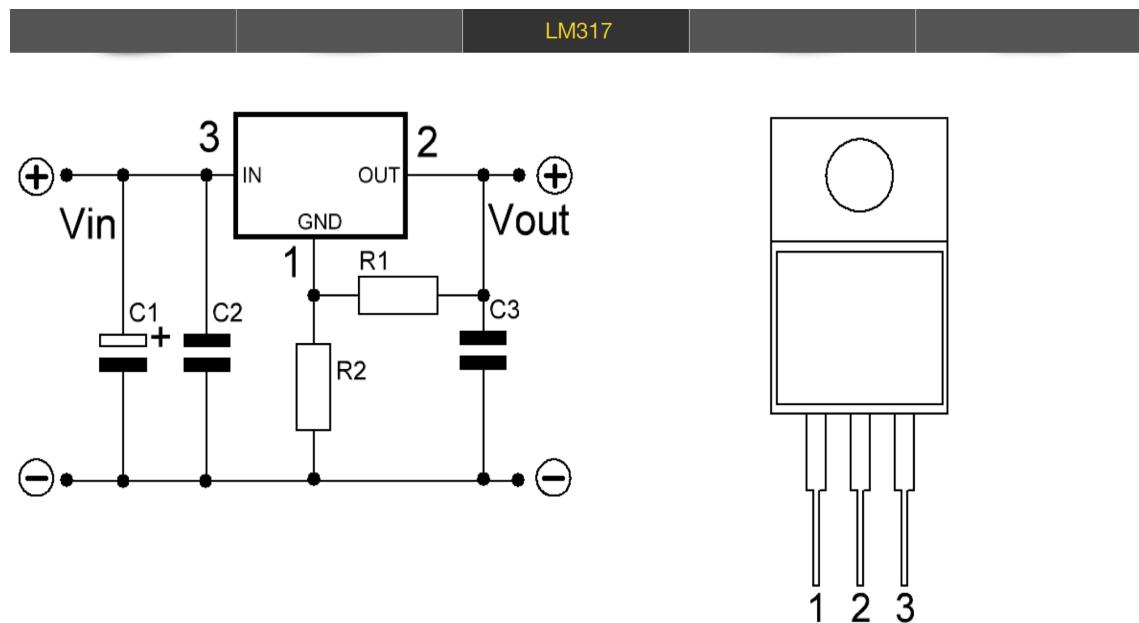
**Figur 5.11.** Forbindelse af LM7805 - Side 18/24 LM7805.pdf

Spændingsregulatoren LM7805 er opbygget i multilsim, sammen med 12V dc forsyningen. Kredløbet er simuleret og dette viser at outputtet er de ønskede 5 V.

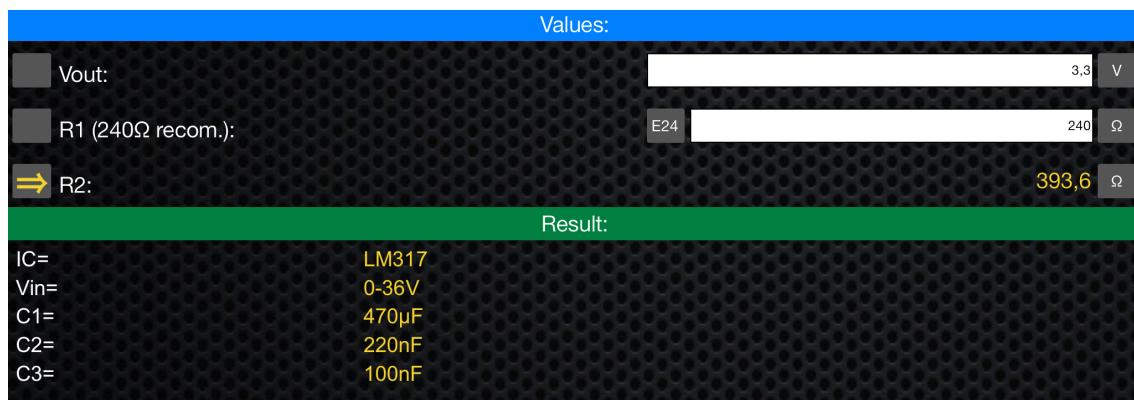
**12V DC 2A***Figur 5.12.* LM7805 Simulering

For at opnå 3,3V benyttes en justerbar spændingsregulator LM317. Denne regulator kan levere 1,2V-33V 3A. Det er fugt,-temperatursensoren der skal forsynes med 3,3V/180uA, det er LM317'erns opgave.

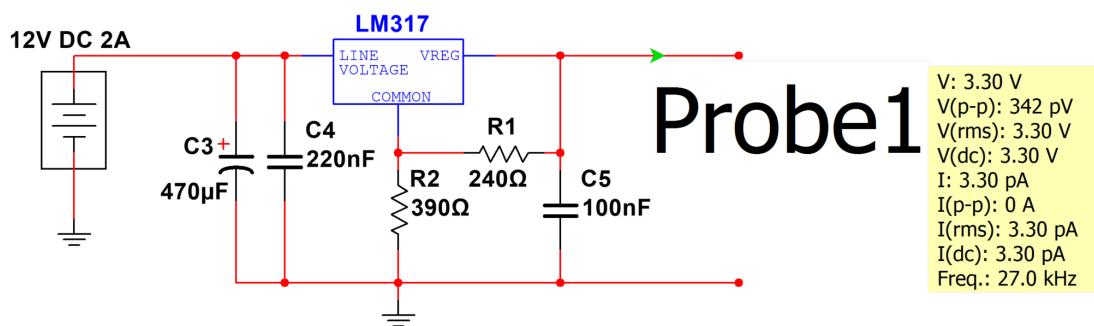
LM317 er slæet op iPad appen "Electronic Toolbox", vha. denne gives hvordan opsætningen af denne skal være for at opnå 3,3V

*Figur 5.13.* Kredsløb for LM137

Figur 5.14 viser hvorledes appen beregner en værdi for modstanden  $R_2$ , når  $V_{out}$  er sat til 3,3V og modstanden  $R_1$  anbefales til 240 ohm. I følge appens beregning skal  $R_2$  være 393,6 ohm.

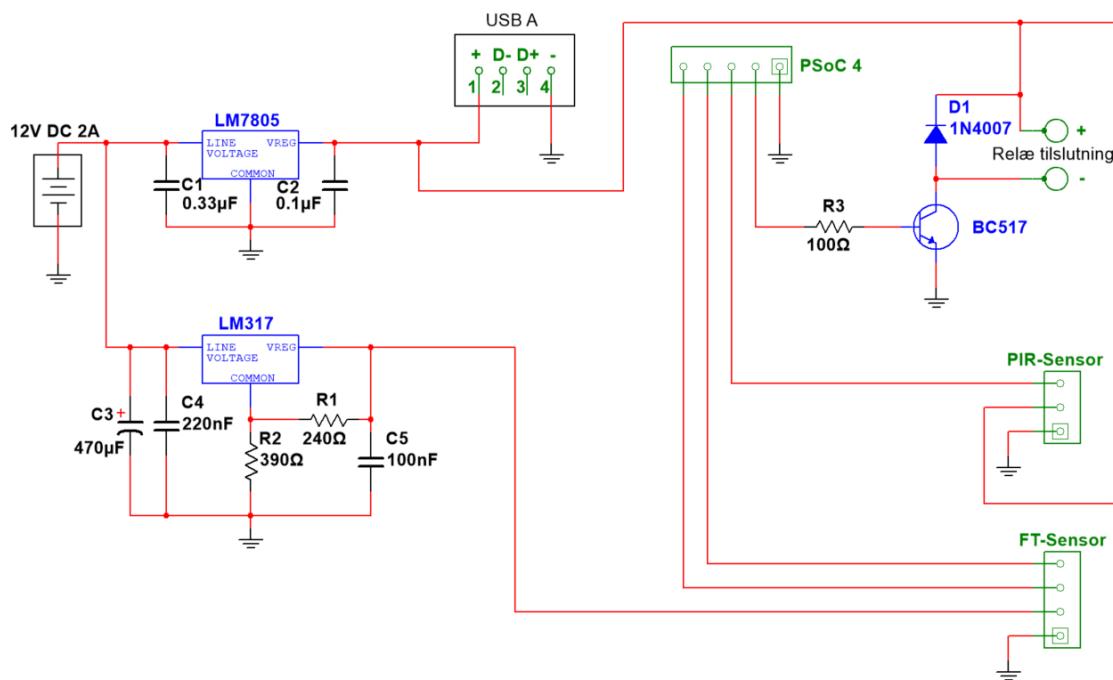
*Figur 5.14.* Kredsløb for LM137

Figur 5.15 viser simuleringen af LM317 spændingsregulatoren. Outputtet er 3,3V som forventet. Der simuleres med en modstand R2 på 390 ohm og ikke 393,6 ohm.

*Figur 5.15.* Simulering i Multisim ud fra Electronic Toolbox appens værdier

### 5.5.2 Tilslutningsprint

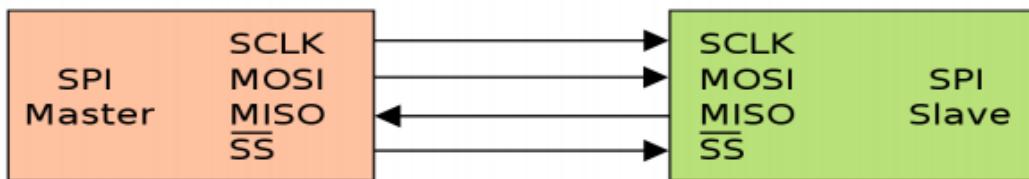
Tilslutningsprintet, se figur 5.16, sørger for at sammenkoble Enheden med de ønskede sensorer, samt at forsyne Enhed og sensorer med ønsket forsyning.



**Figur 5.16.** Kredsløb over samlet spændingsforsyning (3,3V og 5V) og tilslutninger til PSoC4

## 5.6 SPI

Serial Parallel Interface (SPI) er en måde at lave hurtig seriell dataudveksling på. SPI er udviklet af Motorola og fungerer ved at man har, som regel, en enkelt master enhed der styrer flere slave enheder. Ved SPI er der ingen fejl-check men adressering af flere ender kan dog være HW krævende. I EasyWater8000 projektet er der SPI kommunikation imellem Master(devkit8000) og Enhed(PSoC), det giver muligheden for at overføre flere data på samme tid imellem disse to.

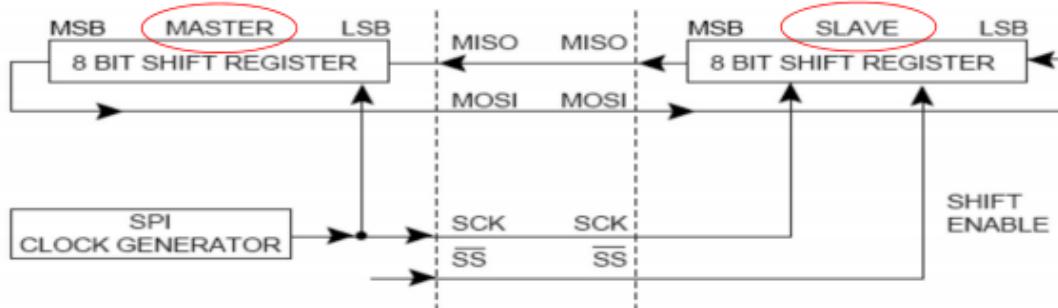


**Figur 5.17.** SPI

På figur 5.17 ses en typisk opkobling imellem Master og Slave, det kræver 4 tråde.

- SS står for "Slave select".
- MISO står for "Master in slave out".
- MOSI står for "Master out slave in".
- SCLK står for "Serial clock"

SPI kommunikation er baseret på skifregister princippet, som ses på figur 5.18. Der vælges hvilken slave der ønskes at skrives til ved slave select(SS), derefter shiftes et 8-bit register 1 bit ad gangen. Serial clock er den clock der sørger for at shiftningen af bits sker korrekt, clockfrekvensen må ikke overstige grænsen for hvad enhederne kan håndtere.



*Figur 5.18.* SPI register

En oftes anvendt kommunikation mellem master og slave foregår ved at master sætter SS til 0. Den fortæller nu til slaven at den er klar til dataoverførelse. Nu sender masteren en bit over til slaven og påbegynder en full duplex data transmission, samtidigt sender slaven en bit over til masteren. Denne process fortsætter indtil masteren har sendt alle sine bits, derefter stopper masteren med at toggle på clocken og slave enheder bliver frigivet.

### 5.6.1 Driver

#### Pseudokode

```

1
2 }
```

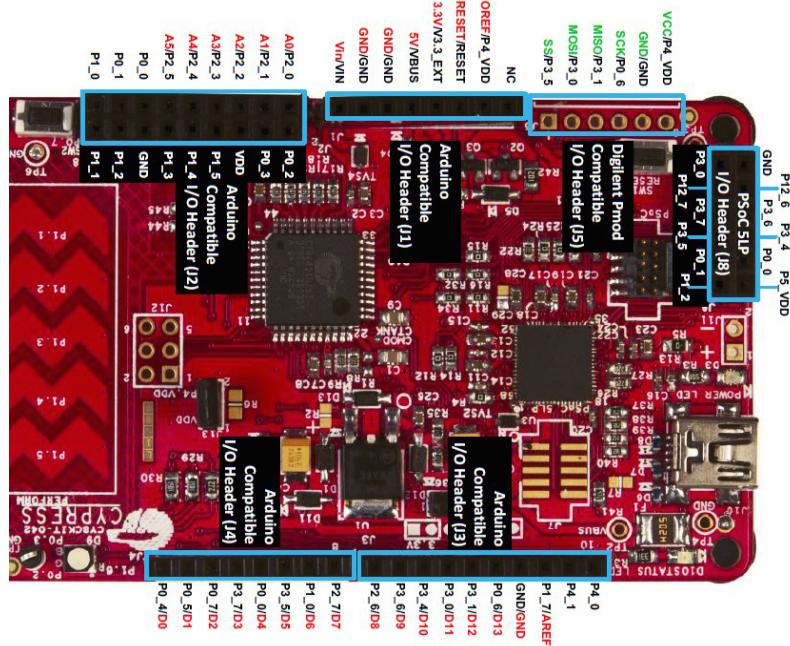
## 5.7 Hardware forbindelser

### 5.7.1 Enhed (HW)

Her beskrives hvor mange og hvilke pins der skal benyttes på PSoC4 til de forskellige forbindelser.

**Tabel 5.1.** Tabel der viser pins på PSoC4

Forbindelse	Antal pins	Signalnavne	Pinnavne	Kommentar
SPI (P_DK)	4	MOSI	P3[0]	
		MISO	P3[1]	
		SCK	P0[6]	
		SS0	P0[7]	
		GND	J3(GND)	
PIR-TTL (P_PIR)	1	P_PIR	P1[1]	
Vandpumpe (VP_01)	4	SPRINKLER_1	P2[0]	
		SPRINKLER_2	P2[1]	
		SPRINKLER_3	P0[2]	
		SPRINKLER_4	P0[3]	
FT-sensor (P_FT)	5	SELECT	P0[0]	Fugt/Temp.
		ANALOG	P2[5]	Analog niveau
		BIT0(LSB)	P2[2]	Adressering
		BIT1	P2[3]	Adressering
		BIT2(MSB)	P2[4]	Adressering
Fælles GND	1	GND(Forb. print)	J2(GND)	Forb. print
Enhedsadr.	4	BIT0(LSB)	P1[2]	
		BIT1	P1[3]	
		BIT2	P1[4]	
		BIT3(MSB)	P1[5]	

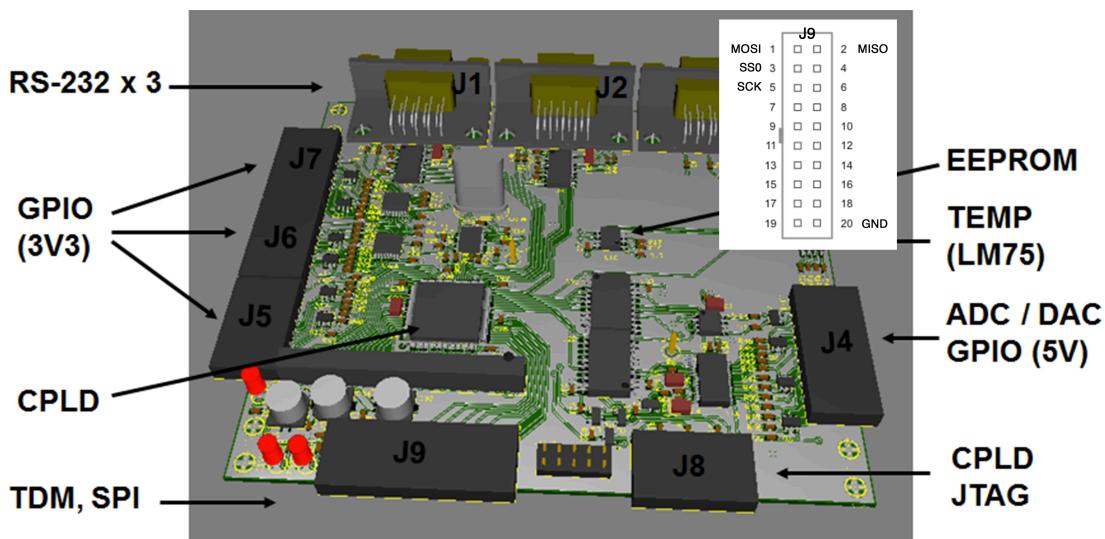


### 5.7.2 Master (HW)

Her beskrives hvor mange og hvilke pins der skal benyttes på Devkit8000 til de forskellige forbindelser.

**Tabel 5.2.** Tabel der viser pins på Devkit8000

Forbindelse	Antal pins	Signalnavne	Pinnavne	Kommentar
SPI (P_DK)	5	MOSI	J9.1	
		MISO	J9.2	
		SCK	J9.5	
		SS0	J9.3	
		GND	J9.20	



**Figur 5.20.** Model af tilføjelses-print og J9-stik



# Softwaredesign 6

---

## 6.1 Dataprotokol (BS)

Når data flyttes mellem Master og Enhed, ifm. logning, anvendes følgende dataprotokol i mellem *Application*-lagene.

Den data som skal flyttes er følgende:

1. Temperatur
2. Fugtighed
3. Bevægelsesregistrering
4. Påbegyndt vanding

Systemet ved ikke på forhånd hvor meget data der skal flyttes. Derfor deles det op i tre typer. Data fra sensorene, besked om bevægelse og fejlregistreringer. Fælles for de tre er at tidsstemplet altid sendes med.

Fra standardbiblioteket for C++ vælges datatypen `vector` som container. Dette er en dynamisk array-struktur som automatisk udvides og mindskes efter behov. Ved at anvende `vectoren` med `string` som undertype er det nemt at identificerer indholdene og formaterer dem som nødvendigt.

Som nævnt er der flere muligheder for ”datapakker”. I tabellerne 6.1, 6.2 og 6.3 er deres opbygninger vist.

Første streng der læses afgør hvor mange af de efterfølgende strenge som høre her til. Hvis det første der modtages er ”D”, betyder det at der kommer information fra sensorene og der skal læses fire efterfølgende strenge. Hvis der modtages et ”B” betyder det at der er registreret bevægelse, og der skal kun læses en dato efterfølgende. Hvis der modtages et ”E” er der en fejl, og fejlkoden fra Enheden sendes.

**Tabel 6.1.** Dataformatering ifm. sensordata

Type	Temperatur	Fugtighed	Vanding
”D”	”TTT.T”	”FFF”	”Tilstand”

**Tabel 6.2.** Dataformatering ifm. bevægelse

Type
”B”

Et eksempel på strukturen af en datahentning er vist i tabel 6.4. Her kan man se at der siden sidste hentning har været bevægelse på hullet, der er hentet sensordata med værdierne 14.5 gader og 20% fugtighed. Der har også været en fejl 13.

**Tabel 6.3.** Dataformatering ifm. fejl

Type	Fejlkode
"E"	"XXXX"

**Tabel 6.4.** Dataformatering ifm. log-information

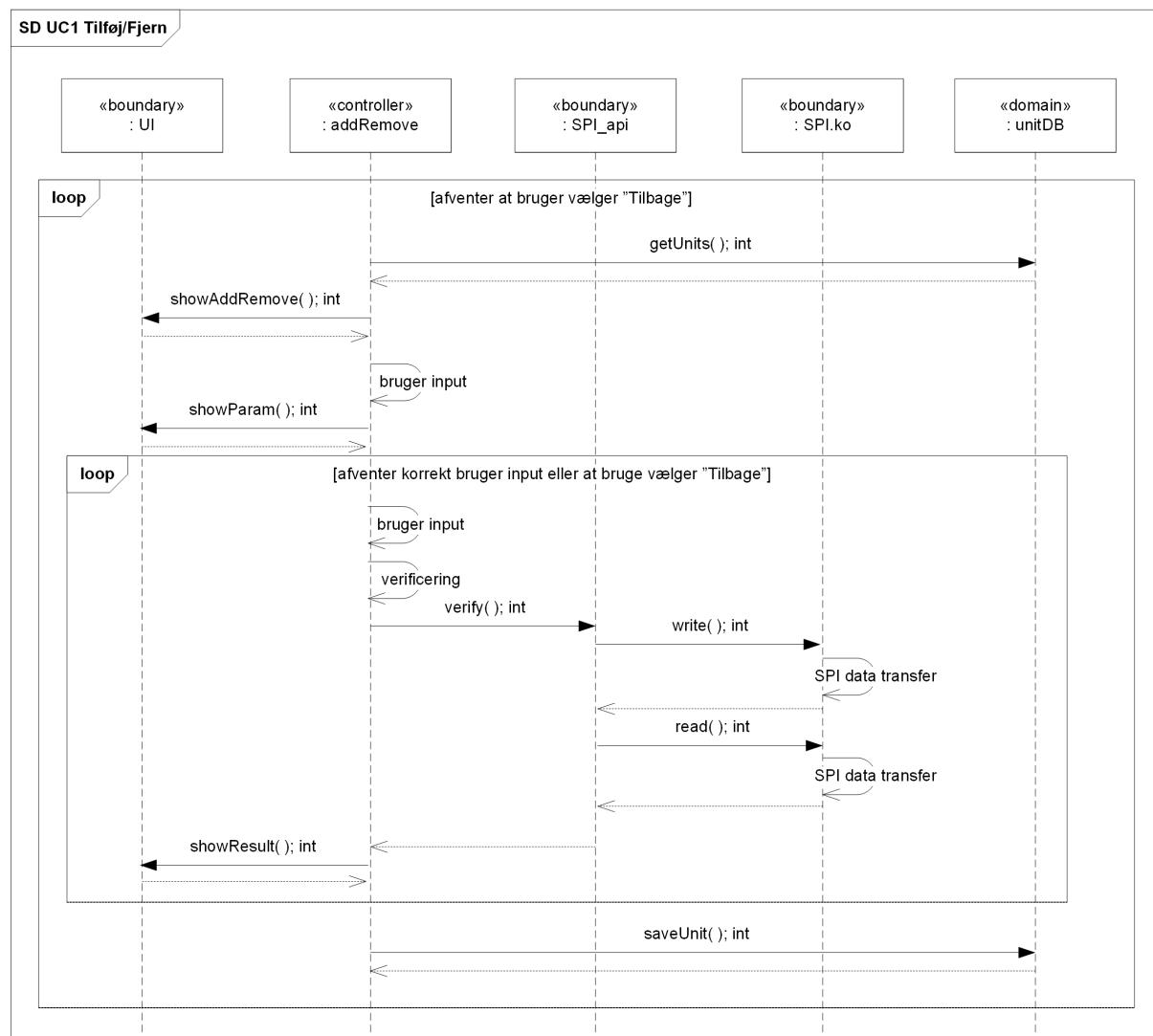
vector<string>[0]	[1]	[2]	[3]	[4]	[5]	[6]
"B"	"D"	"014.5"	"020"	"Slukket"	"E"	"013"

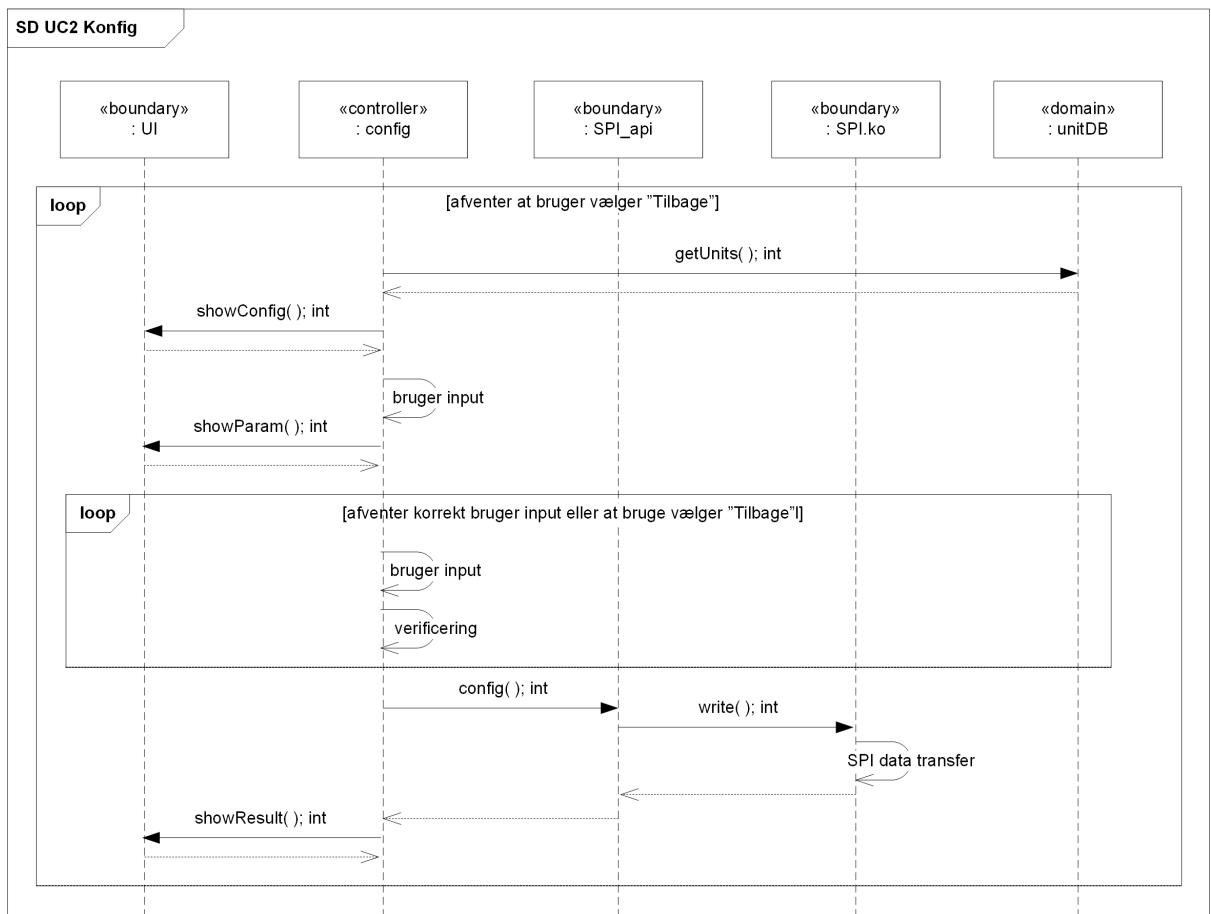
## 6.2 Applikationsmodeller (JC BS)

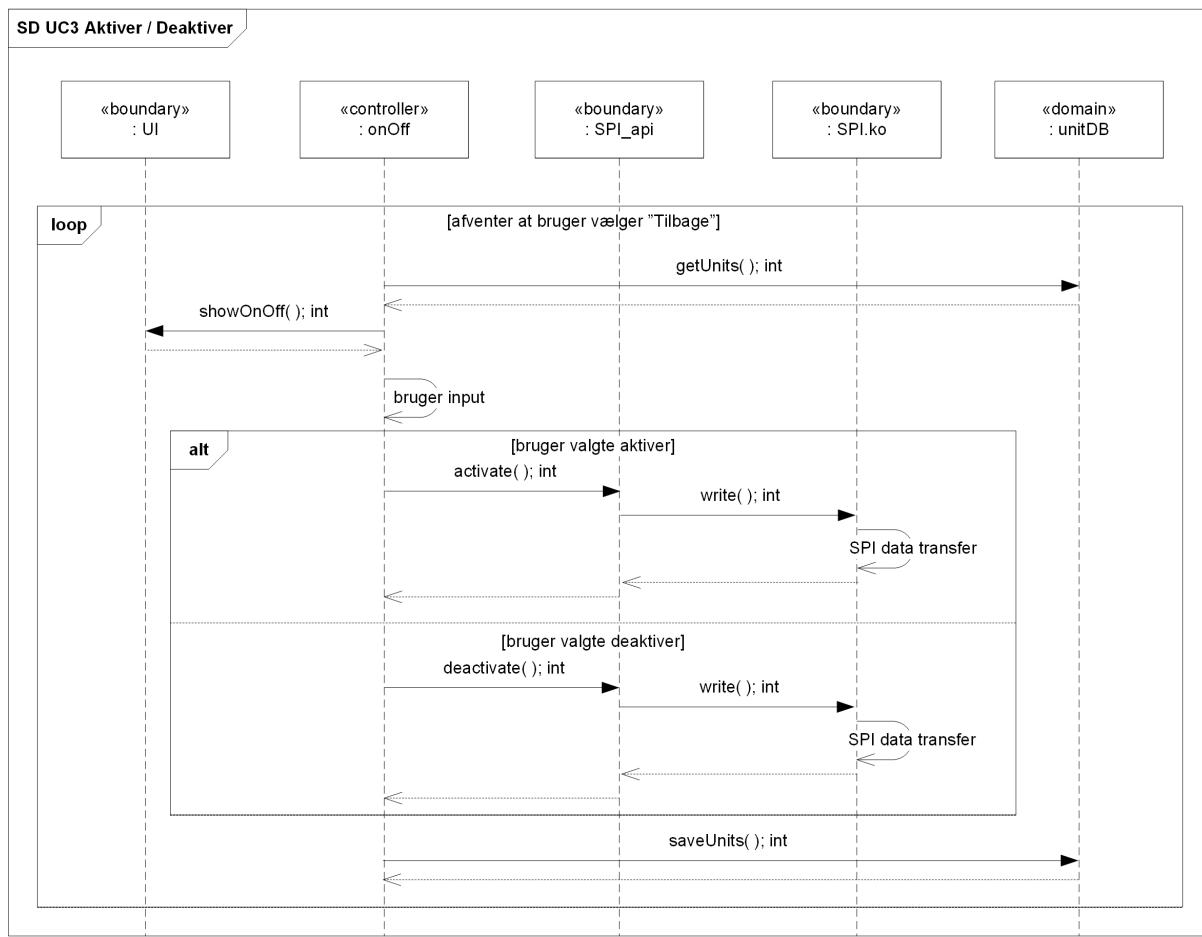
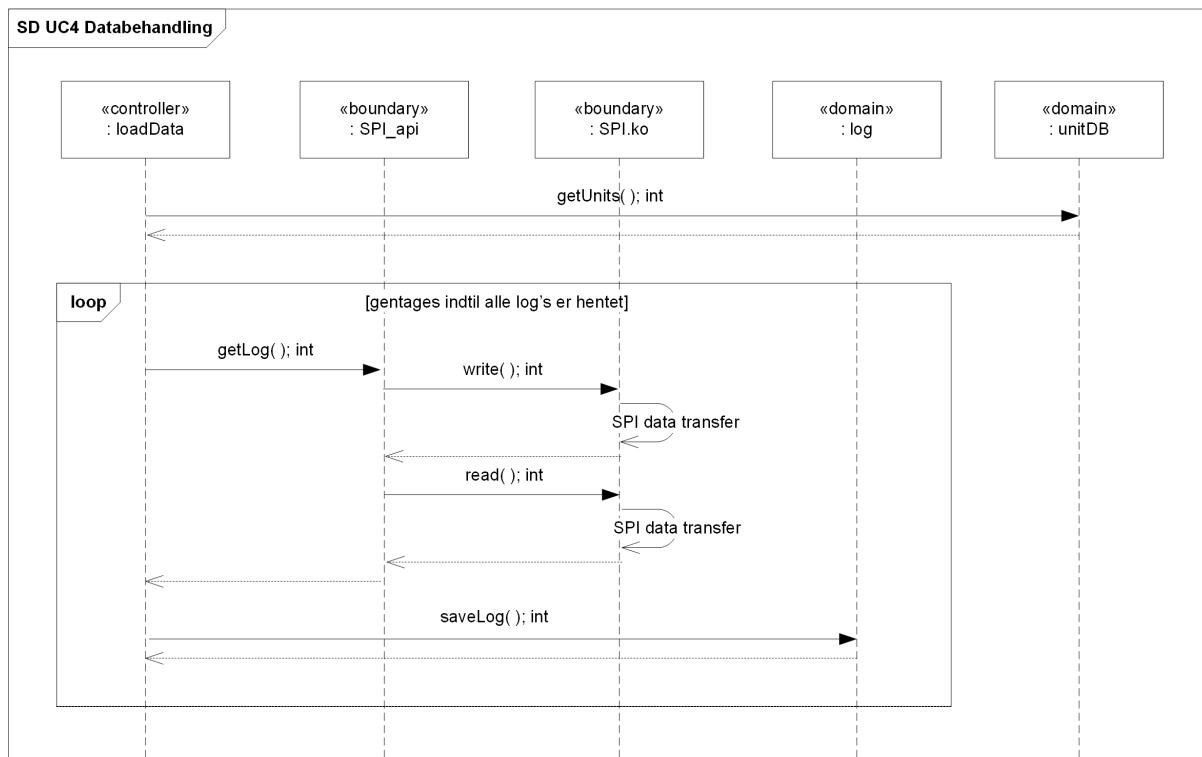
Selve designet af softwaren bygger på de følgende applikationsmodeller. Her laves der sekvens- og klassediagrammer over hver del af systemet samt klassebeskrivelser hvor funktionen for de enkelte metoder beskrives.

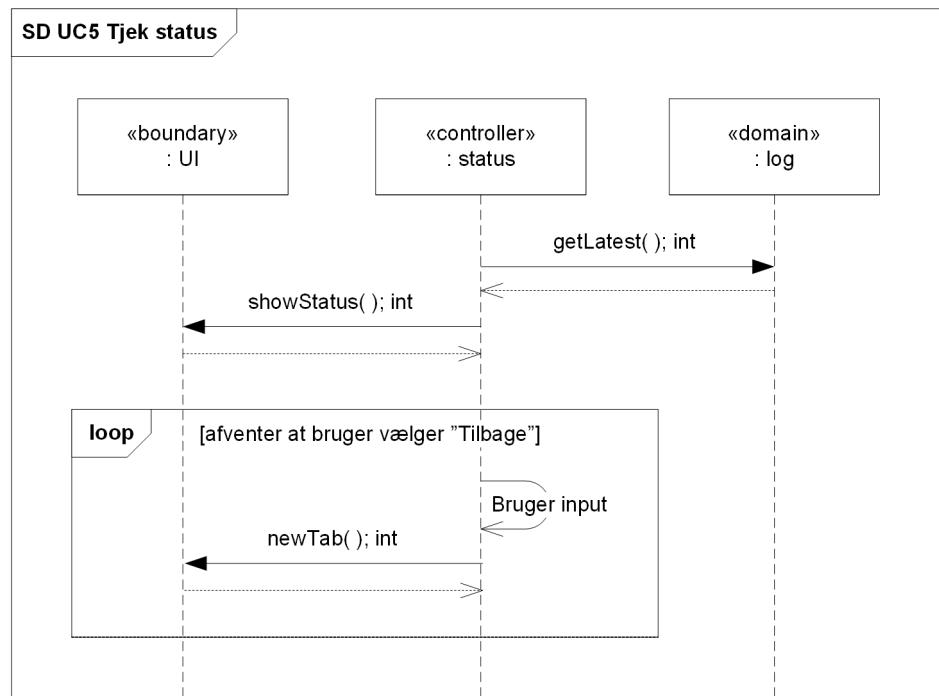
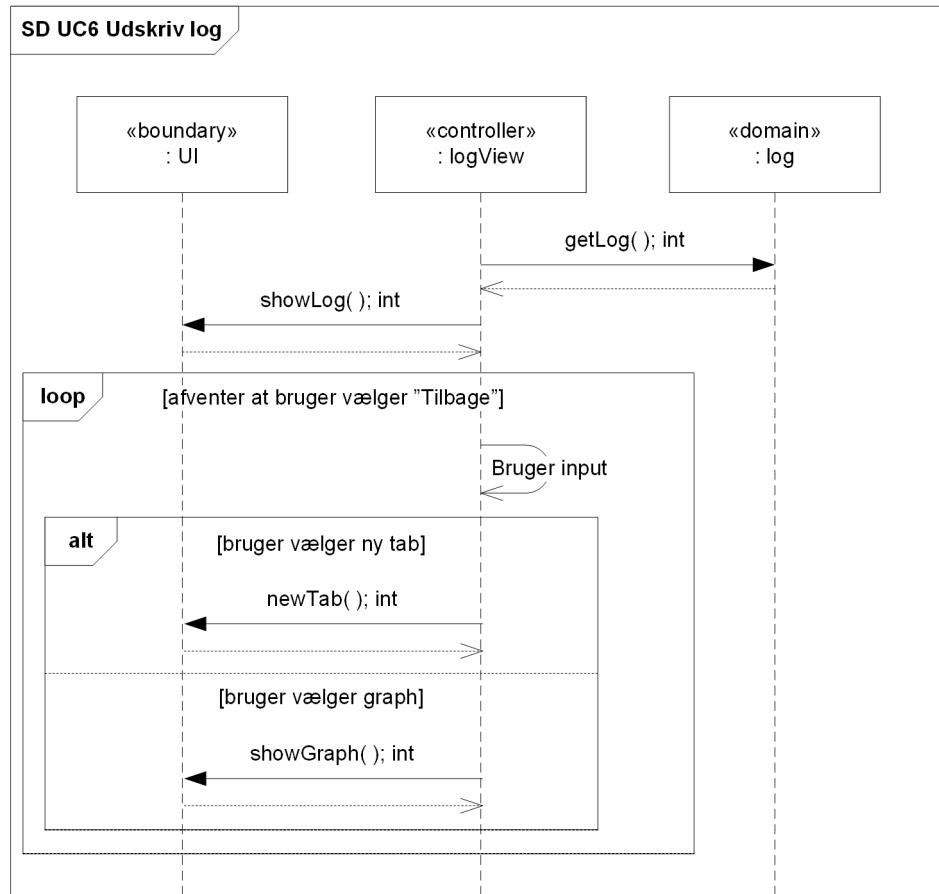
### 6.2.1 Master

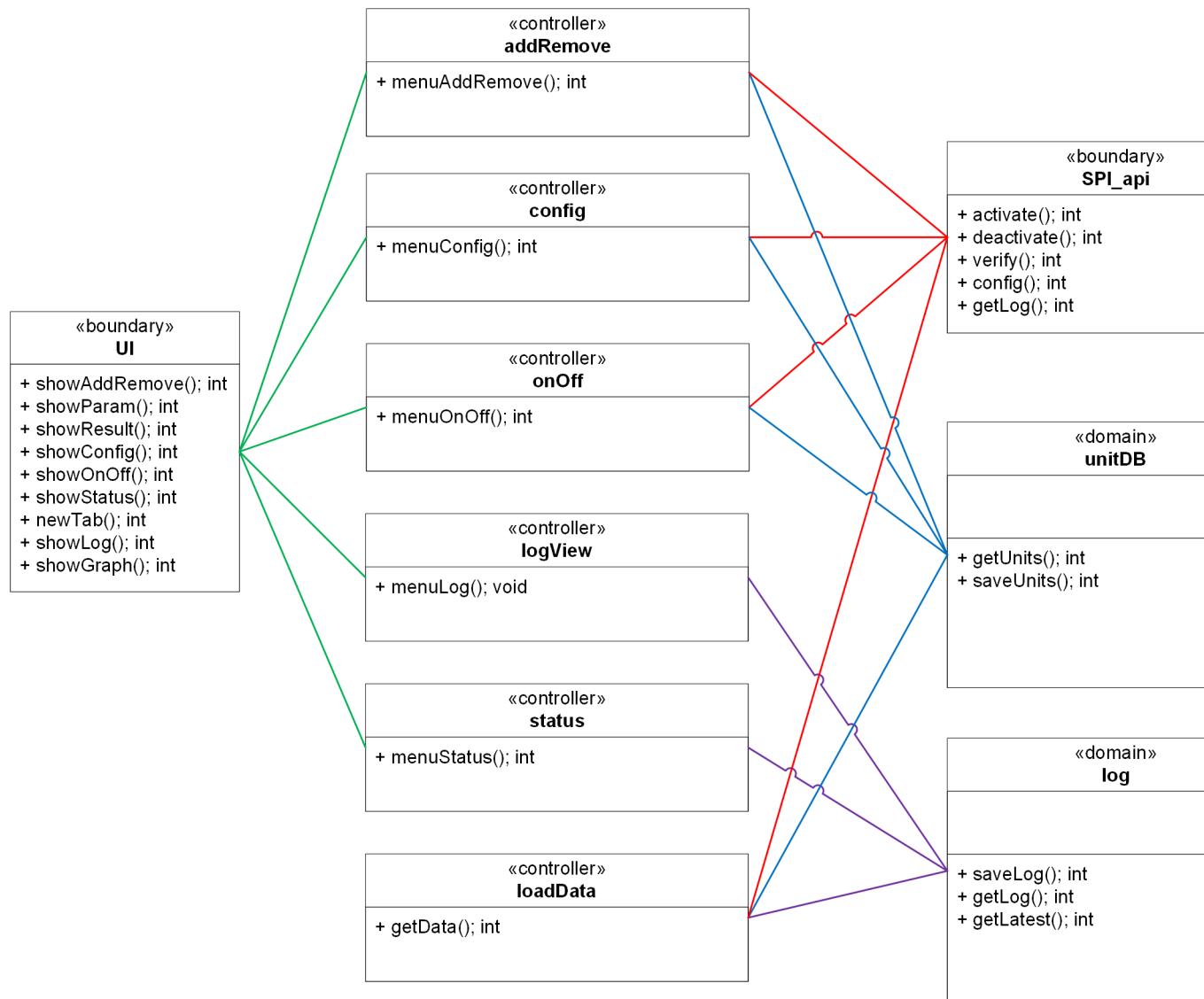
Applikationsmodeller for Master.

**Figur 6.1.** Sekvensdiagram UC1

*Figur 6.2.* Sekvensdiagram UC2

*Figur 6.3.* Sekvensdiagram UC3*Figur 6.4.* Sekvensdiagram UC4

*Figur 6.5.* Sekvensdiagram UC5*Figur 6.6.* Sekvensdiagram UC6



Figur 6.7. klassediagram devkit8000

Efter udarbejdelsen af sekvensdiagrammer samles alle metode kald imellem kasserne til et klassediagram som ses ovenfor på 6.7. Efter dette klassediagram udarbejdes en klassebeskrivelse hvor der tænkes over hvilke attributer de forskellige metoder skal have for at kunne udføre deres ansvar. Det bliver så fuldt op med et endeligt statisk klassediagram som inkludere alle attributer og medlems data.

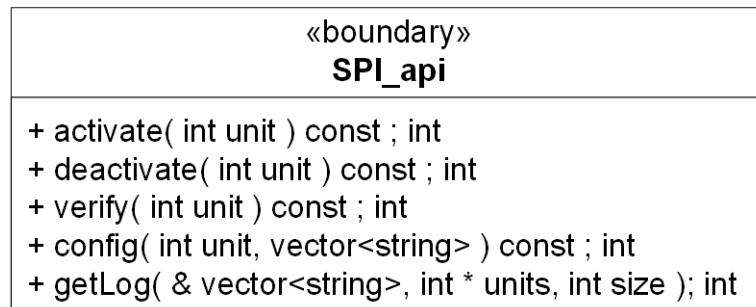
### 6.2.2 Enhed

Applikationsmodeller for Enhed.

## 6.3 Klassebeskrivelser

Her følger klassebeskrivelser for de udledte klasser fra applikationsmodellerne.

### 6.3.1 Master



*Figur 6.8.* klassediagram SPI API

### SPI\_api

**Ansvar:** At være et lag imellem *applications*-laget og *device driver*-laget ifm. SPI kommunikation.

int activate( int unit ) const

**Parametre:** modtager en integer på enhed som skal aktiveres

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** metoden skal aktivere enheden **unit** over SPI netværket.

int deactivate( int unit ) const

**Parametre:** modtager en integer på følgende enhed som skal deaktiveres

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** metoden skal deaktivere enheden **unit** over SPI netværket.

int verify( int unit ) const

**Parametre:** modtager en integer på følgende enhed som skal verificeres

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** metoden skal verificere om enheden **unit** er tilkoblet SPI netværket.

int config( int unit, vector<string> ) const

**Parametre:** modtager en integer på følgende enhed som skal konfigureres. Derudover modtager den en vector af typen string som indeholder parametrene enheden skal konfigureres med

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** metoden skal sende konfigurations-parametrene i **vector** til enheden **unit**

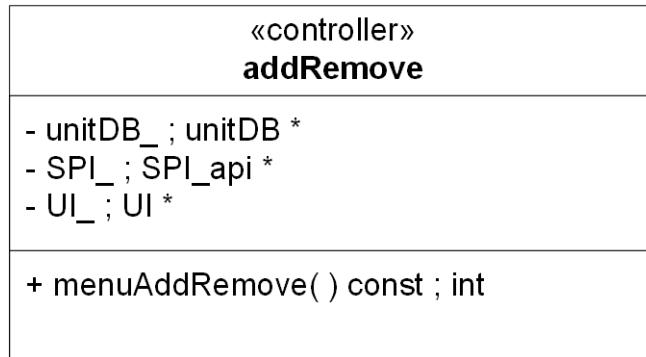
over SPI netværket.

```
int getLog( & vector<string> , int * units, int size )
```

**Parametre:** modtager en reference til en vector af typen string som loggen skal gemmes i. Pointer til array af enheder som skal logges samt int med antallet af enheder i arrayet.

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** metoden skal hente log fra enheder i arrayet `units` på SPI netværket og gemme dem i `vector` i henhold til dataprotokollen.

*Figur 6.9.* klassediagram addRemove**addRemove**

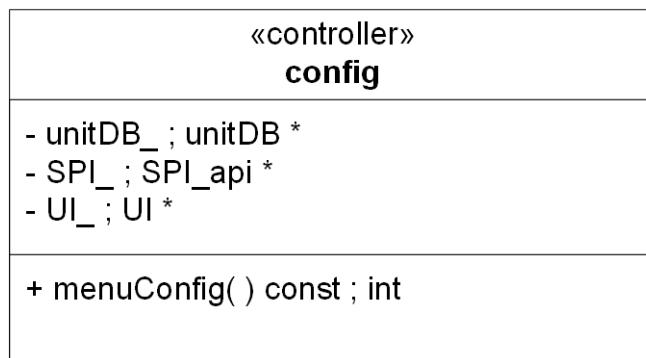
**Ansvar:** at styre forløbet i UC1: Tilføj / fjern enhed.

int menuAddRemove( ) const

**Parametre:** Modtager ingen parametre

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** Modtager information fra brugeren omkring enhed, verificere enheden over SPI netværket og gemme information i databasen..

*Figur 6.10.* klassediagram config**Config**

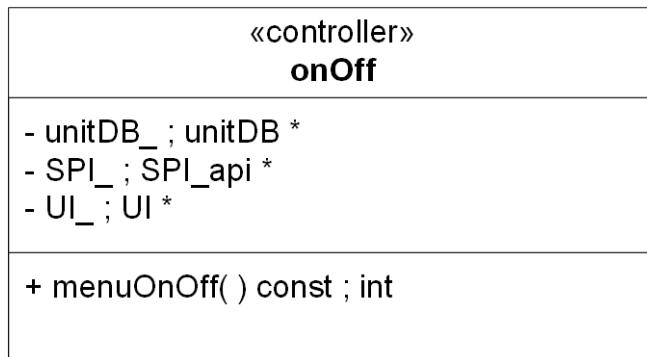
**Ansvar:** at styre forløbet i UC2: Konfig.

int menuConfig( ) const

**Parametre:** Modtager ingen parametre

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** Modtager information fra brugeren omkring enhed og parametre. Parametrene skrives til den pågældende enhed på SPI netværket.

*Figur 6.11.* klassediagram onOff**onOff**

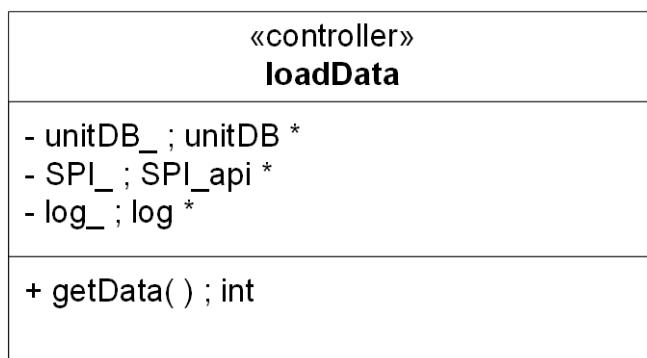
**Ansvar:** at styre forløbet i UC3: Aktiver / deaktiver.

int menuOnOff( ) const

**Parametre:** Modtager ingen parametre

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** Modtager information fra brugeren omkring hvilken enhed som ønskes aktiveret eller deaktiveret. Enheden modtager informationen over SPI netværket.

*Figur 6.12.* klassediagram loadData**loadData**

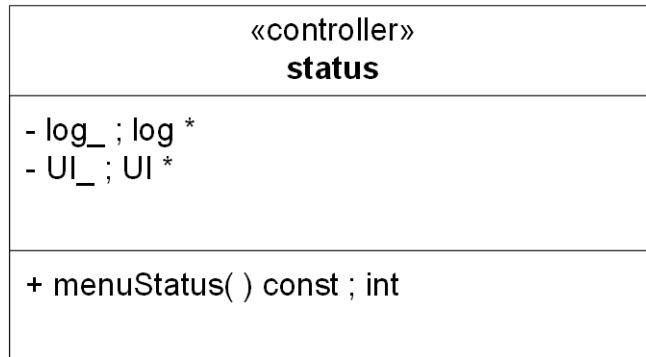
**Ansvar:** at styre forløbet i UC4: Databehandling .

int menuloadData( )

**Parametre:** Modtager ingen parametre

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** Sættes igang med et interrupt styret af en timer. Henter log over SPI netværket og gemmer den.

*Figur 6.13.* klassediagram status**status**

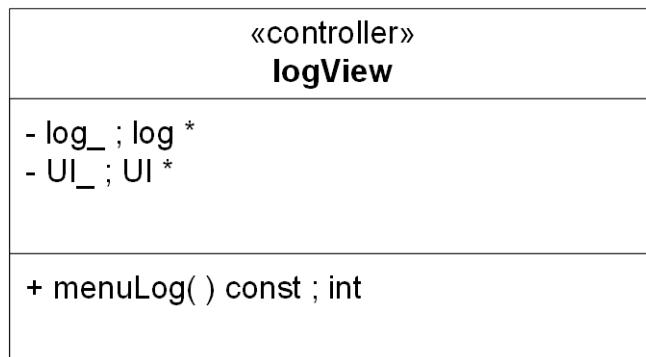
**Ansvar:** at styre forløbet i UC5: Tjek status.

int menuStatus( ) const

**Parametre:** Modtager ingen parametre

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** Henter den seneste log i hukommelsen og viser brugeren den, for den ønskede enhed.

*Figur 6.14.* klassediagram logView**logView**

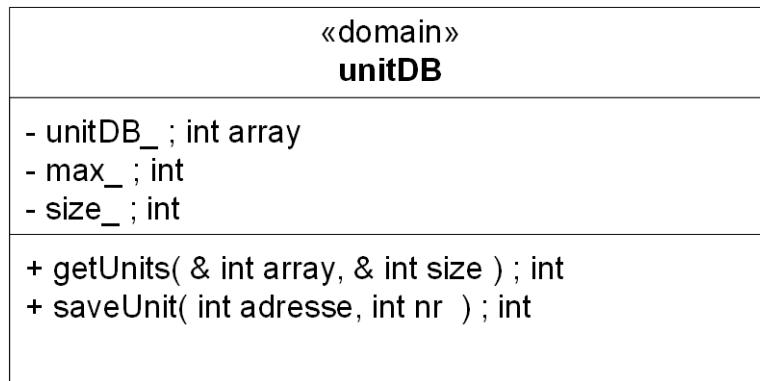
**Ansvar:** at styre forløbet i UC6: Udskriv log.

int menuLog( ) const

**Parametre:** Modtager ingen parametre

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** Henter hele loggen i hukommelsen og viser brugeren den, for den ønskede enhed.

*Figur 6.15.* klassediagram unitDB**unitDB**

**Ansvar:** at styre forløbet i UC1: Tilføj / fjern enhed.

int getUnits( & int array, & int size )

**Parametre:**

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

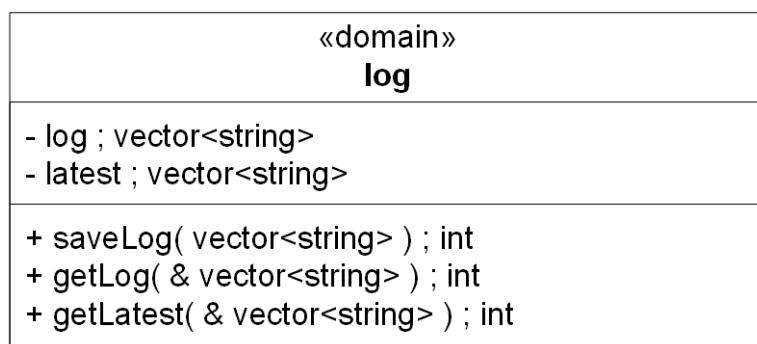
**Beskrivelse:**

int saveUnit( int adresse, int nr )

**Parametre:**

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** .

*Figur 6.16.* klassediagram log**log**

**Ansvar:** Gemme information loggen til senere brug.

int saveLog( vector<string> )

**Parametre:** Modtager en vector af typen string.

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** Modtager log fra enhed og skriver den over i den samlede log og gemmer den i latest

```
int getLog( & vector<string> )
```

**Parametre:** Modtager en adresse til en vector af typen string.

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** .

```
int getLatest( & vector<string> )
```

**Parametre:** Modtager en adresse til en vector af typen string.

**Returværdi:** 0 ved succes ellers negativ i overenstemmelse med fejl-listen

**Beskrivelse:** .



# **Modultest 7**

---



# Accepttestspezifikation

8

Punkterne i accepttestspezifikationen er skrevet ud fra punkterne i hovedforløbet for de enkelte usecases beskrevet i kapitel 2.3.

UC1: Tilføj/fjern enhed				
	Test	Forventet Resultat	Resultat	Godkendt/Kommentar
1	Bruger vælger ”Tilføj/fjern enhed” i hovedmenu	Visuel: Menu fremkommer på skærmen	N/A	N/A
2	Bruger vælger "Tilføj enhed"	Visuel: Menu for "Tilføj enhed" fremkommer på masters skærm	N/A	N/A
3	Der udføres visuelt en test for om listen med opsatte enheder fremkommer	Visuel: Listen vises for bruger med opsatte enheder	N/A	N/A
3.a	Der testes visuelt om indtastningen er mulig	Felt for indtastningen er til rådighed for bruger	N/A	N/A
3.b	Bruger indtaster navn og adresse for Enhed	Visuel: Den indtastede information vises på skærmen	N/A	N/A
3b.a	Bruger indtaster navn: <tom> og adresse: 0xFF	Master giver fejlmeddeelse omkring ugyldig indtastning	N/A	N/A
3.c	Masterenhed tilfører Enhed til systemet med default parametre	Visuel: Enheden tilføjes med de indtastede informationer	N/A	N/A
3.d	Enheden forbindes til kommunikationsnetværket	Testes i 3.e	N/A	N/A
3.e	Master verificere forbindelse til Enhed og sender dato og tidspunkt	Visuel: Master viser godkendt Enhed	N/A	N/A

...fortsat fra forrige side

	<b>Test</b>	<b>Forventet Resultat</b>	<b>Resultat</b>	<b>Godkendt/ Kommentar</b>
<b>3e.a</b>	Forbindelsen mellem enhed og master afbrydes	Visuel: Master giver besked om mistet kommunikation	N/A	N/A
<b>4</b>	Bruger vælger "Fjern enhed"	Visuel: Menu for "Fjern enhed" fremkommer på skærmen	N/A	N/A
<b>5</b>	Der udføres visuelt en test for, om listen med opsatte enheder fremkommer	Listen præsenteres på skærmen	N/A	N/A
<b>5.a</b>	Bruger indtaster adresse på Enhed der ønskes fjernet	Visuel: Bekræftelse vises på skærmen	N/A	N/A
<b>5.b</b>	Master deaktivere Enhed	Visuel: Adressen accepteres og Enhed deaktivieres	N/A	N/A
<b>5.c</b>	Der testes visuelt at master giver information omkring at Enhed er slettet fra systemet	Al information om enhed slettes fra systemet	N/A	N/A
<b>6</b>	Der testes visuelt at enheden er fjernet fra listen over opsatte enheder	Den opdateret liste præsenteres på skærmen	N/A	N/A
<b>7</b>	Brugeren vælger at returnere til hovedmenuen	Visuel: Master returnerer til hovedmenuen	N/A	N/A

#### UC2: Konfig

	<b>Test</b>	<b>Forventet Resultat</b>	<b>Resultat</b>	<b>Godkendt/ Kommentar</b>
<b>1</b>	Bruger vælger "Konfig" i hovedmenuen	Visuel: Liste over opsatte enheder fremkommer på skærmen	N/A	N/A
<b>2</b>	Bruger vælger en Enhed der ønskes omkonfigureret	Visuel: Menu for konfiguration for valgt Enhed vises på skærmen	N/A	N/A
<b>2.a</b>	Bruger vælger at afbryde konfigurationen	Visuel: Master forlader menuen og går tilbage til hovedmenuen	N/A	N/A

...fortsat fra forrige side

	<b>Test</b>	<b>Forventet Resultat</b>	<b>Resultat</b>	<b>Godkendt/ Kommentar</b>
<b>3</b>	Bruger indtaster nye værdier for den valgte enhed	Indstillingsmulighederne er tilgængelige	N/A	N/A
<b>3.a</b>	Der indtastes temperatur: -273 og fugt: -1	Visuel: Fejlmeddeelse vises på skærmen	N/A	N/A
<b>4</b>	Bruger gemmer de ønskede indstillinger gemmes	Visuel: Indstillingerne for den valgte enhed gemmes	N/A	N/A
<b>4.a</b>	Bruger vælger at afbryde konfigurationen	Visuel: Indstillingerne slettes og master går tilbage til hovedmenu. Master gemmer ikke indstillingerne	N/A	N/A
<b>5</b>	Der testes at den valgte enhed reagerer på de nye indstillinger	Enheden opererer efter de nye parameter	N/A	N/A

#### UC3: Aktiver/Deaktiver

	<b>Test</b>	<b>Forventet Resultat</b>	<b>Resultat</b>	<b>Godkendt/ Kommentar</b>
<b>1</b>	Bruger vælger ”Aktiver/Deaktiver” i hovedmenuen	Visuel: Menu for ”Aktiver/Deaktiver” fremkommer på skærmen	N/A	N/A
<b>2</b>	Bruger markere en opsat Enhed i menuen	Visuel: Den valgte bliver markeret	N/A	N/A
<b>3</b>	Bruger vælger ”Aktiver” for den valgte enhed	Visuel: Enhed indikere at den er aktiv  Der analyseres på SPI kommunikationen at Enhed modtager Aktivering	N/A	N/A
<b>3</b>	Bruger vælger ”Deaktiver” for den valgte enhed	Visuel: Enhed indikere at den er deaktiv  Der analyseres på SPI kommunikationen at Enhed modtager Deaktivering	N/A	N/A

...fortsat fra forrige side

	<b>Test</b>	<b>Forventet Resultat</b>	<b>Resultat</b>	<b>Godkendt/ Kommentar</b>
<b>3.a</b>	Bruger vælger at afbryde handlingen	Visuel: Master returnerer til hovedmenuen	N/A	N/A
<b>4</b>	Master udskriver på skærmen, at ønsket Enhed er aktiveret eller deaktiveret	Visuel: Besked om aktivering eller deaktivering vises på skærmen	N/A	N/A
<b>5</b>	Bruger vælger "Afslut"	Visuel: Master returnerer til hovedmenuen	N/A	N/A
<b>6</b>	Der testes visuelt at Master går tilbage til hovedmenu	Hovedmenuen præsenteres for bruger	N/A	N/A
<b>3.a</b>	Bruger vælger "Afbryd"	Visuel: Hovedmenu vises	N/A	N/A

<b>UC4: Databehandling</b>				
	<b>Test</b>	<b>Forventet Resultat</b>	<b>Resultat</b>	<b>Godkendt/ Kommentar</b>
<b>1</b>	Der laves en testopstilling hvori det er muligt at ændre omgivelserne til sensoren således at den udlæste data ændres	Visuel: Udlæst data svarer overens med fysiske ændringer	N/A	N/A
<b>2</b>	Der ændres på omgivelserne således at grænseværdier for Enheden overskrides	Pumpe starter og begynder at vande	N/A	N/A
<b>2.a</b>	PIR-sensoren påvirkes med bevægelse	Vandingen bliver afbrudt på baggrund af registreret bevægelse.	N/A	N/A
<b>3</b>	Der testes visuelt at Enhed har gemt behandlerne af data i buffer	Data er gemt og kan tilgås via Masterens log	N/A	N/A
<b>4</b>	Master henter information fra bufferen i Enhed	Testes i 5	N/A	N/A
<b>5</b>	Bruger tilgår log i Master	Visuel: Informationen ligger tilgængelig i loggen	N/A	N/A

...fortsat fra forrige side

	<b>Test</b>	<b>Forventet Resultat</b>	<b>Resultat</b>	<b>Godkendt/ Kommentar</b>
<b>6</b>	Ved test måles det at der sendes et signal til Enhed ved registreret bevægelse	Enhed registrer signal fra PIR-sensor	N/A	N/A
<b>7</b>	Ved test ændres værdier for Enhed således at denne ville starte en vanding	Vandingen er ikke mulig i 30 min.	N/A	N/A
<b>8</b>	Der testes visuelt at Enhed har gemt behandlingerne af data i buffer	Testes i 5	N/A	N/A

<b>UC5: Tjek status</b>				
	<b>Test</b>	<b>Forventet Resultat</b>	<b>Resultat</b>	<b>Godkendt/ Kommentar</b>
<b>1</b>	Bruger vælger ”Tjek status” på Master	Visuel: Det er muligt at vælge ”Tjek status”	N/A	N/A
<b>2</b>	Der testes visuelt at Master giver information omkring status	Master henter seneste status fra log-register og udskriver dette på skærmen	N/A	N/A
<b>3</b>	Bruger vælger ”Tilbage”	Visuel: Det er muligt at vælge ”Tilbage”	N/A	N/A

<b>UC6: Udskriv log</b>				
	<b>Test</b>	<b>Forventet Resultat</b>	<b>Resultat</b>	<b>Godkendt/ Kommentar</b>
<b>1</b>	Bruger vælger ”Udskriv log” på Master	Visuel: Det er muligt at vælge ”Udskriv log”	N/A	N/A
<b>2</b>	Det testes visuelt at loggen udskrives på Master	Informationen vises på skærmen	N/A	N/A
<b>3</b>	Bruger vælger muligheden ”Tilbage”	Visuel: Det er muligt at vælge ”Tilbage”	N/A	N/A

<b>Ikke-funktionelle krav</b>				
	<b>Test</b>	<b>Forventet Resultat</b>	<b>Resultat</b>	<b>Godkendt/ Kommentar</b>
<b>1</b>	Systemet skal opsættes af autoriseret personale	Ikke testbar	N/A	N/A
<b>2</b>	Udenforstående bruger gennemlæser manual	Bruge har nu information nok til at kunne betjene systemet	N/A	N/A
<b>3</b>	Systemet skal kunne have en MTBF på min. 2 år	Ikke testbar	N/A	N/A
<b>4</b>	Software oppe tid > 1 måned, uden nødvendigt genstart	Ikke testbar	N/A	N/A
<b>5</b>	Der oprettes 18 Enheder i Master med unikke adresser	Ikke testbar. Der laves kun én Enhed	N/A	N/A
<b>6</b>	Sprinkler indstilles til at vande et areal af 360 grader	Sprinkler vander arealet 360 grader med min. 3,5 m radius	N/A	N/A
<b>7</b>	Der tilsluttes en yderligere Enhed efter opsætning af systemet	Ikke testbar. Der laves kun én Enhed	N/A	N/A
<b>8</b>	Systemet køre i 15 minutter. Der kontrolleres visuelt at loggen indeholder målinger efter det 15. minut	Loggen indeholder data fra tilkoblet Enhed.	N/A	N/A
<b>9</b>	Testet i punkt 8.	N/A	N/A	N/A
<b>10</b>	Sprinkler udskiftes uden det er nødvendigt at tilgå anden hardware	Sprinkler udskiftes og punkt 6 udføres igen	N/A	N/A
<b>11</b>	Personale skal nemt kunne tilgå sprinklere	Sprinklere er monteret således at disse let kan tilgås af personale	N/A	N/A
<b>12</b>	Enhed opsættes og forbindelse til Master afbrydes	Enheden opererer autonomt	N/A	N/A
<b>13</b>	Måling af lufttemperatur udføres ved stuetemperatur	Korrekt data for temperatur sendes til Master fra Enhed	N/A	N/A





# Bilag (CD-indhold) 9

---