



EasyWater 8000



AARHUS
UNIVERSITY
SCHOOL OF ENGINEERING



Projektrapport E3PRJ3 - Gruppe 3 - Vinter 2014

Projektdeltagere:

201270090 Simon S. Kirchheimer	SK
201270402 Jakob Schmidt	JS
201270432 Bjørn Sørensen	BS
201270478 Jesper O. Christensen	JC
201370794 Poul O. Pedersen	PO
201370970 Mick Kirkegaard	MK
201371023 Lennart Remme Balle	LB

Vejleder:
Tore Arne Skogberg

Dato for aflevering:
17. december 2014

Underskrifter

Dato for aflevering: 17/12-2014

Gruppemedlemmer:

Bjørn Sørensen (BS) 201270432

Jakob Schmidt (JS) 201270402

Jesper Christensen (JC) 201270478

Lennart Balle (LB) 201371023

Mick Kirkegaard (MK) 201370970

Poul Overgaard (PO) 201370794

Simon Kirchheimer (SK) 201270090

Vejleder: Tore Arne Skogberg

Arbejdsfordeling

Navne	Bjørn	Jesper	Jakob	Lennart	Mick	Poul	Simon
	Softwaregruppe		Hardwaregruppe				
Kravspecifikation	✓	✓	✓	✓	✓	✓	✓
Forundersøgelse	✓			✓			✓
Systemarkitektur	✓	✓	✓	✓	✓	✓	✓
Hardwarebeskrivelse			✓	✓	✓	✓	✓
Softwarebeskrivelse	✓	✓					
- Kommunikationsprotokol	✓				✓	✓	
FT-sensor			✓	✓			
PIR-sensor					✓	✓	✓
Sprinkler og pumpe						✓	✓
Forsyningsprint						✓	✓
Hardwareforbindelser			✓	✓	✓	✓	✓
Dataprotokol	✓						
Applikationsmodeller	✓	✓					
SPI					✓	✓	✓
Mastersoftware	✓	✓			✓	✓	
- Main	✓	✓					
- AddRemove		✓					
- Config		✓					
- OnOff		✓					
- LogView		✓					
- Status	✓						
- LoadData	✓						
- UnitDB		✓					
- Log		✓					
- UI	✓	✓					
- SPI API					✓	✓	
Enhedssoftware	✓	✓	✓	✓	✓	✓	✓
- Main	✓						
- Buffer	✓						
- LoadData	✓						
- Controllers	✓						
- Parameters		✓					
- SensorPackage			✓	✓			
- FT-sensor API			✓	✓			
- PIR API					✓		✓
- SPI-handler					✓	✓	

Resume

Følgende rapport og tilhørende dokumentation beskriver arbejdet med gruppens semesterprojekt for 3. semester på Aarhus School of Engineering. Formålet med projektet er at afprøve de metoder og emner som semesterets fag har introduceret. Rapporten beskriver arbejdsmetoderne der er brugt fra idé til produkt, det egentlige produkt og udviklingen af dette samt de overvejelser og værktøjer som er benyttet. Dokumentationen indeholder en udtømmende teknisk beskrivelse af systemet.

Udgangspunktet for systemet er at kunne styre vanding af goldbaner på en nem og simpel måde. Fra et computerprogram kan en bruger styre systemets dele og fastlægge grænser for hvornår vanding skal starte mv. Autonome enheder placeres rundt på goldbanens huller og fra disse enheder distribueres et netværk af sensorer som overvåger de miljømæssige data.

Udviklingsforløbet er styret efter ASE-modellen, som er en halv-iterativ projektledelsesmetode. Produktet er udviklet på to forskellige platforme. Brugerinteraktion sker igennem Devkit8000-platformen og det autonome system anvender PSoC4-boardet fra Cypress samt egen udviklet hardware til sensor-håndteringen.

Projektet er endt ud i et funktionelt system som kan aflæse data fra sensorerne og reagerer på disse. Selve vand-pumpe-konstruktionen til vanding af banerne er kun lavet som en simpel konceptuel opstilling.

Abstract

The following document describes the work and process of the groups 3rd term project at Aarhus School of Engineering. The purpose of the project is to use and evaluate the methods and subjects taught at this terms courses. The report describes how the product came from an idea to a physical product, as well as the details of the product and the methods used. The documentation holds all technical details about the product.

The product developed is an automatic watering system that helps controlling the environment around golf courses. From a computer program the user can control the whole system and set the limits from which the autonomous system reacts. The autonomous units are placed along the fairways with a grid of sensors placed in the ground on the fairway collecting environmental data.

Development is managed with the ASE-model, which is a semi-iterative project management process. Further more is it done on two different platforms namely the Devkit8000 for the user interaction component and the PSoC4 from Cypress as the core of the autonomous system along with specialised hardware for the sensors.

The results includes a functional user interface and a fully established data-collection system. Though limited by the implementation of the water pump system, which is suppressed due time restrictions.

Indholdsfortegnelse

Kapitel 1	Indledning	13
Kapitel 2	Ordliste	15
Kapitel 3	Opgaveformulering (MK PO)	17
Kapitel 4	Projektafgrænsning	19
Kapitel 5	Systembeskrivelse (MK PO)	21
Kapitel 6	Krav	23
Kapitel 7	Arbejdsproces	25
7.1	Udviklingsmodeller	25
7.2	Møder, tidsplan, logbog og referater	26
7.3	Arbejdsfordeling	27
7.3.1	Arbejdsgrupper	27
7.3.2	Rollefordeling	27
Kapitel 8	Udviklingsværktøjer	29
8.1	LaTeX og TexMaker	29
8.2	Multisim	29
8.3	PSoC Creator	29
8.4	Eclipse	29
8.5	Qt Creator	30
8.6	Microsoft Visual Studio 13	30
8.7	Logic	30
8.8	EAGLE	30
8.9	Microsoft Visio	30
8.10	Maple	30
8.11	Electronic Toolbox	30
8.12	Filhåndtering	30
8.12.1	GitHub	30
8.12.2	Google Drev	31
Kapitel 9	Systemarkitektur	33
9.1	Hardwarearkitektur (HW)	33
9.1.1	Domænemodel	33
9.1.2	Blokdefinitionsdiagram	33
9.1.3	Internt blokdiagram	34
9.2	Softwarearkitektur (BS JC)	35
9.2.1	Logical View	35

9.2.2 Deployment View	36
9.2.3 Implementation View	37
9.2.4 Data View	37
9.2.5 Kommunikationsprotokol (MK, PO)	38
Kapitel 10 Design	39
10.1 Hardwaredesign	39
10.1.1 SPI (MK PO SK)	39
10.1.2 FT-sensor (JS LB)	40
10.1.3 PIR-sensor (MK PO SK)	41
10.1.4 Strømforsyning og tilslutningsprint (MK PO SK)	41
10.1.5 Sprinkler-pumpe system (MK PO SK)	42
10.2 Softwaredesign (BS JC)	44
10.2.1 Applikationsmodel	44
10.2.2 Klassebeskrivelser	45
10.2.3 SPI (MK PO)	46
Kapitel 11 Implementering	47
11.1 Hardwareimplementering	47
11.1.1 PIR-sensor (SK MK)	47
11.1.2 Tilslutningsprint (SK)	47
11.1.3 Sprinkler-pumpe system	47
11.1.4 Udkumenteret pumpeløsning	48
11.1.5 sensorPackage-API	48
11.1.6 SPI (MK PO)	49
11.2 Softwareimplementering (BS JC)	49
11.2.1 SPI (MK PO)	50
Kapitel 12 Resultater	53
12.1 Software	53
12.2 Hardware	53
Kapitel 13 Erfaringer	55
13.1 Hardware-erfaringer	55
13.2 Software-erfaringer	55
13.3 Generelle erfaringer	55
Kapitel 14 Fremtidigt arbejde	57
Kapitel 15 Konklusion	59
Kapitel 16 Individuelle konklusioner	61
16.1 Bjørn Sørensen	61
16.2 Jakob Schmidt	61
16.3 Jesper Christensen	62
16.4 Lennart Balle	63
16.5 Mick Kirkegaard	63
16.6 Poul Overgaard	63

16.7 Simon Kirchheimer	64
Kapitel 17 Litteraturliste	65
17.1 Bøger	65
17.2 Hjemmesider	65
Kapitel 18 Bilags-CD indhold	67

Indledning 1

Denne rapport beskriver udviklingen af et system kaldet EasyWater8000. Systemet er tiltænkt som en hjælp til greenkeeperen på en golfbane, hvor opgaven med at holde græsset pænt varetages automatisk. Begrundelsen for systemets berettigelse er at det tager meget tid for en greenkeeper at kontrollere banens stand i form af fugtighed i græsset.

Systemet udvikles som et distribueret system hvor en hovedcomputer kan indstille grænseværdierne for fugtighed og temperatur på nogle autonome enheder som er placeret rundt på golfhullerne. Disse enheder starter selv vandingen hvis deres sensorer registrerer værdier uden for grænserne.

I processen fra idé til produkt har gruppen arbejdes sammen med at få idéen og lavet alle de indledende dokumenter som kravspecifikation og arkitekturbeskrivelse.

Se kapitel 2 for en komplet ordliste over relevante forkortelser og termer brugt i denne rapport samt projektdokumentationen.

Rapporten er delt op i kapitler inspireret af ASE-modellen. Først beskrives systemet og kravene her til, så følger beskrivelse af arbejdsprocessen og udviklingsværktøjer. Herefter kommer tre store dele med systemarkitekturen, designet og implementeringen med detaljeret beskrivelse af de enkelte faser. Til sidst følges der op på projektets gjorte erfaringer og muligheder for fremtidigt arbejde samt konklusioner og litteraturliste.

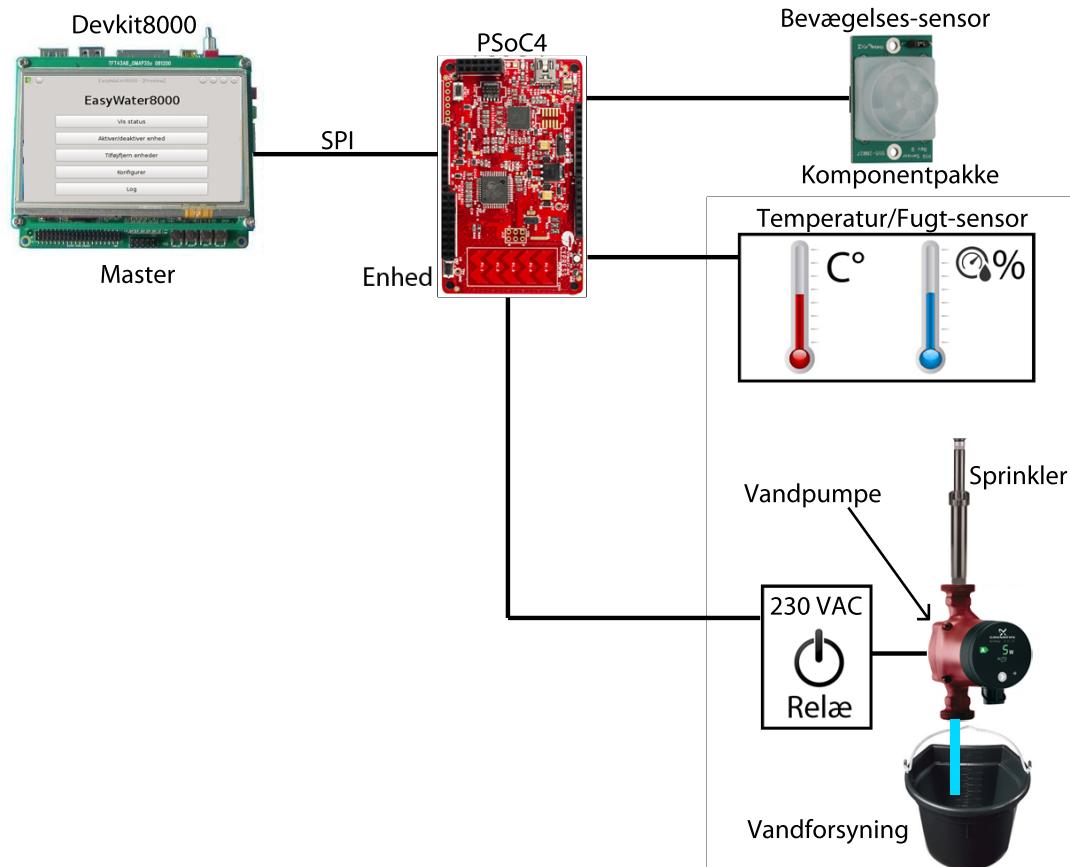
Ordliste 2

- AASH** Antal af samtidige hændelser
- API** Application programming interface
- BDD** Block Definition Diagram (Blokdefinitionsdiagram)
- 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
- 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)
- HW** Hardware
- IBD** Internal Block Diagram (Internt blokdiagram)
- 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
- PIR-sensor** Passive Infrared sensor
- PSoC** Udviklingsboard fra Cypress
- PSU** Power Supply Unit (strømforsyning)
- PWM** Puls-Width Modulation (Puls bredde modulation) - PWM spænding
- SW** Software
- 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)

Opgaveformulering (MK PO) 3

Konstruer et system til vanding af golfbaner. Systemet skal indeholde én Master (Devkit8000) som kontrollerer et netværk af Enheder (PSoC4). Kommunikationen imellem Master og Enhed skal foregå via den serielle kommunikation kaldet SPI. Via denne kommunikation skal Masteren kunne kontrollere de enkelte Enheder, samt indhente log-data fra disse. Enheden skal fungere autonomt, når denne er aktiveret. Enheden er tilkoblet en komponentpakke bestående af fugt- og temperatursensor samt sprinkler med tilhørende pumpe og vandforsyning. Enheden skal således selv styre vanding af trængende områder. En bevægelsessensor skal placeres ved indgangen til hvert golfhul, hvis der registreres bevægelse blokeres en evt. vanding i 30 minutter.

Figur 3.1 illustrerer ovenstående.

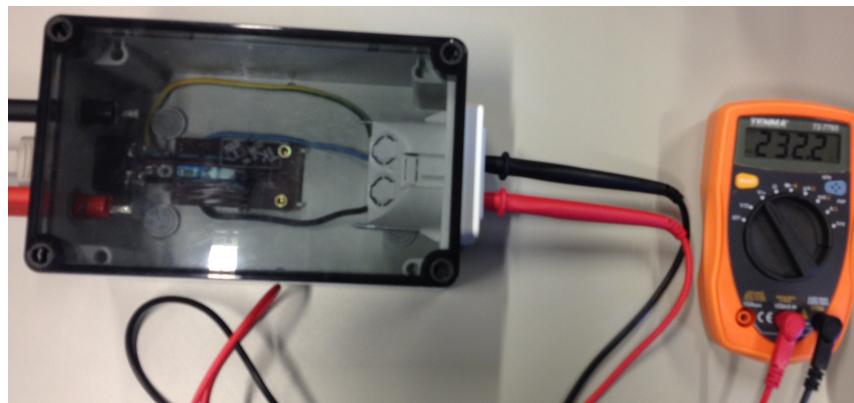


Figur 3.1. Oversigt over EasyWater8000 og forbindelser mellem delsystemer

Projektafgrænsning 4

Et fuldt integreret vandingssystem er, grundet manglende ressourcer og tid, ikke en mulighed på dette semesterprojekt, men der udarbejdes i stedet en skaleret prototype. Det vil sige at der i alt implementeres én Master og én Enhed med tilhørende komponentpakke og sprinkler. Foruden størrelsesbegrænsning er der fra skolens side givet forbud mod, uanset tidligere baggrund, at der arbejdes på 230 VAC. Men da der indgår en 230 VAC pumpe i projektet har skolen givet særtilladelse til at der laves et relæ-kredsløb i en lukket kasse, hvori alle ledende dele er helt afskærmet for fysisk kontakt som kan ses på figur 4.1.

Hurtigt i projektets forløb fik vi tildelt en pumpe fra Grundfos, det viste sig senere at denne ikke kunne bruges til det ønskede formål med at skabe et vandtryk, da denne er en cirkulationspumpe.



Figur 4.1. Relæ i afskærmet kasse. Multimeteret mäter 230 V på udgangen, viser hvor de konceptuelle pakker af software er placeret på hardwaren

Efter at den serielle kommunikation (SPI) var implementeret færdig og klar til test, viste det sig at overførelsесafstanden på 22 cm, som først var tiltænkt, ikke kunne lade sig gøre, da dataen blev korrupt undervejs. Denne afstand blev da nedsat til 7 cm for at kunne overføre data uden fejl.

I softwareimplementeringen er der tiltænkt en avanceret fejl-håndtering og filstruktur til at gemme opsætning- og logdata på Devkit8000. På grund af at vi mistede en softwaremand i gruppen er disse ting udeladt til fordel for at få den grafiske brugerflade og den autonome funktionalitet op at køre. Dette har resulteret i at alle fejlbeskeder på den grafiske brugerflade ikke er implementeret.

Prototypen udvikles med nogle andre tidsintervaller end dem specificeret i kravspecifikationen. Dette skyldes at vi gerne vil kunne se systemet virke i lille skala og i test øjemed.

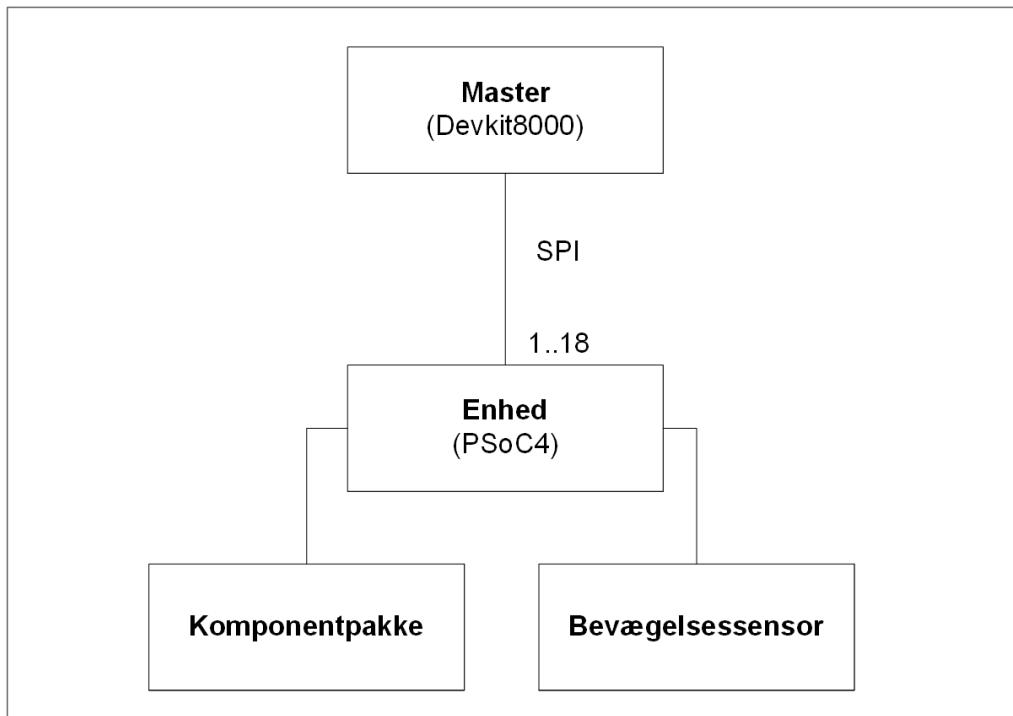
Tiderne er som følger.

- Datalogning fra Enheder på Master hvert 15. sekund
- Datalogning og autonom reaktion på Enheder hvert 8. sekund
- Afbrydelse af autonomt system ifm. bevægelse: 20 sekunder

Systembeskrivelse (MK PO)

5

EasyWater8000 er et samlet vandingssystem, hovedsagligt målrettet imod vanding af en golfbane. Her følger en detaljeret beskrivelse af systemet.



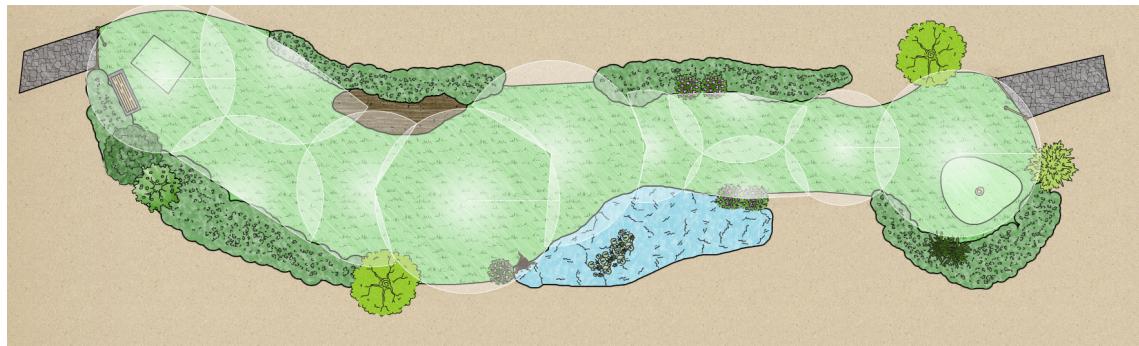
Figur 5.1. Oversigt over det samlede system

Figur 5.1 viser et simplificeret overblik over hvad EasyWater8000 består af. Der er en Master, som er hovedcomputeren i systemet med Linux (Angstrom) som styresystem. Masteren kommunikerer og styrer op til 18 Enheder der hver har en komponentpakke og en bevægelsessensor tilsluttet.

Komponentpakken er en samling af 3 komponenter, en temperatursensor og en fugtsensor til at hhv. måle temperaturen og fugtigheden i jorden omkring komponentpakken, samt en sprinkler til at vande græsset og jorden i samme område.

Bevægelsessensoren sidder ved hvert teested på golfbanen, og registrerer bevægelse i det område. Når der bliver registreret bevægelse vil sensoren give signal til at deaktivere vandingen på det givne golfhul, så golfspillerne ikke bliver utilsigtet våde.

Et golfhul kunne se ud som på Figur 5.2



Figur 5.2. Eksempel på golfbane med sprinklere

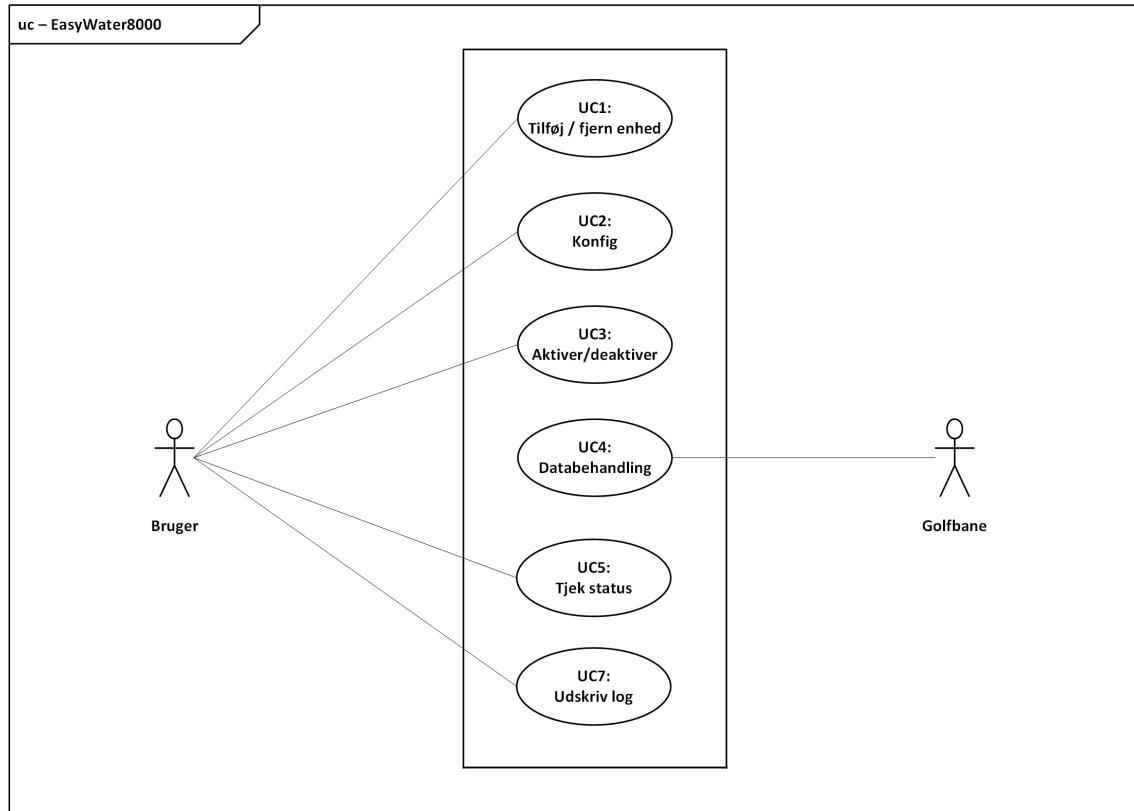
Enhederne kører autonomt, dvs. de selv monitorerer fugt og temperatur og vander hvis grænserne overskrides. Enhederne får deres grænseværdier fra Masteren og logger sine målte data til Masteren, via SPI-kommunikation.

Krav 6

EasyWater8000-systemet skal automatisk tænde for sprinklerne på golfbanen når græsset bliver for tør eller temperaturen bliver for høj. Systemet måler både fugt og temperatur og alt data samles i en log som kan printes på Masteren (hovedcomputeren). Desuden observerer systemet om en person eller spiller går ind på banen og deaktiverer vandingssystemet i en periode så systemet ikke vander hullet, mens spilleren er der på.

Greenkeeperen kan fra Master sætte en række værdier, for hvor tørt og varmt der skal være før der vandes.

Der er blevet lavet en kravspecifikation, som tager udgangspunkt i use case-diagrammet på figur 6.1. Use case-diagrammet viser hvilke use cases som Bruger og Golfbane har kontakt til. Den komplette use case-beskrivelse kan ses i projektdokumentationen, kapitel 2, Kravspecifikation.



Figur 6.1. Use case-diagrammet for EasyWater8000 beskriver operationerne systemet kan udføre

I tabel 6.1 ses en beskrivelse af aktørerne i EasyWater8000.

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)

Tabel 6.1. Aktørbeskrivelser til Use case-diagrammet 6.1

Ikke-funktionelle krav

Herunder ses en beskrivelse af de ikke-funktionelle krav i EasyWater8000.

Brugbarhed

1. Opsætningen skal ske af autoriseret personale
2. GUI skal kunne benyttes af Bruger, efter gennemlæst manual

Pålidelighed

3. Software IDLE tid: minimum 1 måned uden genstart

Ydeevne

4. Master skal kunne håndtere op til 18 Enheder
5. Sprinkler skal kunne vande et areal af 360° med radius på 3.5 meter
6. Der skal kunne tilføjes yderligere Enheder efter opsætning af systemet
7. Master skal hente data fra tilkoblede Enheder hvert 15. minut
8. Enhed skal hente data fra sensorer i et fast interval

Vedligeholdelse

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

Enheder

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

Begrænsninger

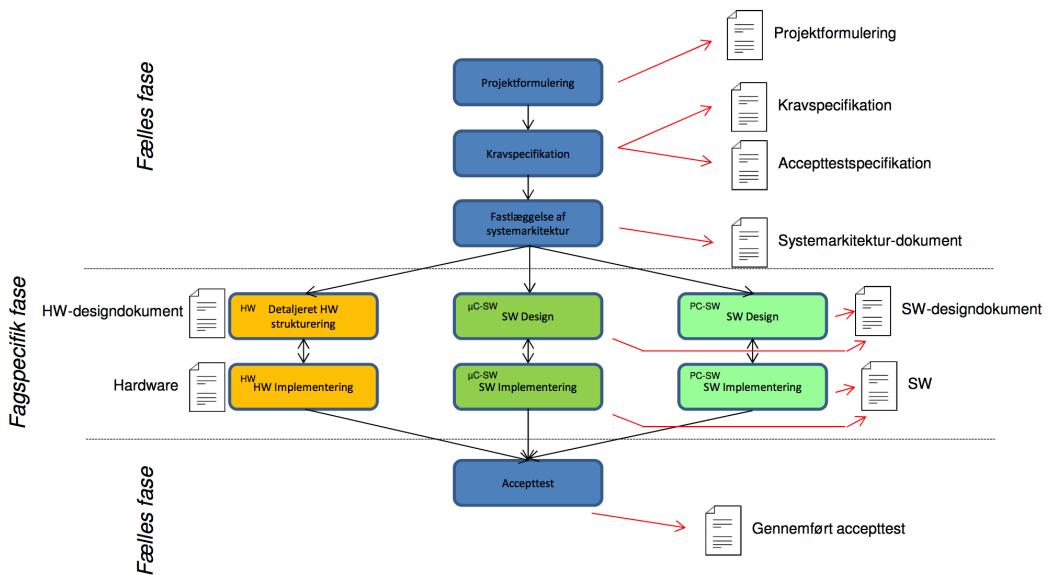
- Der udarbejdes en skaleret prototype, da vi ikke har adgang til en hel golfbane
- Grundet tidsbegrænsninger udvikles der ikke et fuldt system
- Som minimum skal der produceres en funktionel Master og én komplet Enhed
- Systemet udvikles således, at der til hver sprinkler tilhører en pumpe. Denne løsning 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

Arbejdsproces

7

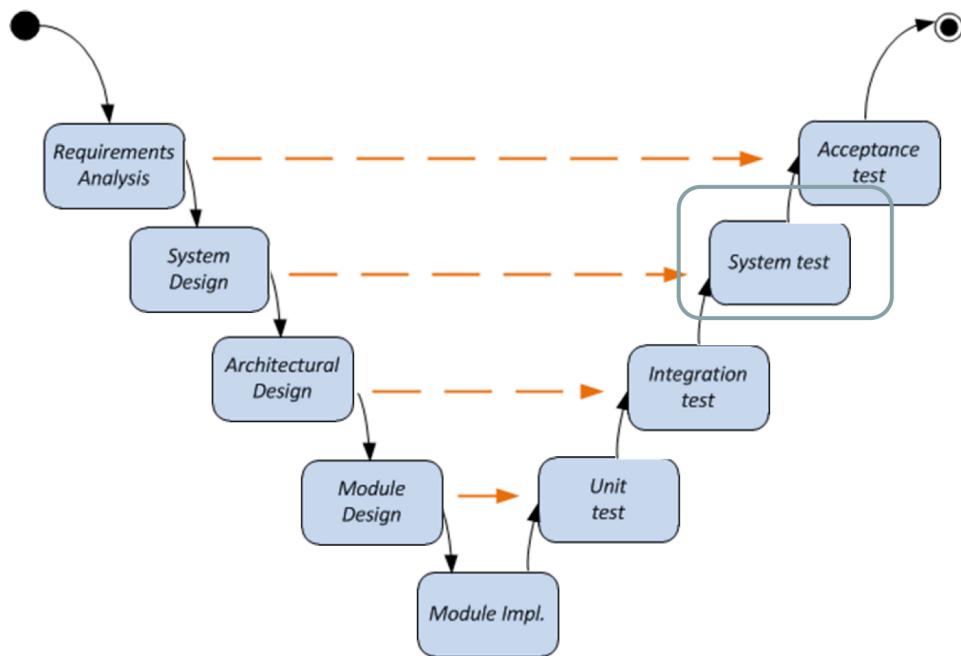
7.1 Udviklingsmodeller

Gruppearbejdet til dette semester-projekt har taget udgangspunkt i arbejdsmetoden fra sidste semester, i det gruppen dengang havde succes med netop denne. Arbejdsmetoden tager udgangspunkt i ASE-modellen, se figur 7.1. ASE-modellen har en fælles fase og en fagspecifik fase. Gruppen har arbejdet sammen i fællesfasen med udarbejdelse af projektformulering, kravspecifikation, accepttest og til dels systemarkitektur. Herefter har gruppen arbejdet i teams i den fagspecifikke fase, hvor design og implementeringen ligger. Efter den fagspecifikke fase samles gruppen igen i en fælles fase for den endelige accepttest udføres og rapport udarbejdes.



Figur 7.1. ASE-modellen. Fremgangsmetoden der er arbejdet ud fra

V-modellen, se figur 7.2, er benyttet sammen med ASE-modellen. V-modellen bygger på at hver enkelt delfase afsluttes før en ny delfase påbegyndes. En test af hver fase planlægges ved slutningen af den forgående.



Figur 7.2. V-modellen. Viser sammenhæng mellem faserne igennem forløbet

7.2 Møder, tidsplan, logbog og referater

I forbindelse med gennemførelse af dette projekt er der afholdt en række møder. Gruppemøder, vejledermøder samt reviewmøder.

Som hovedregel har der været afholdt ét vejleder møde og mindst ét gruppemøde hver uge, dog med enkelte undtagelser.

Vejledermødet har været fast hver tirsdag. Dette møde har fungeret som den primære kontakt mellem gruppen og vejleder. Vejleren har ofte haft adgang til foreløbig dokumentation, som vejlederen har givet konstruktiv feedback på.

Gruppemøderne har haft et fast punkt: Status, hvem er i gang med hvad og hvor langt er hver i sær med de pågældende opgaver. Ydermere er der afholdt tre trivselsrunder. Disse giver mulighed for at give ris/ros til de enkelte gruppemedlemmer samt mulighed for at gøre opmærksom på ikke-projekt-relaterede emner. Gruppen har fra start udarbejdet en gruppekontrakt¹. Denne kontrakt udgør et fælles standpunkt for gruppen og de indbyrdes aftaler. Gruppen indførte et koncept kaldet "Stå-op-møde", dvs. et møde der tager under 10 minutter. Mødets mål er planlagt, således at tiden ikke spildes.

Der er afholdt reviews. Et review af kravspecifikationen samt et review af systemarkitekturen. Disse reviews har fået aklaret nogle problemstillinger som reviewgruppen fremlagde. Gruppen aftalte efter samtalé med gruppe 2, at aflyse review af deres systemarkitektur, da de ikke ville vente med at rette dokumentet til før reviewet var modtaget, dette skete med vejleders sammentykke.

¹Gruppekontakten foreligger på bilags-CDen navn: Gruppekontrakt.pdf

Alle møder er skrifteligt dokumenteret af sekretæren med logbog og mødereförater.

Tidsplanen² er løbende blevet opdateret

7.3 Arbejdsfordeling

Gruppen har som helhed haft et godt og solidt samarbejde.

Allerede en måned efter projekts start oplevede gruppen ustabilitet i Jeppe Stærks fremmøde. Jeppe havde private problemer, som vi i gruppen selvfølgelig mente, at han skulle have tid til. Sidste gang vi så og hørte fra Jeppe var i uge 41. Siden har han, af ukendte årsager, valgt at ignorere alt kontakt til gruppen og dens medlemmer. Gruppen valgte d. 27/10-2014 at meddele Jeppe, skrifteligt, at han fra dags dato var ude af gruppen.

7.3.1 Arbejdsgrupper

Gruppen har hovedsagligt været opdelt i tre mindre arbejdsgrupper i forbindelse med faserne: Design, implementering og modultest. Forfattere til de enkelte kapitler er angivet med initialer. Overordnet er opdelingen: HW (Jakob, Lennart, Mick, Poul og Simon) og SW (Bjørn og Jesper). HW og SW har arbejdet i nedenstående tre undergrupper.

Gruppen indeholdende Bjørn og Jesper

Bjørn og Jesper har stået for alt software over HW API-niveau. Design og arkitektur er udviklet i tæt samarbejde og selve koden er implementeret ved at uddelegere de forskellige klasser, efterhånden som de er vurderet vigtige og nødvendige.

Gruppen indeholdende Jakob og Lennart

Jakob og Lennart har primært arbejdet med FT-sensoren, både HW- og SW-delen her for. Undervejs i forløbet er arbejdet delt lidt op, så Jakob hovedsageligt arbejdede med softwaren til PSoC4 og Lennart med den fysiske hardware. Der har været et godt arbejde og udregning af filteret samt designet af hele sensoren blev lavet sammen mens implementering heraf blev delt mere op.

Gruppen indeholdende Mick, Poul og Simon

Mick, Poul og Simon har igennem designfasen arbejdet tæt sammen om SPI-kommunikationen, PIR-sensoren, vandpumpe med tilhørende relæ og tilslutningsprintet. Mick og Poul har herefter stået for implementeringen af SPI-kommunikationen, mens Simon har færdiggjort de andre dele samt implementeret tilslutningsprintet.

7.3.2 Rollefordeling

Opsætning LATEXdokumenter

Hovedansvarlig: Mick

Sekretær i forbindelse med logbog og referater

Hovedansvarlig: Simon

Projektleder, mødeindkalder og holder samt tavlestyring

Hovedansvarlig: Poul

²Tidsplanen forefindes på bilags-CDen. Navn: Tidsplan.pdf

Vejlederkontakt

Hovedansvarlig: Poul

Udviklingsværktøjer 8

I det følgende afsnit beskrives, de benyttede, udviklingsværktøjer i forbindelse med dette projekt.

8.1 LaTeX og TexMaker

Både denne rapport og tilhørende dokumentation er skrevet i sproget L^AT_EX. Største delen af gruppen havde erfaring med L^AT_EX fra sidste semester projekt.

L^AT_EX er et tekstbaseret kodesprog som gør brugeren fri af layout således at fokus kan rettes mod indholdet. Det kræver selvfølgelig at man sætter sig ind i dette sprog og overholder kodestandarden.

TexMaker er benyttet som teksteditor til L^AT_EX. Denne editor gør det muligt at navigere rundt imellem forskellige .tex-filer og bygge dokumenterne via TexMakers indbyggede pdf-viewer.

Da hele dokumentationen skulle laves færdig kunne det konstateres at alle billeder skulle gemmes i .pdf-formatet, i stedet for i .png-formatet. Det gav bedre billedkvalitet og L^AT_EX reagerede hurtigere.

8.2 Multisim

National Instruments Multisim er benyttet til kredsløbsdesign og simulering af disse.

8.3 PSoC Creator

PSoC Creator fra Cypress er benyttet i forbindelse med softwareprogrammeringen af Enheden (PSoC4). PSoC4 er Cypress' nyeste ARM baserede Programmable System-on-Chip med en Cortex-M0 processor.

8.4 Eclipse

Eclipse er benyttet i forbindelse med softwareprogrammeringen af Masteren (Devkit8000). Eclipse kræver et Linux baseret operativsystem.

8.5 Qt Creator

Qt Creator er benyttet i forbindelse med den grafiske brugerflade på Masteren (Devkit8000).

8.6 Microsoft Visual Studio 13

Er anvendt til at skrive hardware-uafhængige C klasser til Enheden (PSoC4).

8.7 Logic

Logic fra Salae er en logik analysator der er benyttet til at gemme digitale signaler, så det herefter var muligt at analyse f.eks dele af SPI-kommunikationen. Logic har indbygget SPI-protokol.

8.8 EAGLE

EAGLEs PCB layout editor er benyttet til design af print til FT-sensoren.

8.9 Microsoft Visio

Microsoft Visio er benyttet til at udarbejde UML- og SysML-diagrammer.

8.10 Maple

Matematik software, til matematiske udregninger og grafer.

8.11 Electronic Toolbox

App af Marcus Roskosch til smartphones og tablets, til at hjælpe med komponent valg på integreret hardware m.m.

8.12 Filhåndtering

Der er benyttet 2 cloud resultatet væreservices i forbindelse med filhåndteringen for dette projekt

8.12.1 GitHub

GitHub er benyttet til versionsstyring af projektdokumenter dvs. Alle .tex-filer til dokumentation og rapport, softwarekildekode, UML- og SysML-diagrammer, Multisim-kredsløbsdiagrammer. GitHub sørger for at alle altid har nyeste version.

8.12.2 Google Drev

Google Drev er benyttet i flere forskellige forbindelser. Fælles dokumenter, møderefereater, logbog, tidsplan og fælles dokumenter i forbindelse med rette runder.

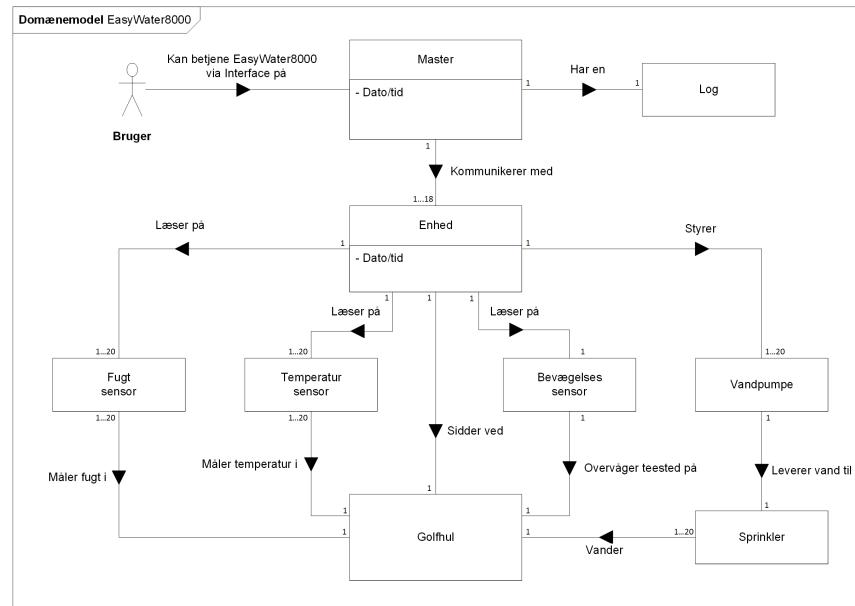
Systemarkitektur 9

9.1 Hardwarearkitektur (HW)

Følgende diagrammer beskriver hardwarearkitekturen.

9.1.1 Domænemodel

Første del af hardwarearkitekturen ligger i at få udarbejdet en domænemodel som er afbilledt på figur 9.1.

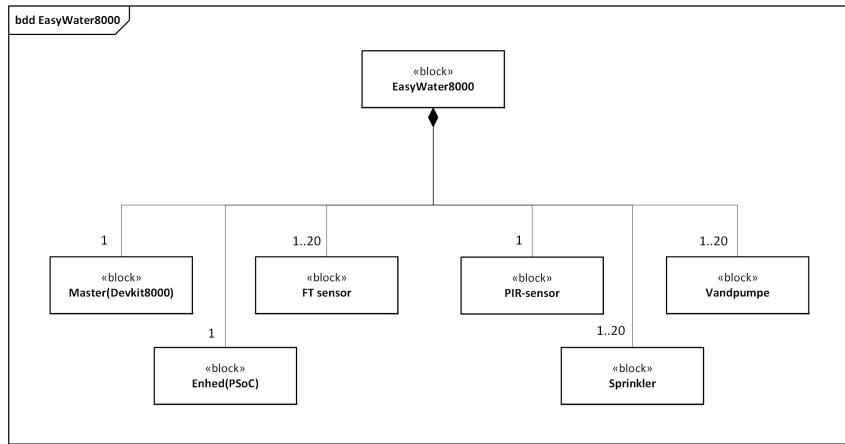


Figur 9.1. Overblik over systemets sammenhæng samlet i en domænemodel

Dette diagrams formål er at skabe et overbliksbillede for lægmænd. Af denne grund er der ikke anvendt tekniske eller faglige udtryk, men gjort brug af tale sprog. Hver kasse repræsenterer en fysisk genstand i systemet og pilene imellem disse beskriver en handling eller sammenhæng her imellem.

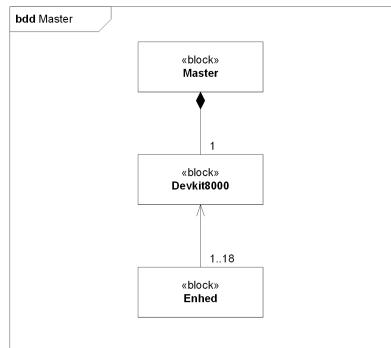
9.1.2 Blokdefinitionsdiagram

BDD-diagrammer beskriver de overordnet hardwaredele i systemet og hvilke dele disse er opbygget af. Et BDD er opbygget i lag, for hver gang en systemdel beskrives som bestående af flere komponenter udvides diagrammet med et lag til beskrivelse af disse dele. Et BDD kan ses på figur 9.2.

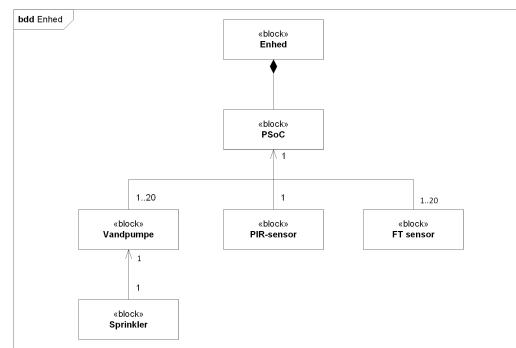


Figur 9.2. BDD af EasyWater8000 som samlet system

Yderligere ses det at EasyWater8000 er opbygget af 6 dele som alle ligger i andet lag og hvor EasyWater8000 alene udgør det øverste lag. Ydermere fortæller tallene ved hver blok hvor mange af disse der forekommer i systemet. Master og Enhed blokkene er så omfattende at disse er beskrevet yderligere med hver deres BDD som kan ses på figur 9.3 og 9.4



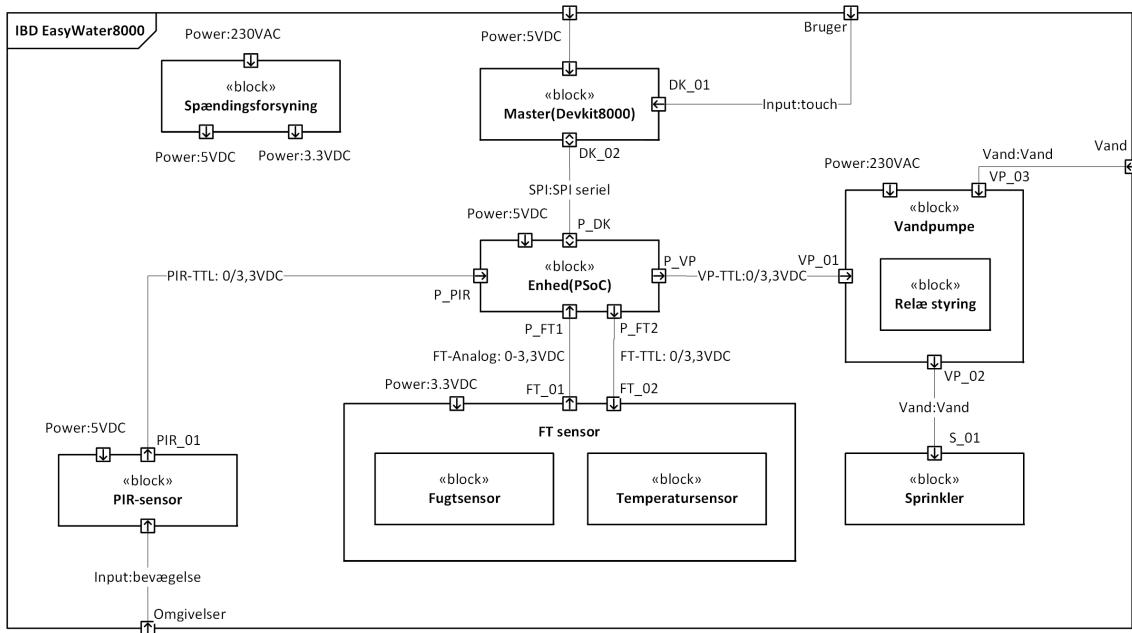
Figur 9.3. BDD af Master-blokken



Figur 9.4. BDD af Enhed-blokken

9.1.3 Internt blokdiagram

Når blokdiagrammerne er beskrevet, er det nødvendigt at definere hvordan disse er forbundet og hvilke signaler der gøres brug af. Dette er beskrevet i et IBD, et eksempel herfor kan ses på figur 9.5



Figur 9.5. IBD af EasyWater8000 med overblik af opbygningen af de enkelte blokke

I dette diagram bliver alle signaler og porte tildelt navne og typer. Ud fra disse er designet lavet. Ikke alle porte er tilsluttet et elektrisk signal, disse kan også styre andre elementer som skal transportereres rundt i systemet. EasyWater8000 gør f.eks. brug af en port til styring af vand.

9.2 Softwarearkitektur (BS JC)

For at få et overblik over hvad der skal laves af software anvendes en metode kaldet N+1¹. Metoden kan justeres til at dække det omfang som er nødvendigt. I projektet bliver følgende fire dele anvendt.

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

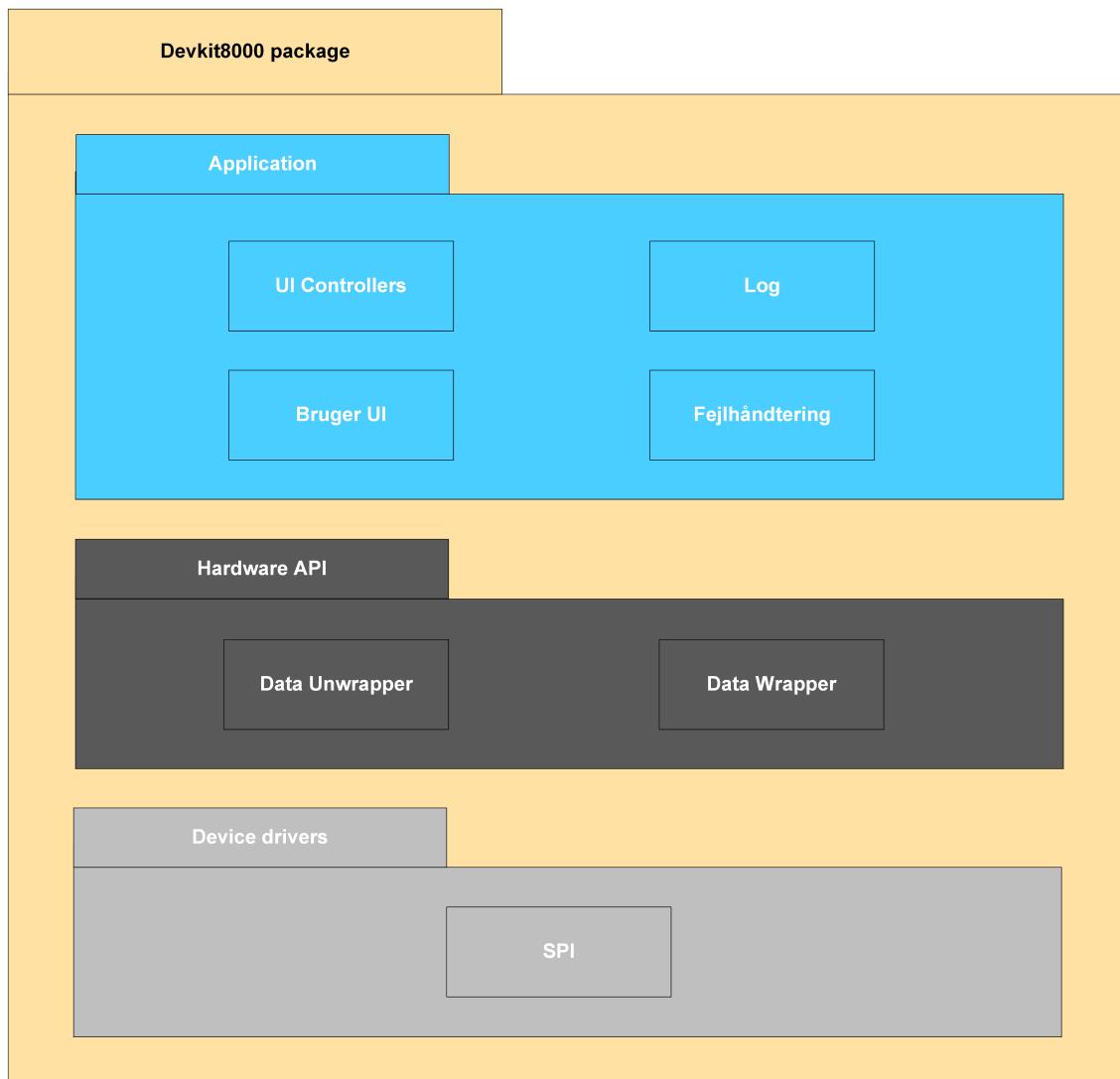
Her følger en kort beskrivelse af indholdet af hvert punkt og de væsentlige konklusioner der laves her fra. Se mere i dokumentationen i afsnit 4.3 Softwarebeskrivelse.

9.2.1 Logical View

Dette view giver et overblik over hvilke logiske blokke som skal bruges på de forskellige platforme, i dette tilfælde Master (Devkit8000) og Enhed (PSoC4). Det viser også forskellige softwarelag. Disse er udledt ved at kigge use case-beskrivelserne igennem og resonere over hvilke ting der konceptuelt er nødvendige.

¹<http://goo.gl/kU89oe> [2014-11-02]

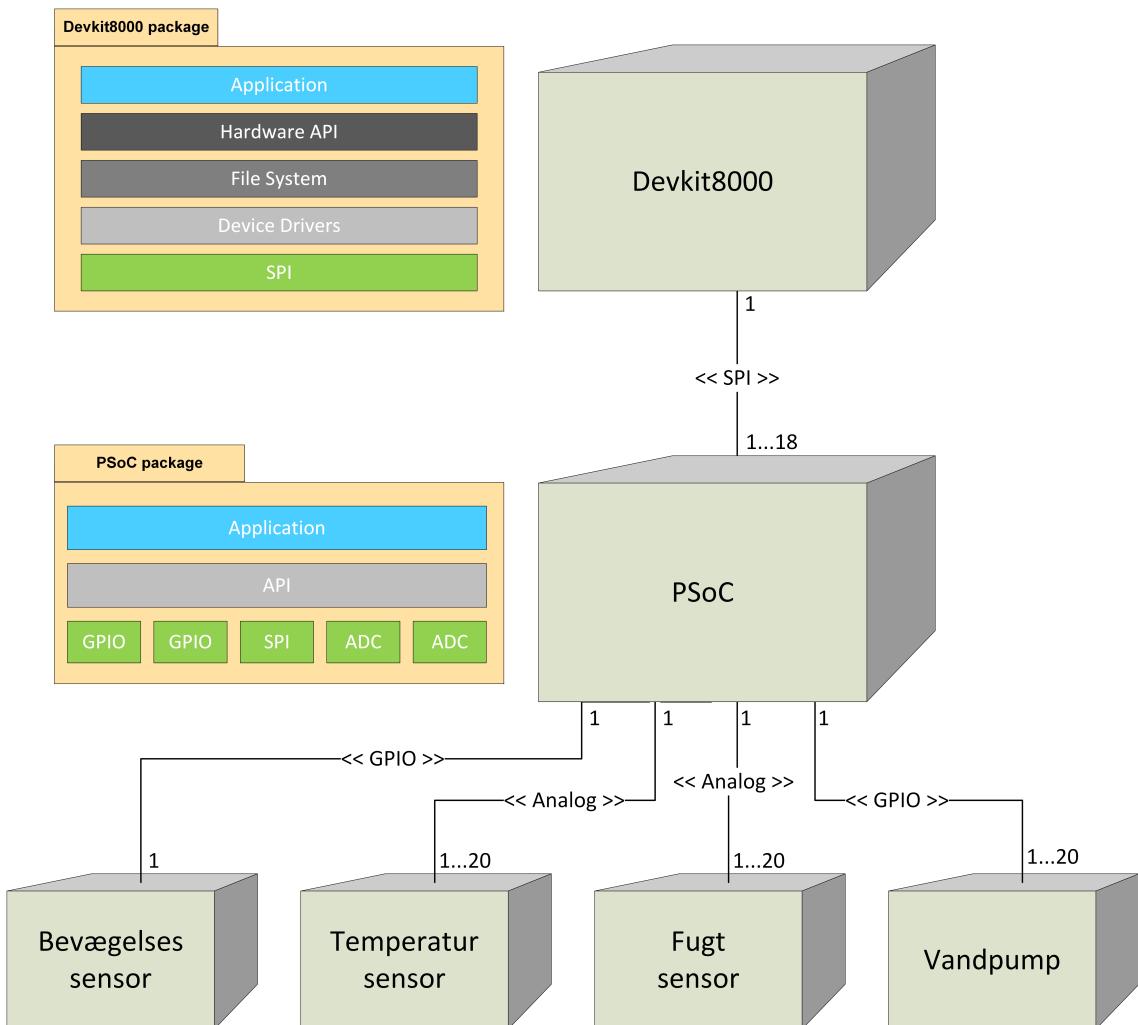
På figur 9.6 kan man se de tre lag samt, i alt, syv pakker. Bemærk at disse ikke nødvendigvis ender med at blive implementeret således, men de giver et overblik over den funktionalitet der er nødvendig.



Figur 9.6. Logical View for Devkit8000, viser softwarelag og -pakker

9.2.2 Deployment View

Her beskrives hvor de logiske pakker fra Logical View ligger på hardwaren. Dette er for at få et overblik over et distribueret system og for at kunne lave en vurdering af hvilket framework eller programmeringssprog der skal anvendes til de enkelte klasser. På figur 9.7 kan man se hvor de to logiske pakker skal implementeres.



Figur 9.7. Deployment View for EasyWater8000

9.2.3 Implementation View

Dette er en kort beskrivelse af hvilken filstruktur der anvendes til opbygningen af software. Dette beskrives for at ensrette kildefilerne og give et godt overblik over hvor de forskellige ting ligger gemt. Se dokumentationen afsnit 4.3.3 Implementation View for detaljer.

9.2.4 Data View

Her beskrives datahåndteringen i systemet. Oprindeligt var det tiltænkt at information om enheder skulle gemmes på Master så de kan findes frem, hvis systemet slukkes og tændes igen. Dette er ikke implementeret, men beskrives alligevel.

Her er beskrivelsen af selve filstrukturen af den gemte data og hvordan de enkelte filer skal være opbyggede. Et eksempel er vist i listing 9.1.

Listing 9.1. 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>;

```

9.2.5 Kommunikationsprotokol (MK, PO)

Kommunikationsprotokollen for SPI-kommunikationen er herunder beskrevet. Tabel 9.1 viser de mulige kommandoer for SPI-kommunikationen.

Opsætningen er som følger:

- Hastighed: 1 MHz
- SPI mode: 0 (CPOL 0 - CPHA 0)
- Antal bits: 1 char pr. transmission

PSoC4 kan håndtere SPI kommunikation ved 8 MHz, men der er ikke behov for så høj hastighed. Stabiliteten blev forbedret væsentligt ved at vælge en lavere hastighed.

SPI mode = 0 - er valgt på baggrund af default indstillinger.

CPOL = 0 - vil sige at clocken er lav når den er passiv (aktiv-høj).

CPHA = 0 - vil sige at data udlæses på rising-edge.

Der transmitteres en karakter pr. transmission dvs. 8 bits.

Tabel 9.1. Kommandoer for SPI-kommunikation

ASCII	HEX	Funktion
'A'	0x41	Aktiver Enhed
'D'	0x44	Deaktiver Enhed
'P'	0x50	Parametre sendes til Enhed
'V'	0x56	Verifier Enhed i systemet
'L'	0x4c	Forespørg logdata fra Enhed
'C'	0x4c	Write buffer og clearing af TX-buffer
'R'	0x4c	Læsning af data fra Enhed

For yderligere specifikation af de enkelte kommandoer se dokumentationen afsnit 4.3.5 Kommunikationsprotokol.

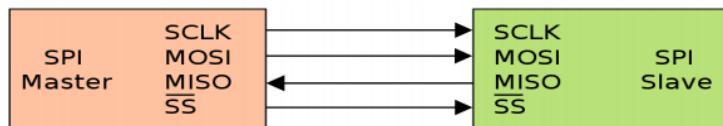
Design 10

10.1 Hardwaredesign

Hardwaredesignet består af en kombineret fugt- og temperatursensor (FT-sensor), en bevægelsessensor (PIR-sensor), et sprinkler-pumpesystem samt et tilslutningsprint.

10.1.1 SPI (MK PO SK)

Serial Parallel Interface (SPI) er en måde at lave hurtig seriell dataudveksling på. SPI er udviklet af Motorola og fungerer ved at man, som regel, har en enkelt masterenhed der styrer flere slaveenheder. Ved SPI er der ingen fejl-check og adressering af flere enheder kan være hardware krævende. I EasyWater8000-projektet er der SPI-kommunikation imellem Master (Devkit8000) og Enhed (PSoC4). Det giver muligheden for at overføre flere data på samme tid imellem disse to.



Figur 10.1. SPI-bus, viser hvordan master og enhed er forbundet

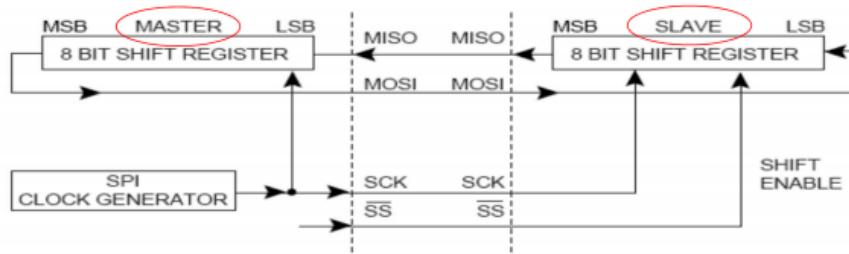
På figur 10.1¹ ses en typisk opkobling imellem master og slave. Det kræver fire ledninger.

- 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 skifтерegister-princippet, som ses på figur 10.2². Der vælges hvilken slave der ønskes at skrives til med slave select (SS), derefter shiftes et 8-bit register 1 bit af 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.

¹Kilde: Wikipedia om SPI

²Kilde: SPI Slide i I3GFV, Michael Alrøe

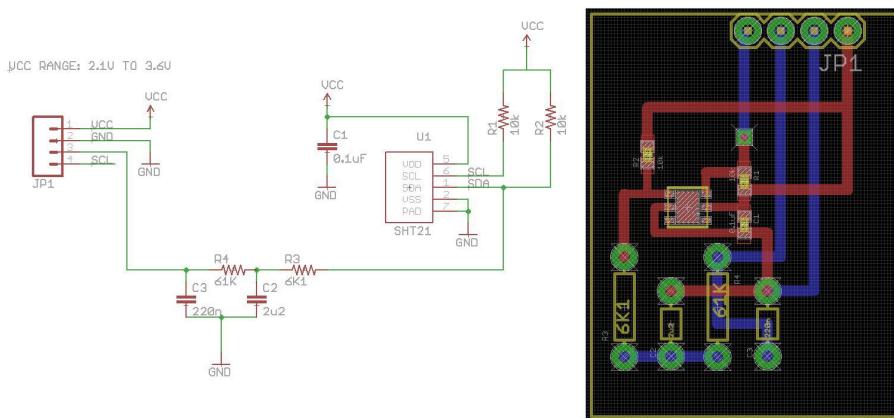


Figur 10.2. SPI-register, viser sammenhæng mellem registre og clock

En ofte anvendt kommunikation imellem master og slave foregår ved at master sætter SS til 0. Den fortæller nu til slaven at den er klar til dataoverførsel. Nu sender masteren en bit over til slaven og påbegynder en half-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 enheden bliver frigivet.

SPI kan køre både full- og halfduplex, dvs. at der kan være to-vejs kommunikation ved full-duplex (flere masterenheder) eller to-vejs kommunikation ved half-duplex, hvor en masterenhed styrer/skriver til en eller flere slaveenheder. I EasyWater8000 er der behov for at der kommunikeres to veje, men da Enheden fungerer som slave, kan den ikke selvstændigt sende til Masteren, men behøver et kald, om at nu skal den sende information. Derfor er der valgt half-duplex.

10.1.2 FT-sensor (JS LB)



Figur 10.3. Diagram og printudlæg af SHT21p kredsløb

Systemet indeholder en fugt- og temperatursensor. Sensorens opgave er at måle temperaturen og fugtigheden.

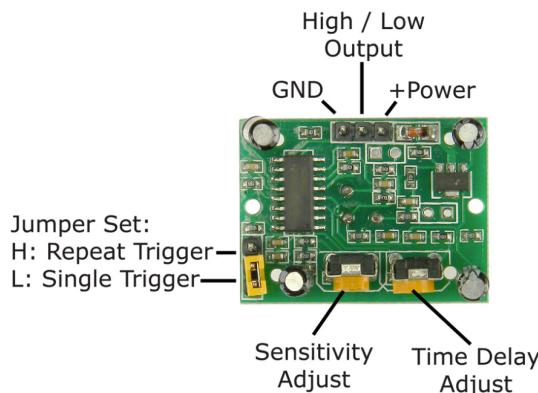
Sensoren udsender et PWM-signal, på ca. 120 Hz, hvor puls-bredden beskriver temperaturen og fugtigheden, alt efter hvilken tilstand der er valgt. Tilstanden vælges med en høj eller lav TTL-spænding, mere information om dette kan læses i dokumentationen afsnit 5.2, Temperatur- og fugtsensor.

Da ADC'en i PSoC4 ikke kan læse et PWM-signal, skal der findes en anden løsning. Løsningen er at midde PWM-signalen så det bliver lavet om til en varierende DC-spænding som ADC'en kan aflæse. Dette gøres med et 2. ordens lavpasfilter med en knækfrekvens på 12 Hz. Mere information om dette filter kan ses i dokumentationen afsnit 5.2.1, Midlingsfilter.

Der er også foretaget præcisions- og støjbehandling. Da nogle af komponenterne, som var til rådighed under projektet havde en tolerance på $\pm 5\%$, har denne ikke været meget nødvendig. For yderligere information om præcisions- og støjbehandling, se dokumentationen afsnit 5.2.2, Præcisions- og støjhåndtering.

10.1.3 PIR-sensor (MK PO SK)

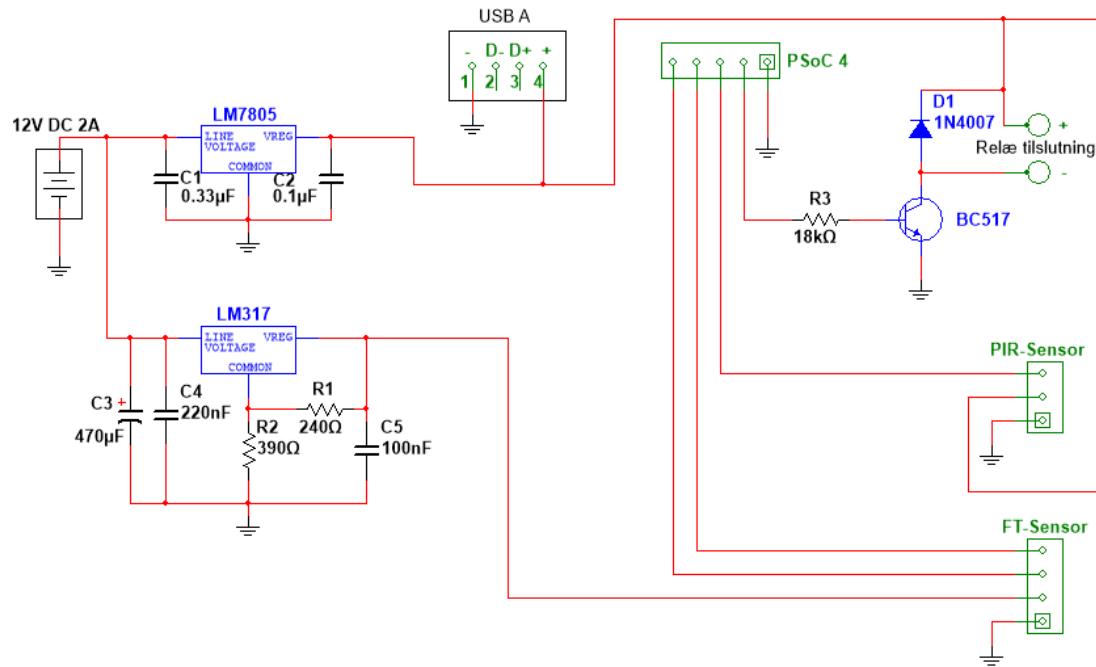
PIR-sensoren (HC-SR501) er en færdigbygget komponent. Den har tre tilslutningsben, et til forsyningsspænding, et til ground og et output ben der giver signal ved bevægelse. Der er mulighed for at indstille følsomheden og forsinkelse via to potentiometre. Jumperen indstiller hvordan PIR-sensoren skal trigge, ved "L" trigger den en enkelt gang og ved "H" trigger den flere gange. På figur 10.4 ses hvor de forskellige ben er placeret på bagsiden af PIR-sensoren. Driver-beskrivelse og nærmere indblik i PIR-sensoren kan findes i projektdokumentationen afsnit 5.3, PIR-sensor.



Figur 10.4. Billede af PIR-printet og dets indstillingsmuligheder

10.1.4 Strømforsyning og tilslutningsprint (MK PO SK)

Der er blevet udarbejdet et tilslutningsprint med tilhørende strømforsyning for at gøre systemet uafhængig af laboratoriet, se figur 10.5. Strømforsyningen er en 230 VAC / 12 VDC switch-mode-konverter.

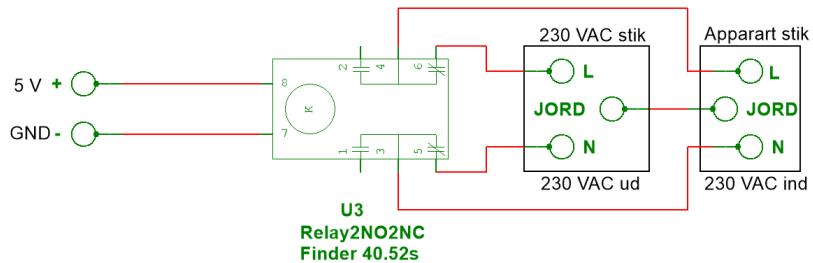


Figur 10.5. Kredsløb over samlet spændingsforsyning (3,3 V og 5 V) og tilslutninger til PSoC4

Tilslutningsprintet forsyner PSoC4, Sprinkler-relæ, PIR-sensor og FT-sensor. LM7805 og LM317 er spændingsregulatorer som er blevet brugt til henholdsvis at regulere spændingen ned til 5 VDC og 3,3 VDC. I projektdokumentation afsnit 5.5.1, Strømforsyning, kan beregninger ses for udgangsspænding og afsat effekt i spændingsregulatoren. Effektudregningen viser om det er nødvendigt at køle på spændingsregulatoren, dette er vigtigt at tage højde for, da komponenten ellers kan tage skade. Effektudregningen viser at det kun er nødvendigt med køling på LM7805, men der er fortaget køling på begge regulatorer alligevel. Det skyldes at det altid er en god ide at køle på en komponent, hvis der er mulighed for det, dette sikre bl.a. en længere levetid og berøringssikkerhed.

10.1.5 Sprinkler-pumpe system (MK PO SK)

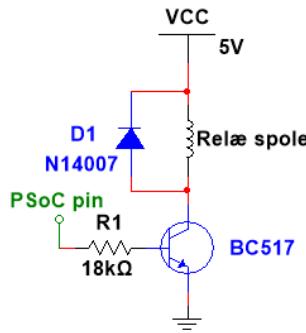
Sprinkleren skal forsynes med et vandtryk fra en vandpumpe. Grundfos har doneret en Alpha2 pumpe til formålet. Alpha2 pumpen kræver 230 VAC. Det overskrides hvad der må arbejdes med for studerende på Aarhus Universitet. Der er derfor, efter aftale med Torben Lund Jensen fra værkstedet, givet særskilt tilladelse til at designe og implementere et lukket 230 V / 5 V relæ. Figur 10.6 viser kredsløbet for det lukkede 230 V / 5 V relæ. Jordforbindelsen løber direkte igennem og er altid forbundet, mens fasen og nul tilsluttes/afbrydes af relæet.



Figur 10.6. Kredsløbsdiagram for lukket 230 V / 5 V relæ-boks

Relæstyring

PSoC4 giver ikke nok spænding eller strøm til sprinklerrelæet, det har derfor været nødvendigt at anvende en transistor som det ses på figur 10.7. Relæet kræver en strøm på 100 mA og det trækker indenfor 3,7 VDC - 8,8 VDC, dette problem løses med BC517 transistoren.



Figur 10.7. BC517 opsætning

Modstanden R1 er indsatt for at begrænse den strøm der går ind i transistorens base. Ifølge databladet må basen højst belastet med 100 mA. I_{base} strømmen er beregnet ud fra Ohms lov og der er valgt en modstand på 18 kohm, der begrænser strømmen til 0,1 mA, se formel 10.1.

$$I_{base} = \frac{3,3V - 1,4V}{18k\Omega} = 0,1mA \quad (10.1)$$

Når PSoC4-benet til BC517 transistoren går høj (3,3 VDC), så skabes der forbindelse imellem collector og emitter på transistoren. Herved er der spænding over relæets spole og relæets kontaktsæt slutter. Yderlige overvejelser og information kan findes i projektdokumentation afsnit 5.4.3, Relæstyring.

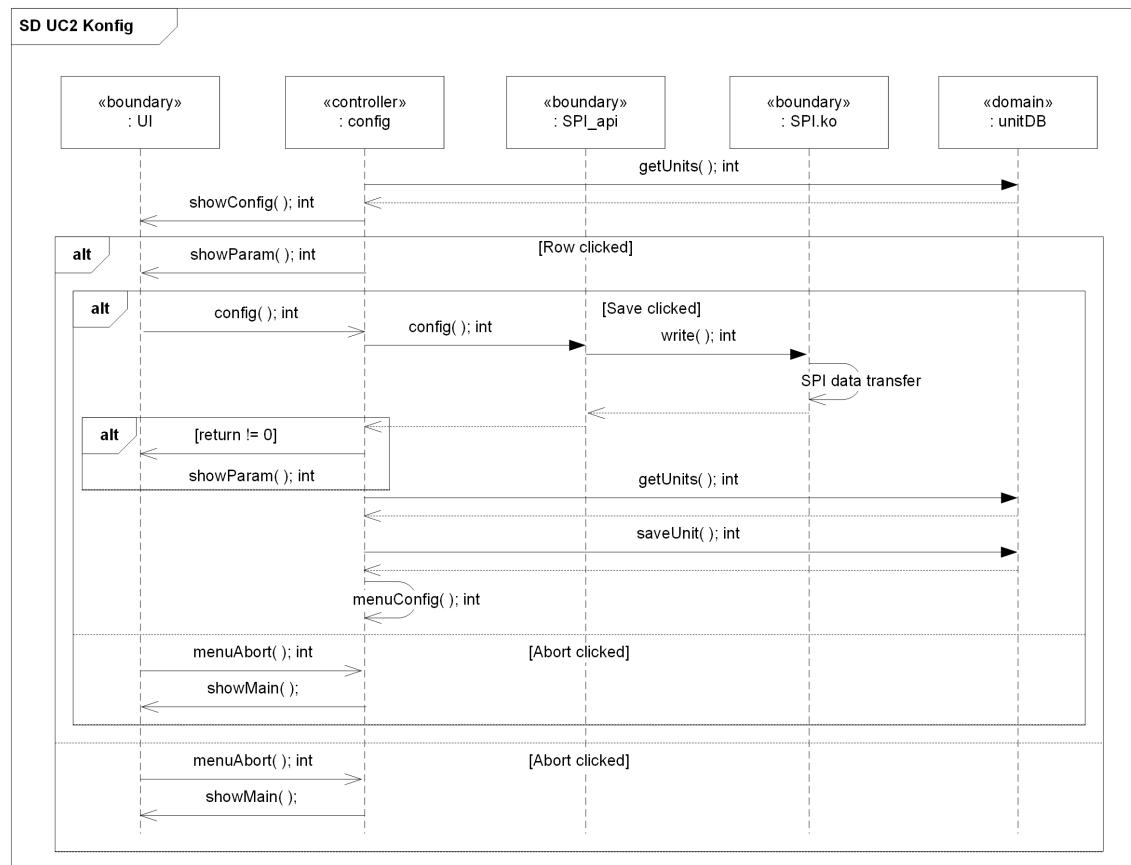
10.2 Softwaredesign (BS JC)

10.2.1 Applikationsmodel

I softwaredesign-fasen laves applikationsmodeller for de forskellige use cases, for at få et overblik over hvilke metoder og funktioner der skal bruges. Applikationsmodellen består af, at man først laver sekvensdiagrammer for hver use case. Hvis det findes nødvendigt for at få et overblik kan der også laves state machine-diagrammer over hver use case. Dette er ikke vurderet nødvendigt hvorfor det er udeladt.

Næste step i applikationsmodellen er at få alle metode-kald i sekvensdiagrammerne, over i et klassediagram. Dette klassediagram er ment som et overblik, som kan benyttes når der arbejdes med den endelige klassebeskrivelse.

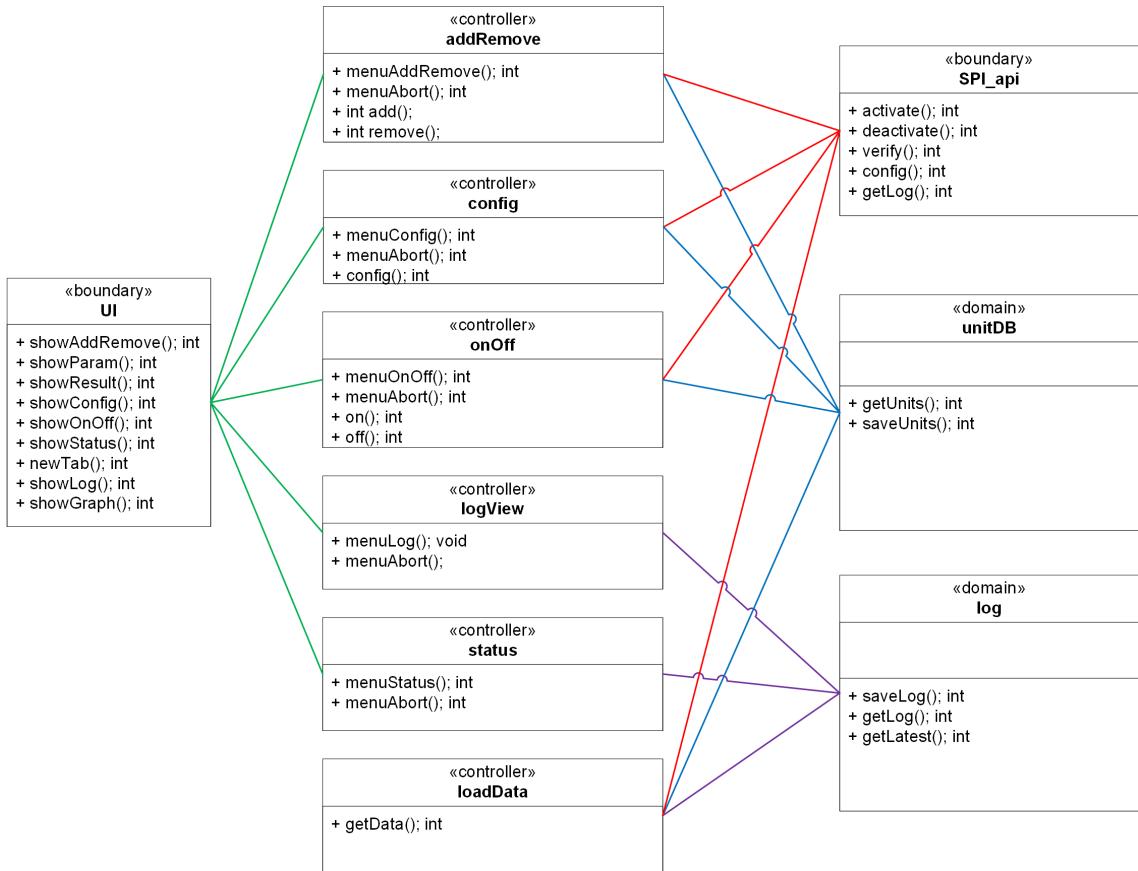
I klassebeskrivelsen besluttes der så hvilke datatyper der ønskes i de forskellige klasser og hvad de enkelte klasser hver i sær har ansvar for. Når klassebeskrivelserne er udarbejdet vil resultatet være et, endeligt, statisk klassediagram. Se dokumentationen afsnit 6.2 Applikationsmodeller for detaljer.



Figur 10.8. Sekvensdiagram fra applikationsmodel. Viser forløbet igennem programmet

I figur 10.8 ses et eksempel på et sekvensdiagram udviklet i applikationsmodellen. Sekvensdiagrammet er lavet ud fra use case 2: Konfig. Den viser alle de involverede klasser og metode-kaldene imellem dem.

På figur 10.9 ses det konceptuelle klassediagram som er udviklet i applikationsmodellen.



Figur 10.9. klassediagram fra applikationsmodellen. Giver en beskrivelse af klasse med tilhørende metoder og attributter

Applikationsmodellen er benyttet både til softwaredesign af koden til Devkit8000 og PSoC4. Det er dog to forskellige frameworks der arbejdes i, men dette har ingen indflydelse på applikationsmodellen, da denne beskæftiger sig med konceptet i hvordan koden skal fungere og ikke implementeringen af denne. På PSoC4 er det ikke muligt at arbejde i C++ og med objekt orienteret programmering som dokumentationen lægger op til. Dette løses ved at fortolke klassebeskrivelsen iht. UML-Light³.

10.2.2 Klassebeskrivelser

Efter applikationsmodellen blev klasserne beskrevet. Hver klasse har et overordnet ansvar. F.eks har controller-klassen, **Config** som blev beskrevet på figur 10.8, ansvaret for at styre UC2. Hver metode i klassen har således en beskrivelse af hvad metoden skal gøre, hvilke parametre den modtager og hvad metoden returnerer. Over hele projektet er det aftalt, at alle metoder skal returnere en integer for at der kunne laves fejl-tjek på dem. Det vil sige at når metoderne skal modulere noget data som denne fik tilsendt, så skal det være en pointer den modtager.

Efter klassebeskrivelserne er det konceptuelle klassediagram, med alle de nye medlemsdata, parametre og datatyper der benyttes i de pågældende klasser, blevet opdateret. Det nye,

³T-133 UML-Light, Finn Overgaard Hansen, 2013, Ingeniørhøjskolen Aarhus Universitet

opdaterede, klassediagram kan ses på figur 11.3 i softwareimplementeringen, afsnit 11.2. Se alle klassebeskrivelser i dokumentationen afsnit 6.2.3 Klassebeskrivelser.

10.2.3 SPI (MK PO)

Der udarbejdes et kernemodul, en driver og en API for SPI-kommunikationen for Master og en SPI-handler til Enheden. Disse er beskrevet i dette afsnit. For yderligere information se projektdokumentation afsnit 7.6, SPI.

Kernemodul og driver

Kernemodulet til SPI-kommunikationen skal sørge for at oprette en fil i filsystemet på Master. Det skal oprette en systemfil/systemnode der kan skrive og læses fra. Dette er praktisk da der kan skrives metoder der lige netop kan læse og skrive til en fil i filsystemet i Linux-operativsystemet. Kernemodulet kan genbruges fra HAL Exercise 7⁴ og driveren kan videreudvikles ud fra driveren i samme øvelse, så denne kommunikerer med én `char` per transmission.

API

SPI-APIet skal være en klasse, skrevet i C++, der skal indeholde metoder til at kommunikere via SPI med driveren og det dertilhørende kernemodul. Det skal bestå af en række metoder beskrevet i projektdokumentationen afsnit 7.6.2 SPI-API og afsnit 6.2.3, Klassebeskrivelser.

Alle metoderne skal bruge en filnode `/dev/spi_dev`, som den skriver og læser fra. Alle metoderne udarbejdes sammen med en SPI-handler på Enheden (PSoC4) der kan håndtere de forskellige kommandoer fra tabel 9.1.

Handler

Handleren skal som sagt udarbejdes sammen med APIet. Det skal være en interrupt-routine der handler på kommandoer i tabel 9.1, i afsnit 9.2.5, med et switch-case, der kan kalde metoder eller læse data ved en given kommando.

⁴Hardware abstraktioner. Exercise 7: LDD with SPI. Øvelse med SPI-kommunikation

Implementering

11

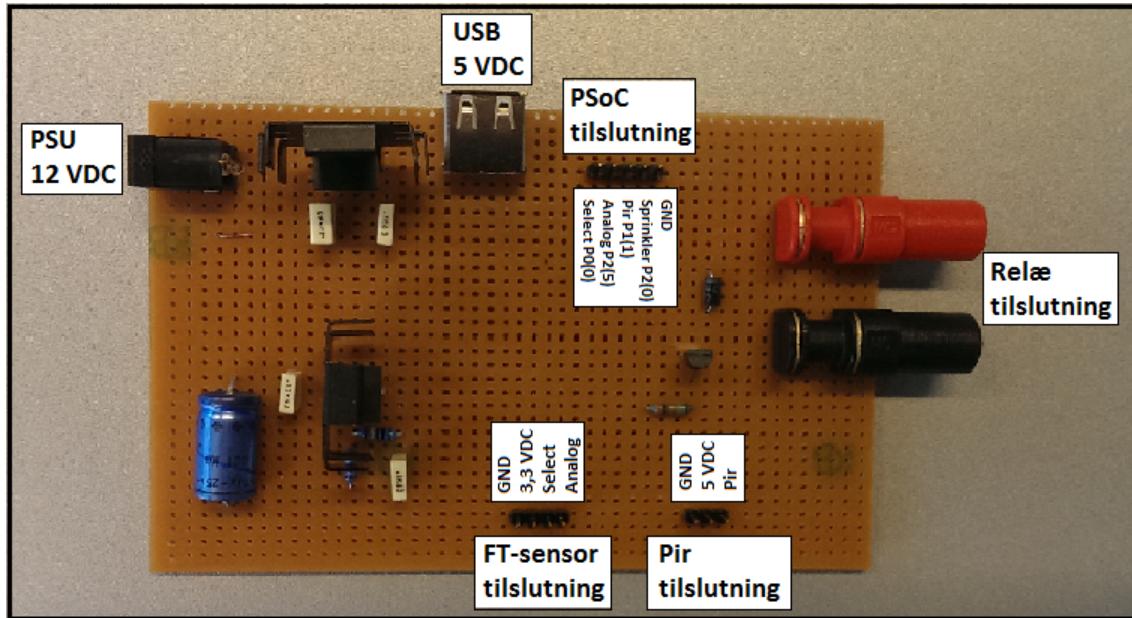
11.1 Hardwareimplementering

11.1.1 PIR-sensor (SK MK)

PIR-sensoren er en færdigbygget komponent, implementeringen af den er fortaget ved at lave et PIR-API. APIet er lavet i PSoC Creator, den forbinder PIR-sensoren med softwaren. Hvis der registreres bevægelse sættes en input pin på PSoC4 høj og der bliver startet en ISR der kalder en funktion i kontrolleren, som stopper vandingen på banen i 30 min. Yderlig information om PIR-API kan findes i projektdokumentation afsnit 7.7, PIR-API.

11.1.2 Tilslutningsprint (SK)

Implementeringen af tilslutningsprintet er fortaget på veroboard. På figur 11.1 kan det færdige resultat ses, samt hvilke tilslutningsmuligheder der er. I projektdokumentationen afsnit 7.2, Tilslutningsprint, kan der findes yderligere information her om.



Figur 11.1. Tilslutningsprint som det er implementeret på veroboard

11.1.3 Sprinkler-pumpe system

Som bekendt er der anvendt en sprinkler der kan vande i 360° og som er skjult under græsset når denne ikke er under vandtryk. Når pumpen starter og danner et vandtryk

kommer sprinkleren op over græsset og fordeler vandet jævnt i et område¹. Pumpen til at forsyne sprinkleren styres af et relæ som trækker når temperatur eller fugtighed kommer udenfor de valgte værdier.

Relæet til styring af Alpha2-pumpen er implementeret som beskrevet i designet, se projektdokumentationen afsnit 5.4.1, 230 V / 5 V relæ. For yderligere information omkring implementeringen se projektdokumentationen afsnit 7.4, 230 V / 5 V relæ.

Fittings til Alpha2 pumpen er fremstillet, se projektdokumentationen afsnit 7.5, Sprinkler og Alpha2 pumpe. Ved nærmere test viste Alpha2-pumpen sig, ikke at være brugbar til formålet.

Der er som midlertidig løsning valgt en anden pumpe som beskrives i afsnit 11.1.4 Udokumenteret pumpeløsning

11.1.4 Udokumenteret pumpeløsning

Alpha2-pumpen vidste sig i modultesten, se projektdokumentationen afsnit 8.6, Alpha2 pumpe, ikke at være brugbar som vandpumpe til sprinkleren. Herunder beskrives den valgte prototype løsning.

Alpha2-pumpen byttes ud med en boremaskinepumpe til vand, denne pumpe drives af en 230 VAC boremaskine. Haveslangen forbides med 1/2" slangetilkoblinger.

Figur 11.2 viser boremaskinepumpen med tilhørende slangetilkoblinger. Boremaskinens kontakt tvinges ind vha. en strips og herved pumpes der med maksimal kraft så snart relæets kontaktsæt sluttet.



Figur 11.2. Boremaskinepumpe som bruges til alternative løsning for Alpha2 pumpen

Boremaskinepumpen er testet med vand og sprinkler og virker som forventet.

11.1.5 sensorPackage-API

For at hardwaren kan kommunikere korrekt med den høj-intellektuelle software, er der oprettet hardware APIer, som sørger for at behandle hardwaren og trække det nødvendige data ud. Dette gør det nemmere, i tilfælde af at en hardwarekomponent med anden

¹En demonstration af dette kan ses i Demonstrationsvideo.mp4 på Bilags-CDen

virkemåde skal udskiftes, så skal APIet kun rettes til, mens resten af softwaren kan forblive uforandret.

SHT21p, som er den sensor EasyWater8000 gør brug af, er en analog sensor. Skulle man vælge at skifte sensoren ud til en digital sensor, kræver dette kun en ændring i APIet for at hele systemet virker korrekt igen.

Disse APIer er udviklet i PSoC Crator med tilhørende PSoC4.

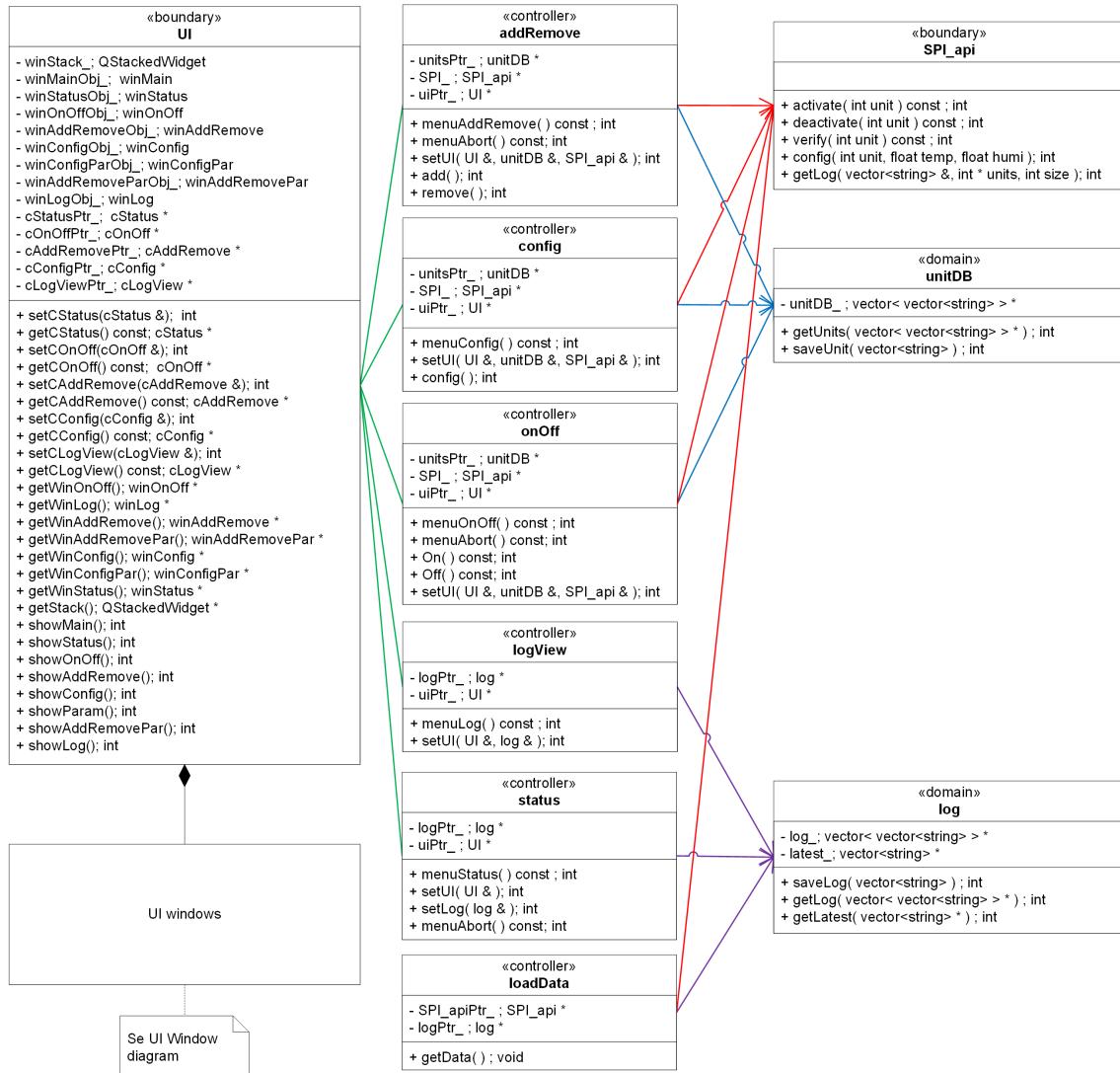
11.1.6 SPI (MK PO)

Den hardware mæssige implementering af SPI, er udført ved at sammenkoble Master og Enhed med et fladkabel. Der har været nogle udfordringer mht. længden af fladkablet, der har medført at afstanden imellem Master og Enhed er begrænset til 7 cm.

11.2 Softwareimplementering (BS JC)

Efter designet kan programmeringen påbegyndes. Den tager udgangspunkt i de beskrevne klasser og klassediagrammer. Da det er valgt at lave den grafiske brugerflade med Qt framework², anvendes deres API til mange standardfunktionaliteter. Det betyder at `std::string` typen omskrives til `QString`. Det betyder også at der anvendes en række grafiske elementer kaldet *widgts* til at opbygge klasserne. På figur 11.3 ses det endelige statiske klassediagram. Her fremgår alle klasser, deres metoder og attributter. Se dokumentationen afsnit 7.8 Software, for detaljer.

²<http://qt-project.org>, Qt Project Hosting [2014-12-10]



Figur 11.3. Statisk klassediagram for Master (Devkit8000)

Implementeringen er udført med programmerne Qt Creator og PSoC Creator. Se mere om disse i kapitel 8.

11.2.1 SPI (MK PO)

Der er taget udgangspunkt i HAL Exercise 7³ da SPI blev udarbejdet. Hele kildekoden for Kernemodulet er blevet genbrugt med kun få rettelser. Dele af driveren er også blevet genbrugt. I designet for SPI, er der lagt op til at der skal kunne sendes en 8 bit CHAR ved hver datatransmission. Derfor har det været nødvendigt at rette driveren til, så den kunne håndtere dette.

API

SPI API'en er en klasse med metoder, skrevet ud fra klassebeskrivelserne i software designet, se Projektdokumentation afsnit 6.2.3 Klassebeskrivelser. Implementeringen kan findes på

³Hardware abstraktioner. Exercise 7: LDD with SPI. Øvelse med SPI-kommunikation

bilags-CD, se CD/SW/Master/Kildekode/spi_api.

Handler

Handleren er en switchcase i en interrupt-rutine skrevet simultant med APIen, til at handle på forskellige kommandoer sendt fra Master via APIen. Den tilgår interrupt-rutinen ved en SPI transmission og handler i switchcasen udfra hvilken **CHAR** der modtages. Implementeringen kan findes på bilags-CD, se CD/SW/Enhed/Kildekode/spi_handler.

Resultater 12

På vedlagte CD¹ ligger en demonstration af systemet hvor vanding aktiveres af det autonome system.

12.1 Software

På software-siden af projektet er der opnået et responsivt og funktionelt GUI som styres via touch-skærmen på Devkit8000. Derudover er der opnået at få opsat SPI kommunikation imellem Devkit8000 og PSoC4. På PSoC4-siden er der opnået at få udlæst data fra sensorene og få formateret dem så de er klar til at blive sendt over SPI. Over SPI kommunikationen lykkedes det at sende kommandoer og at hente data fra sensorene så det kan fremvises for brugeren i GUI'en.

12.2 Hardware

På hardware-siden er det lykkedes at implementere en funktionsdygtig prototype. I hardwaregruppen er det lykkedes at få SPI-kommunikationen op at køre, samt at få hentet data ud af den analoge FT-sensor, denne data er via et API behandlet til en temperatur eller relativ fugtighed. Ydermere er der lavet et forsyningsprint, der forsynes fra en 12 VDC / 2 A PC-strømforsyning, der forsyner de nødvendige komponenter med korrekt spænding.

Den oprindelige Alpha2-pumpe, som vi fik sponsoreret af Grundfos, er desværre ikke mulig at anvende til det tiltænkte formål, så der er i stedet lavet en alternativ løsning, med en pumpe til en boremaskine. Denne styres stadig af relæet som det også var tiltænkt med Alpha2-pumpen.

¹Bilags CD: Demonstrationsvideo.mp4

Erfaringer 13

13.1 Hardware-erfaringer

Hardwaregruppe har opnået erfaringer indenfor arbejde med SPI, PSoC Creator og kendskab til diverse sensorer. En vigtig erfaring angående SPI er at afstanden imellem Master og Enhed ikke må være for lang, det medførte at signalet blev meget fejlfyldt. Kablet blev kortet ned til ca. 7 cm, som medførte at fejlraten blev minimeret til under 1%.

I forbindelse med FT-sensoren, erfares det, at det er nemmere at lave API til en analog sensor end til en digital sensor. Signalet fra en analog sensor kan tages direkte, hvor signalet fra en digital sensor kræver ekstra kodning. I vores tilfælde skulle der laves en analog spænding fra en PWM-spænding, som gav yderligere erfaringer inden for opbygning af filtre.

Da tilslutningsprintet skulle udvikles, var det meget vigtigt at tage hensyn til effektforbruget som skaber varme i spændingsregulatoren. Hvis der ikke blev foretaget korrekt køling kunne komponenten blive beskadiget.

13.2 Software-erfaringer

I software-gruppen har vi for første gang stiftet bekendtskab med at skabe et GUI. Dette er lavet i Qt frameworket som det også er første gang vi arbejder i. Vi har for første gang haft en integreret versionsstyring, i form af Git, i vores udviklingsværktøj (Qt Creator), hvilket viste sig at være meget brugbart. Generelt har arbejdet med Qt og Qt Creator været en fornøjelse, da dokumentationen for frameworkt er så veludført. Vi har fået en bedre forståelse for hvordan signaler og events fungerer i et event-drevet system.

I modsætningen til forrige år, har vi også haft flere mennesker inde over programmeringen. Hardware-gruppen har fået til opgave at lave drivere til deres hardware. Software-gruppen har lavet dokumentationen så hardware-gruppen havde et udgangspunkt til deres implementering. Det har virket rigtig godt og viser at hvis dokumentationen er i orden, kan arbejdet uddelegeres til andre som ikke har været med til at udtænke arkitekturen.

13.3 Generelle erfaringer

Det kan erfares at det er meget vigtigt med korte statusmøder, for at holde hinanden opdateret. I midten af projektet, kørte de forskellige grupper meget individuelt på og det medførte at grupperne mistede overblikket over hvor langt de andre hold var med deres opgaver. En mere specifik gruppekontrakt er at foretrække, da der opstod en del

tvivl om hvad konsekvensen var da Jeppe meldte sig ud af gruppen. I forbindelse med at gruppen mistede et medlem, har det været godt at gruppen kunne tilpasse de forskellige arbejdsopgaver efter behov, det indebar at hardware-grupperne kom til at lave en del software.

Det har været godt med tidlige deadlines, der har presset grupperne til at få lavet de forskellige ting færdig.

Fremtidigt arbejde 14

Fretnidigt arbejde for prototypen kunne være at udarbejde et mere kompakt design. Afstanden imellem Master og Enhed skal udvides så Enheden kan være placeret ved hvert golfhul, og Masteren er placeret i en af golfbanens bygninger, hvorfra greenkeeperen kan styre systemet. Denne kommunikation kunne med fordel være trådløs. Ydermere bør der produceres flere Enheder og komponentpakker således at samspillet kan vises for kommende kunder.

Den midlertidige pumpeløsning bør udskiftes til en fast vandforsyning og sprinklerne skal i stedet styrers af magnetventiler.

Systemet er pt. ikke sat op til at kunne håndtere flere Enheder. Hvis systemet skal videreudvikles ville dette være et vigtigt punkt. Derudover mangler fejlhåndtering og filstruktur på Masteren. Det er tiltænkt at Enhederne samt loggen skal gemmes i filer, således at data ikke mistes hvis Masteren slukkes.

En yderligere udvidelse af systemet kunne være en app, så greenkeeperen kan styre systemet fra en ekstern enhed. Sprogvalg på Masteren kunne med fordel være en funktion, således at produktet bliver attraktivt på det globale marked, og kan sælges i større omfang.

Konklusion 15

Projektets formål har været at udvikle et automatiseret vandingssystem til golfbaner kaldet EasyWater8000. Dette vandingssystem skal vha. sensorer vande når der er brug for det og blokere vandingen når der er folk på de enkelte golfhuller.

Der er opnået en prototype, som fungerer efter hensigten. Enheden fungerer autonomt og starter vandingen når det er nødvendigt. Aktiveres bevægelsessensoren blokeres vandingen. Gruppen er meget tilfreds med at have en funktionsdygtig prototype, der opfylder de grundlæggende krav til projektet.

Gruppen har arbejdet seriøst og fremadrettet med projektet. I forbindelse med brug af ASE-modellen har gruppen været delt op i den fagspecifikke fase. Det har givet muligheden for at de enkelte grupper kunne fordybe sig i bestemte områder, og herved bidrage til det samlede produkt.

Gruppemøderne, inklusiv de afholdte trivselsrunder har sørget for at vi som gruppe har haft det godt med hinanden og eventuelle problemer kunne afklares i tide. Optil udmeldelsen af Jeppe Stærk var der en del frustration omkring hvorvidt han havde tænkt sig at genoptage projektarbejdet og skolen generelt. Efter udmeldelsen af Jeppe Stærk kunne gruppen igen arbejde koncentreret og fremadrettet, det betød også at hele opgavefordelingen blev genfordelt.

Det kan konkluderes at udviklingsværktøjet L^AT_EX er rigtig smart ved store gruppeopgaver, da man sparer meget tid på opsætningen af rapporten og layout.

Individuelle konklusioner 16

16.1 Bjørn Sørensen

Endnu et semesterprojekt er overstået og udbyttet har været rigtig godt. Den tværfaglige læring er meget nyttelig og sætter alle de enkelte fag ind i en kontekst som virkelige sætter det i perspektiv.

Gruppearbejdet har kørt ud fra skabelonen som blev brugt på 2. semester, med få ændringer. Vi har fået en ny mand i gruppen og mistede en mand i midten af forløbet. Det har godt kunne mærkes at vi har fået en ny gruppe og tingene lige skulle sparkes i gang. Det at vi mistede en mand gav en del uro og frustration i gruppen og vi begyndte at arbejde meget uafhængigt af hinanden. En trivselsrunde fik sat ord på tingene og vi kom godt i gang igen og er nået i mål over al forventning.

Vi har i starten forsøgt at sætte nogle nye og optimistiske mål for projektet hvilket har bevirket at vi en uge før tid var færdig med projektet. Det har været rigtig godt og har givet en ro i den sidste fase til at få styr på dokumentationen og rapporten på en systematisk og produktiv måde.

Produktet er endt ud i et lidt simplere system end oprindeligt tiltænkt. Dog ramte vi mere realistisk og har generelt været bedre til at estimere opgavernes størrelse. Fire uger før afslutning af projektet havde vi i softwaregruppen ikke skrevet noget kode endnu, men kun planlagt designet. Det resulterede i at vi skrev det hele på to uger. Det er fedt!

16.2 Jakob Schmidt

3. semesterprojektet har for mig været udfordrende fagligt, specielt den serielle kommunikation har været en udfordring. I starten var der valgt en digital sensor, til datamåling af temperatur og fugtighed, men for at få en analog komponent med i projektet endte valget på SHT21p. Dette gjorde implementeringen af APIet noget nemmere. Jeg nåede at sidde et par dage, sammen med Lennart, for at implementere kommunikationen fra den digitale sensor til PSoC4, men uden meget succes. Da den analoge sensor udsendte en PWM-spænding og ikke en DC-spænding, som oftest er tilfældet, var det nødvendigt at få omdannet dette signal til en DC spænding. Dette blev løst med et simpelt 2. ordens lavpasfilter.

Det har været en mærkbar fordel, at vi har kunne beholde den gamle projektgruppen fra 2. semester. Det har betydet at vi hurtigt kom ind i de gamle roller, og at arbejdesprocessen

gik glidende fra en start af. Gruppen har dog fået lidt ændringer i form af et nyt medlem og et gammelt medlem der måtte forlade gruppen, men begge ændringer forgik i ro og mag.

Det har været en mærkbar fordel, at vi har kunne beholde den gamle projektgruppen fra 2. semester. Det har betydet at vi hurtigt kom ind i de gamle roller, og at arbejdesprocessen gik glidende fra starten af. Gruppen har dog fået lidt ændringer i form af et nyt medlem og et gammelt medlem der måtte forlade gruppen, men begge ændringer forgik i ro og mag.

Vi har i år været i noget bedre tid end sidste år, hvilket har været tydelig mærkbart de sidste dage op mod afleveringen. Vi har haft god tid, og har holdt dagene korte og overskuelige, hvor vi sidste år havde to riktig lange dage, som varede til sent på aftenen, for at blive færdig til tiden. Vi havde desuden tid til at få lavet en alternativ løsning til pumpen.

Personligt er jeg godt tilfreds med slutproduktet. Det virker efter hensigten og vi kan se at der sker noget. Vi kunne sagtens have lavet det mere komplekst og udfordrende, men da vi havde et stort overtal af HW-folk er det ikke blevet tilfældet. Det store overtal af HW-folk har været mærkbart, og en mere ligelig fordeling havde været at fortrække, specielt da rigtig store dele af projektet består af software.

16.3 Jesper Christensen

Udbyttet fra 3. semesterprojektet har været riktig stort. På softwaresiden har jeg stiftet bekendtskab med noget som jeg synes er riktig spændende at arbejde med, nemlig GUI. Det eneste user interface vi har haft i tidligere projekter har været et konsolvindue, så at vi i år har arbejdet med GUI, i embedded software, har været riktig spændende og lærerigt. Derudover har arbejdet med Qt været en fornøjelse, da deres dokumentation er så gennemført. Qt har også været et framework som er riktig fedt at have fået erfaring med, da det kan bruges på riktig mange platforme.

Vi har haft byttet om på hvilken rækkefølge vi laver vores software arkitektur i forhold til sidste år. 4+1 modellen ligger nu før applications modellen. Det blev vi vejledt til efter sidste års semesterprojekt, og det synes jeg har virket riktig godt, at det kom i den her rækkefølge.

Softwaregruppen som består af Bjørn og mig, fungerer riktig godt og vi arbejder generelt meget tæt sammen under hele udviklingsprocessen. Vi har forskellige styrker og svagheder som jeg synes supplerer hinanden riktig godt.

I projektgruppen har vi desværre haft et mandefald. Men vi kom godt igennem uden de store problemer alligevel. Vi er gode til at få taget nogle trivselsrunder engang imellem, så luften kan blive renset hvis nogen har været trætte af noget eller nogen. Generelt synes jeg at det er en fornøjelse at være en del af gruppen, og jeg synes vi arbejder godt sammen og supplerer hinanden godt. Generelt synes jeg, at det har været et rigtigt lærerrigt og vellykket semesterprojekt.

16.4 Lennart Balle

Som ny mand i en eksisterende gruppe var det med en smule nervøsitet, at jeg gik ind i denne gruppe, hvilket skulle vise sig at være unødvendigt. Jeg blev taget godt imod og følte mig velkommen fra starten af. Det var tydeligt at mærke at gruppen var meget ambitiøs, men stadig kunne forholde sig realistisk til, hvad der var muligt at nå, på den tid der var til rådighed. Kommunikation i gruppen har fungeret upåklageligt, lige undtaget nogle få gange hvor man fokuserede meget på sin egen opgave. Dette problem blev dog opdaget hurtigt alle gange under en jævnlig trivsels-runde.

Fagligt har alle været godt udfordret og der har været mange problemer undervejs, dog har gruppen været god til at hjælpe hinanden på tværs af kompetencer. I projektet er anvendt \LaTeX som tekstbehandlings-værktøj. Dette har været svært at finde ud af i starten, men viste sig senere at være en meget stor fordel, og har sparet gruppen megen tid i sammensætning af projekt- rapport og dokumentation. I projektgruppen havde jeg, sammen med Jakob Schmidt, ansvaret for fugt- og temperatursensoren, hvor jeg stod mest for hardwaren og Jakob mest for programmering af PSoC4.

Alt i alt er jeg godt tilfreds med resultatet af dette semesterprojekt, og jeg føler der har været godt styr på det hele vejen igennem.

16.5 Mick Kirkegaard

Så blev 3. semester-projekt afsluttet, og det har ikke været uden bump på vejen. Vi har dog efter min mening fået udarbejdet et fint og fungerende projekt. Jeg har været rigtig meget inde over SPI-kommunikationen og føler virkelig, jeg har fået lært meget om seriel kommunikation. Det har betydet, at jeg mere eller mindre kun har siddet og programmeret til Devkit og PSoC4, og jeg har nok fundet ud af, at det er der, hvor jeg befinder mig bedst. Selvom jeg synes, at det er meget udfordrende, er det også meget tilfredsstillende for mig at få noget eksisterende hardware til at agere, som man har kodet det til.

Gruppen fik et nyt medlem i starten af semesteret, og det har haft sine udfordringer mht. \LaTeX og vores arbejdsprocessor, som vi mere eller mindre har genbrugt fra sidste semester, og som han derfor skulle sættes ind i.

Da gruppen mistede Jeppe et godt stykke inde i processen, gjorde det, at gruppen gik op i limningen. Jeg mener faktisk ikke, at vi er kommet helt tilbage efter det endnu - på trods af flere trivselsrunder. Der er efter min mening stadig knas i krogene.

I bagklogskabens lys burde vi have sightet meget lavere, så vi kunne have et reelt og færdigt produkt i hånden i stedet for en nedskaleret prototype; så et golfbanesystem var måske ikke lige sagen.

16.6 Poul Overgaard

Jeg er stolt og godt tilfreds med hvad vi som gruppe har opnået i forbindelse med semesterprojektet. Vi var som gruppe opmærksomme på at have "de realistiske briller" på fra start, således at det var muligt at ende ud med et færdigt og funktionelt produkt.

Processen har været særdeles lærerig og med fagligt stort udbytte. Kursernes tværfaglige kobling giver ny viden og gør herved projektet til en realitet.

ASE-modellen giver, via de to faser, en god arbejdsfordeling. Fællesfaserne er gode til at holde sammen på projektet og den fagspecifikke fase gør det muligt at dykke ned og gennemarbejde et bestemt område. Jeg har specielt fået meget ud af den fagspecifikke fase i samarbejde med Mick og Simon.

Jeg har i gruppen fungeret som projektleder, det er en naturlig position for mig og raret at gruppen gerne så mig i den rolle igen. Der er udfordringer som projektleder der skal tackles med omhu, det er her vigtigt at bevare roen og respekten overfor gruppen og se fremad. Det er bestemt en gruppe jeg ser mig selv arbejde sammen med igen.

16.7 Simon Kirchheimer

3. semesterprojektet har givet et stort indblik i at arbejde med bl.a. PSoC Creator, diverse sensorer og SPI-kommunikation. Det har været et spændende projekt med mange udfordringer. Jeg er på stærkstrømslinjen og har derfor fokuseret på HW delen i projektet. Det indebar at arbejde i en gruppe med Mick og Poul, hvor vi havde fokus på design af SPI, sprinkler-relæet, tilslutningsprintet og PIR-sensoren. Da jeg ikke har HAL undervisning og kendskab til Devkit8000 var jeg kun med i opstartsfasen af SPI-delen, derefter forsatte jeg med at færdigudvikle tilslutningsprintet og lave PIR-delen.

Jeg tog ansvaret som referant til alle gruppe- og vejledermøder, det har givet et godt grundlag for at få et struktureret arbejde, da alle kan gå ind i logbogen/referatet og se hvad vi snakkede om. Undervisningen på 3. semester har givet os de nødvendige redskaber til at løse de problemer vi kunne risikere at møde. Vi mistede desværre et gruppemedlem efter korttid, men jeg synes vi som gruppe har håndteret dette godt. Vi fik udarbejdet en funktionsdygtig prototype af hele systemet, som der eventuelt kunne arbejdes videre på til et komplet system. Gruppen har fungeret godt med løbende trivselsrunder og godt samarbejde, det kan skyldes at næsten alle kendte hinanden fra 2. semesterprojektet. Alt i alt synes jeg at projektet har været lærerigt og styrket fagligheden til 4. semester.

Litteraturliste 17

Foruden de tilgængelige Campus-dokumenter i forbindelse med EPRJ3 er nedenstående litteratur benyttet.

17.1 Bøger

Analogteknik T-005, Tore Skogberg, 2014, Ingeniørhøjskolen Aarhus universitet.

UML-Light T-133, Finn Overgaard Hansen, 2013, Ingeniørhøjskolen Aarhus Universitet.

17.2 Hjemmesider

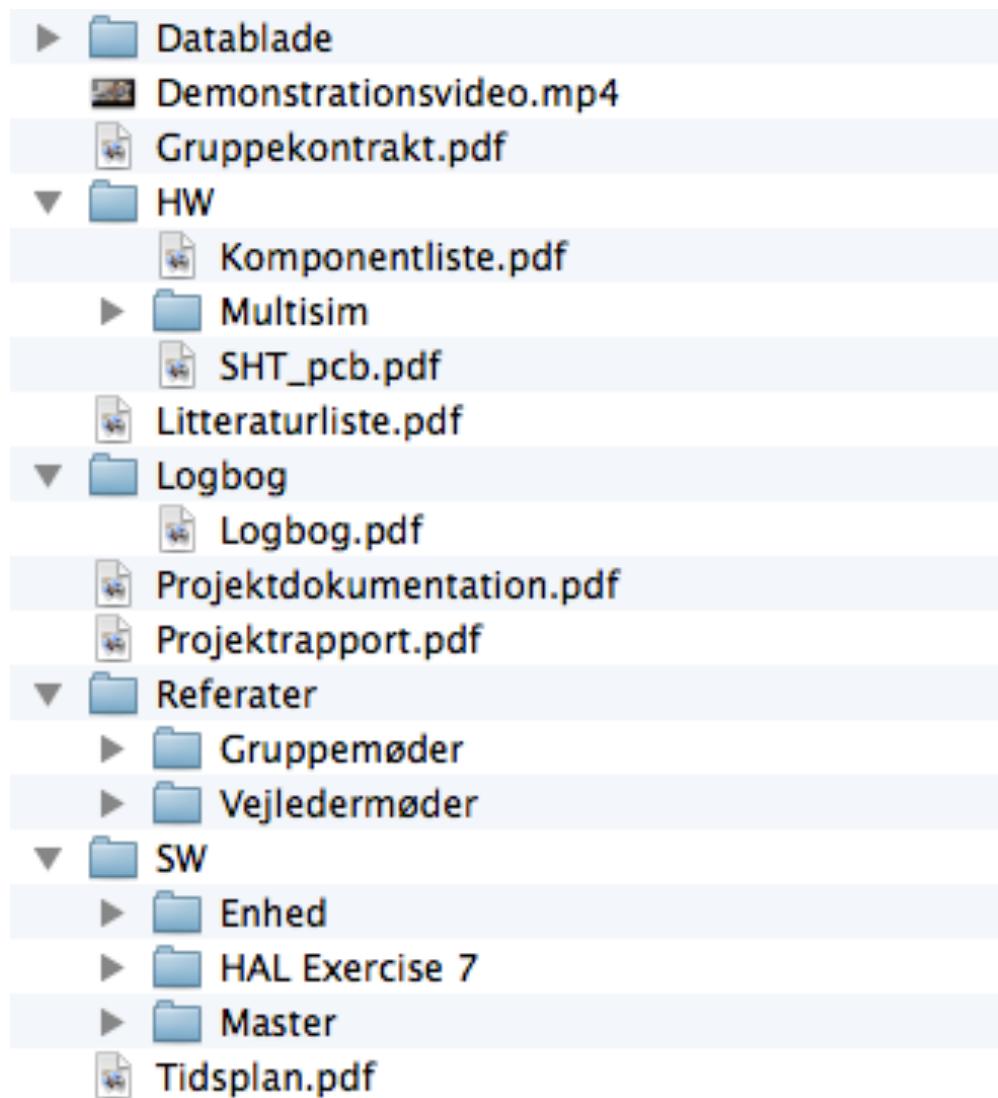
<http://qt-project.org>, Qt Project Hosting [2014-12-10]

http://da.wikipedia.org/wiki/Serial_Peripheral_Interface-bus, Wikipedia om SPI [2014-12-11]

Bilags-CD indhold

18

Figur 18.1 viser oversigten over filstrukturen på den vedlagt CD med bilag.



Figur 18.1. Filstruktur for bilags-CD