Projektrapport
Gruppe 1 - AU2
Den intelligente bil

4. Semesterprojekt E4PRJ4
Ingeniørhøjskolen, Aarhus Universitet
Vejleder: Arne Justesen

3. december 2015

Navn	**Studienummer**	**Underskrift**
Kristian Thomsen	201311478	
Philip Krogh-Pedersen	201311473	
Lasse Barner Sivertsen	201371048	
Henrik Bagger Jensen	201304157	
Kenn Hedegaard Eskildsen	201370904	
Karsten Schou Nielsen	201370045	
Jesper Pedersen	201370530	

Abstract

Here is abstract yes

Resume

Denne rapport beskriver udviklingsprocessen for det fjerde semesterprojekt på ingeniørhøjskolen i Århus for elektro linjen. Projektet omhandler en intelligent bil kaldet AU2 som er en legetøjsbil der kan styres fra en almindelig computer med Windows installeret. Den intelligente del består af at bilen selv er i stand til at måle hvornår den er på vej mod en forhindring og derved selv undvige en kollision, på trods af brugerens styreinput. Brugeren er i stand til at se hvor bilen befinder sig via et kamera som er monteret på bilen, samt se bilens aktuelle hastighed mv. i et program installeret på brugerens computer.

Indhold

Arbejdsopgaver

Under projektarbejdet har arbejdsopgaver været fordelt efter følgende tabel:

	Kristian Thomsen	Philip Krogh-Pedersen	Lasse Barner Sivertsen	Henrik Bagger Jensen	Kenn Hedegaard Eskildsen	Karsten Schou Nielsen	Jesper Pedersen
Projektformulering	X	X	X	X	X	X	X
Kravspecifikation	X	X	X	X	X	X	X
Systemarkitektur	X	X	X	X	X	X	X
Protokol GUI						X	
Protokol Kamera						X	
HW Design og impl. - Bil: Strømforsyning	X						
HW Design og impl. - Bil: Motorstyring							X
HW Design og impl. - Bil: Tachometer							X
SW Design og impl. - Bil: Steering klassen							X
SW Design og impl. - PC: Software						X	
SW Design og impl. - PC: XboxController klassen			X				
SW Design og impl. - Bil: Kamera software						X	
SW Design og impl. - Bil: Log og Data klasserne		X					
SW Design og impl. - Bil: PcCom og Settings klasserne			X				
SW Design og impl. - Bil: Afstandssensor					X		
SW Design og impl. - Bil: Aks og Pi klasserne	X						
Accepttest	X	X	X	X	X	X	X
Rapport	X	X	X	X	X	X	X

1 Forord

Projektgruppen er bestående af i alt 7 mand og alle har bidraget til systemarkitekturen samt deltaget aktivt i oprettelsen af usecasene i aktitekturfasen. Det meste i designfasen blev også lavet af hele gruppen men i implementeringen blev det delt op i mindre grupper eller enkelte personer. Rapporten indeholder derfor sektioner som er skrevet samlet, samt individuelle sektioner. De individuelle sektioner er skrevet ud fra personernes ansvarsområde i implementeringsfasen. Disse ses under Arbejdsopgaver.

2 Indledning

OBS: Fra PRJ3, skal rettes

Denne rapport omhandler udvikling og realisering af en prototype til et system, der kan installeres i et drivhus. Systemet - AutoGreen - hjælper brugeren med at opnå og fastholde optimale forhold for planterne i drivhuset. Systemets vigtigste funktioner er måling og regulering af temperatur, samt måling af jordfugt. Reguleringen af temperatur sker ved hjælp af et varmelegeme, ventilatorer og åbning/lukning af et vindue. Ved manglende fugtighed i jorden gives brugeren besked herom.

AutoGreen er både for den uerfarne bruger, der har brug for hjælp for at sikre sine drivhusplanters overlevelse, men det er også for den mere erfarne bruger, der ønsker optimerede forhold i sit drivhus. Opvarmning af drivhuset medvirker til forlængelse af vækstsæsonen og rettidig vanding af planterne er med til at sikre optimale vækstforhold.

Controlleren og brugerfladen i AutoGreen - realiseret på et DevKit8000 - kommunikerer via UART med et PSoC 4 Pioneer Kit, der agerer I^2C master i systemet. Masteren kommunikerer flere I^2C slaver, der har ansvar for hhv. aktuatorer (varme, vindue og ventilation), måling af temperatur og analoge jordfugtsensorer.

2.1 Læsevejledning

Rapporten er, så vel som projektdokumentationen, opbygget kronologisk, dvs. efter samme rækkefølge som arbejdet er udført. Der er dog den undtagelse at accepttestspecifikationen er udarbejdet i umiddelbar forlængelse af kravspecifikationen, men selve testen er naturligvis først gennemført i slutningen af forløbet, hvorfor dette afsnit er behandlet i slutningen af både rapport og projektdokumentation.

Bemærk at afsnittet Arbejdsopgaver (side v) indeholder en oversigt over hvad de enkelte gruppemedlemmer har haft ansvar for - ikke nødvendigvis hvilke afsnit af rapporten (eller dokumentationen) de har skrevet.

For dokumentationen gælder det, at hvert kapitel har en versionshistorik, hvor der ved hver indtastning er påført et gruppemedlems initialer. Dette betyder udelukkende at det pågældende gruppemedlem har 'siddet ved tasterne', og giver således ikke gruppemedlemmet nogen form for ansvar eller ejerskab af kapitlet.

Ved første øjekast kan denne rapport synes væsentlig længere end de tilladte 30 normalsider. Se [**?**] for detaljeret disposition.

2.2 Ordforklaring

Begreb	**Forklaring**
Plantedatabase	Plantedatabasen indeholder information om ideelle forhold for forskellige typer planter, som brugeren kunne tænkes at plante i sit fysiske drivhus. Informationen i plantedatabasen står til grund for udgangsparametre for nye planter i det virtuelle drivhus. Der findes en række systemplanter, som brugeren ikke kan redigere eller slette, men brugeren kan tilføje egne planter.
Datalog	Systemet er udstyret med en log over de indsamlede data fra sensorer i systemet, og der måles og indskrives i loggen hvert minut. Denne er opbygget som en datastruktur, hvor hver logning indeholder information fra sensorerne samt et tidspunkt.
Systemlog	Systemet er udstyret med en log over hvad systemet foretager sig. Dette kunne f.eks. være et indlæg når systemet foretager en måling, sender en e-mail og regulerer miljøet i drivhuset.
Virtuelt Drivhus	Det virtuelle drivhus er systemets repræsentation af det fysiske drivhus. Brugeren kan tilføje planter fra plantedatabasen i det virtuelle drivhus, og på den måde give systemet indirekte oplysninger om ønskede parametre. Disse informationer lagres i systemets konfigurationsfil.
Fysisk Drivhus	Ved det fysiske drivhus forstås det drivhus, hvori systemet er monteret.
Konfigurationsfil	Dette er en klasse, der er placeret på DevKit8000, som indeholder brugerens konfigurationer om blandt andet notifikationer, e-mailadresser, antallet af fugtsensorer og deres unikke ID mm.
Notifikations e-mail	Dette er en daglig e-mail, som brugeren kan vælge at få tilsendt. Den sendes klokken 12:00, og indeholder informationer om parametrene i det fysiske drivhus.
Advarsels e-mail	Dette er en e-mail, som brugeren kan vælge at få tilsendt. Den sendes, hvis en parameter i det fysiske drivhus kommer uden for tolerancen af den ønskede værdi.

3 Opgaveformulering

OBS: Stjålet fra PRJ3

Herunder er vist en prioritering af funktionaliteter i systemet efter MoSCoW metoden.

Ambitionen for dette projekt var som absolut minimum, at realisere nedenstående punkter under *"skal"*. Det forventedes desuden at punkterne under *"bør"* skulle realiseres, men de har haft lavere prioritet. Punkterne under *"kan"* forventedes ikke realiseret, og punkterne under *"vil ikke..."* realiseredes med sikkerhed ikke. Sidstnævnte punkter kan ses som udviklingsmuligheder i forhold til senere versioner af systemet.

- **Systemet skal:**
  - Kunne monitorere temperaturen i drivhuset og regulere temperaturen i drivhuset vha. varmelegeme, åbning af vinduer og luftcirkulation.
  - Give brugeren mulighed for at vælge varmelegeme og/eller luftcirkulation fra, hvis en mere økonomisk regulering af temperaturen ønskes.
  - Have et grafisk user interface.
- **Systemet bør:**
  - Måle jordfugtighed med op til seks sensorer i drivhuset og give brugeren besked på brugerfladen om, at det er tid til at vande.
  - Måle lysintensitet og luftfugtighed i drivhuset.
  - Indeholde en log over alle målte parametre: Jordfugtighed, temperatur, luftfugtighed og lysintensitet. Dataene præsenteres grafisk for brugeren.
  - Indeholde en database over de mest almindelige drivhusplanter, så brugeren kan orientere sig om en plantes optimale forhold.
  - Indeholde en systemlog, som noterer vigtige system hændelser.
- **Systemet kan:**
  - Sende besked til brugeren via e-mail, om at det er tid til at vande.
  - Tilkobles et automatisk vandingssystem, som aktiveres ved behov for vanding.
  - Give brugeren mulighed for at tilføje planter i databasen.
  - Give brugeren mulighed for at kommunikere trådløst med systemet fra brugerfladen, så denne kan placeres fx inde i brugerens bolig.
- **Systemet vil ikke i denne version:**
  - Indeholde et kamera, og tilhørende billedarkiv, som giver brugeren mulighed for at følge planternes udvikling fra dag til dag.
  - Give brugeren mulighed for at interagere med systemet via en app på dennes mobiltelefon.

For den fulde tekst se afsnit **?? ??** på side **??** i projektdokumentationen.

4 Projektafgrænsning

OBS: Stjålet fra PRJ3

AutoGreen består af en strømforsyning, en række controllere (tre stk. PSoC 4 Pioneer Kits og et stk. DevKit8000), et varmelegeme (en USB strømspareskinne og en 100W 230V AC glødepære), fire 12V DC ventilatorer og en 12V steppermotor - samt tilhørende mosfet drivere. AutoGreen uindeholder desuden sensorer til måling af lufttemperatur, jordfugt, lysintensitet og luftfugtighed.

Selve det fysiske drivhus er således ikke en del af systemet. Under udviklingen af denne prototype er anvendt en model af et drivhus på ca. 33 liter, se evt. Figur **??** på side **??** i Projektdokumenationen. Såfremt prototypen skal monteres i et rigtigt drivhus, skal ventilatorer og varmelegeme dimensioneres derefter.

I de indledende faser af projektarbejdet - Projektformulering, Kravspecifikation, Accepttestspecifikation og Systemarkitektur - omhandler projektdokumentationen det fulde system. Grundet tidsnød og forskellige komplikationer, er der flere dele af det samlede system, som kun er delvist eller slet ikke implementeret. Prioriteringen af hvad der skulle skæres væk undervejs har taget udgangspunkt i MoSCoW prioriteringen, se afsnit 3 Opgaveformulering på side 4.

Alle punkter under "Systemet skal" er fuldt implementeret.

Under "Systemet bør" er det første punkt, vedrørende jordfugtsensorer, fuldt implementeret; øvrige punkter er kun delvist eller slet ikke implementeret.

Punkter under "Systemet kan" og "Systemet vil ikke i denne version" er ikke implementeret.

Det er med fuldt overlæg at denne løbende prioritering har fundet sted. Gruppen ønskede fra starten et system med mange muligheder for udvikling og udbygning; derfor blev der fra begyndelsen beskrevet et system, som var væsentligt mere omfattende, end hvad gruppen regnede med at kunne nå at realisere.

Den gennemførte acceptest (af hele det beskrevne system) giver detaljeret information om hvad der er realiseret, hvad der er delvist realiseret og hvad der ikke er realiseret, se afsnit **?? ??** på side **??** i projektdokumentationen.

5 Systembeskrivelse

OBS: Stjålet fra PRJ3

Det system, der er tænkt realiseret i dette projekt, er en skalamodel af et drivhus med steppermotor til åbning af et vindue og blæsere til udluftning samt et varmelegeme til at varme drivhuset op. Til at måle på drivhuset var det tiltænkt at implementere en temperatursensor, jordfugtmålere samt sensorer til måling af lysintensitet og luftfugtighed. De to sidstnævnte er dog ikke implementeret grundet komplikationer i implementeringsfasen.

Selve systemet styres af et DevKit8000, som brugeren af systemet kan interagere med. På denne platform kører - samtidigt med det grafiske miljø - processer til regulering og monitorering af miljøet i drivhuset. Der er udover dette også mulighed for at tilgå forudindstillede planter i en plantedatabase samt at tilføje og fjerne eksisterende planter i et virtuelt drivhus, som systemet anvender til at afspejle de planter, der eksisterer i det fysiske drivhus. Al aktivitet omkring styringen af systemet logges i en systemlog.

her var et rigt billede

Figur **??** viser et rigt billede af systemet; der ses hvordan det fysiske drivhus påvirkes ved kontrol af varme og luft samt kommunikationen med GUI'en.

her var et billede af det færdige system

På Figur **??** ses et billede af den færdige prototype, og på Figur **??** ses et billede af hovedmenuen på systemets brugerflade.

her var et billede af brugerfladen

6 Krav

OBS: Stjålet fra PRJ3

I dette afsnit beskrives optillede krav for AutoGreen, som er opstillet ud fra opgaveformuleringen. På Figur **??** ses Use Case diagram over systemet. Dette giver et overblik over de funktionelle krav, der er formuleret i dokumentationen på side **??**.

her var et uc diagram

Use Cases på billedet er kort beskrevet herunder:

- UC1: Start
  Denne UC giver brugeren mulighed for at starte systemet, dvs. monitorering og regulering af drivhusklimaet.
- UC2: Stop
  Denne UC giver brugeren mulighed for at stoppe systemet, dvs. monitorering og regulering af drivhusklimaet.
- UC3: Overvåg
  Når UC10 Monitorering er startet, vises der på brugerfladens hovedmenu alle de aktuelle måleværdier. Hvis UC11 Regulering er startet, kan værdierne for lufttemperatur og jordfugtighed være røde, hvis de ikke passer med de ønskede værdier.
- UC4: Administrer Drivhus
  Giver brugeren mulighed for at informere systemet om hvilke planter der er i drivhuset.
- UC5: Se Historik
  Giver brugeren mulighed for at se en grafisk historik over de fire målte parametre i drivhuset.
- UC6: Administrer Plantedatabase
  Giver brugeren mulighed for at se på planter i databasen, samt tilføje og fjerne egne planter i databasen.
- UC7: Konfigurer System
  Giver brugeren mulighed for at rette i systemindstillinger.
- UC8: Se Systemlog
  Giver brugeren mulighed for at se en liste over systemhændelser.
- UC9: Rapportering
  Rapporterer til brugeren ud fra de indstillinger brugeren har valgt. Dette sker ved afsendelse af e-mail til den eller de adresser som er valgt.
- UC10: Konfigurer System
  Lagrer målinger af lufttemperatur, jordfugtighed, luftfugtighed og lysintensitet i en data log.
- UC11: Regulering
  Regulerer temperaturen i drivhuset, vha. vinduesåbner, varmelegeme og luftcirkulation, med mindre brugeren har slået varmelegeme og/eller luftcirkulation fra.

Systemet skal have en grafisk brugerflade, der giver brugeren mulighed for at konfigurere og monitorere drivhuset. På brugerfladen skal brugeren have mulighed for at overvåge den aktuelle temperatur og bør desuden kunne se jordfugt, luftfugtighed og lysintensitet. Disse data logges og bør kunne aflæses på en graf, der viser historik for samtlige parametre. Systemet skal kunne regulere

temperaturen i drivhuset på baggrund af de aktuelle parametre.
Baseret på brugeres præferencer kan systemet advare brugeren via e-mail, hvis systemet fejler eller klimaforholdene bliver kritiske. Til at regulere systemet skal brugeren kunne indstille systemet til at anvende varmelegeme og/eller ventilatorer til at justere klimaet.
AutoGreen bør have en plantedatabase, der indeholder foruddefinerede planter. Planterne kan indsættes i det virtuelle drivhus, og herefter skal systemet kunne regulere klimaet i det fysiske drivhus, så passer bedst til de(n) valgte plante(r). Brugeren skal kunne tilføje, fjerne og redigere planter, som er indsat i det virtuelle drivhus efter behov.

7 Projektbeskrivelse

7.1 Projektgennemførelse

OBS: Stjålet fra PRJ3

Gruppen, som er en videreførelse fra 2. semesterprojekt, har løftet opgaven med fornyet engagement og endnu større handlekraft end tidligere. Fra begyndelsen af projektperioden har arbejdet med projektet været relativt uden problemer. Det har hele tiden været gruppens mål, som på foregående semester, at holde sig foran tidsplanen [**?**], men samtidig have en fornuftig tilgang til arbejdet. Dette har for gruppen betydet en forøget arbejdsindsats i form af anvendelsen af alle fritimer, der var mulige at bruge. Forskellen på dette semesterprojekt og gruppens tidligere, er en meget mere klar opdeling i hardware og software. Opdelingen er kommet som en naturlig konsekvens af opdelingen i uddannelserne E/EP/IKT, og har samtidig betydet øget fokus på de relevante dele af projektet for de individuelle medlemmer.

I projektet er anvendt en kombination af udviklingsmodellerne V-model, Scrum og ASE-modellen, som i en stor blanding, gav muligheden for udarbejdelsen af gruppens projektdokumentation og rapport. Se afsnit 7.2 Metoder på side 11 for nærmere beskrivelse. Modellerne er ikke nødvendigvis fulgt fuldstændigt, men gruppen har efterhånden udarbejdet sin egen fortolkning, som er meget velfungerende. Gennem projektperioden er hvert overemne blevet kørt som et sprint, men det er først i forbindelse med design og implementering at der konkret kan tales om reelle sprint. Eftersom gruppen har valgt at fortsætte fra et tidligere semester, er mange af tingene som blev udarbejdet tidligere, fx samarbejdsaftale, mødeskabeloner og opgaver blevet genbrugt. Genanvendelsen af delelementer har gjort opstarten af projektet en del nemmere, end hvis der skulle startes fra bunden.

Rollerne der er blevet fordelt i projektet ser ud som følger:

- Koordinator
  Morten har haft det overordnede ansvar for administrative opgaver, som mødeindkaldelser, referater og logførelse.
- Ordstyrer
  Philip har været ordstyrer igennem gruppens vejleder- samt arbejdsmøder.
- Dropbox Ansvarlig
  Kristian T. har stået for orden og udlægning af deletjenesten.
- GitHub ansvarlig
  David har været ansvarlig for kildekodedelingen over GitHub.
- SCRUM ansvarlig
  David har været overordnet ansvarlig for anvendelsen af SCRUM.
- Lokale booking
  Kristian S. har været ansvarlig for at booke lokaler når det var nødvendigt.

7.2 Metoder

OBS: Stjålet fra PRJ3

Under Metoder vil de forskellige arbejdsmetoder, der er blevet brugt under dette projekt blive beskrevet. Disse metoder er hhv. V-model, SysML, Scrum, Reviews og Versionsstyring.

7.2.1 V-Model og ASE-Model

Under projektets forløb er V-Modellen fulgt, som en vejledning til udførelsen af projektet. Modellen er dog ikke fulgt fuldstændigt, da der ikke er blevet defineret flere testscenarier ud over den vigtige Accepttest. Dette skyldes, at gruppen har fundet det mere hensigtsmæssigt at lave løbende tests, da der i mange tilfælde har været en vigtig læringsproces i hvilke funktionaliteter, der har kunne lade sig gøre. Derved har det været svært at fastsætte mindre tests imellem de forskellige enheder i tidligere stadier. Det vil sige at tests såsom modultests blev beskrevet og bearbejdet sideløbende med design- og implementeringsfasen.

Ud over V-Modellen **??**, er ASE-Modellen **??** taget i brug som en vejledning til udførelse af projektet. Der er hovedsageligt lagt fokus på at gøre det muligt for HW- og SW-grupper at dele sig op under design og implementering. På baggrund af dette er der lagt stor fokus på at forklare systemets ønskede funktionalitet og kommunikationsveje under systemarkitektur. Opdelingen har gjort arbejdet mere effektivt, da det har givet den enkelte mulighed for mere fordybelse til at arbejde med et specifikt område.

7.2.2 SysML

SysML har været medvirkende til at give overblik over projektet, da systemet har kunnet deles op i blokke, og det herefter var muligt at arbejde med disse individuelt. Ud fra disse blokke var beskrivelsen af parts og ports nemt. BDD-diagrammer har givet overblik over komposition af blokkenes relationer, som er specificeret i diagrammet. IBD-diagrammer har givet mulighed for at holde styr på signaler og kommunikationsveje mellem de forskellige blokke. Signalerne, der går imellem blokkene, gav mulighed for at lave detaljerede grænseflader på systemets elementer. UART protokollen er systemets vigtigste grænseflade, da den definerer grænsen mellem HW og SW gruppen. Use Cases har givet mulighed for at designe det ønskede scenarie, og tage højde for de faldgrupper, der kan opstå undervejs i scenariet. Use Cases er blevet anvendt til at fremstille sekvensdiagrammer, så de stemmer overens med udførelsen af de enkelte steps i use casen.

7.2.3 Scrum

Scrum er primært brugt af SW gruppen. Scrum blev brugt til uddeling af opgaver under design af SW til DevKit8000. En webbaseret løsning er brugt som scrum-board, i stedet for et fysisk scrum-board, da et fast grupperum ikke har været til rådighed. Der blev i SW gruppen brugt en form for daglige scrum-møder, hvor der hurtigt kunne gennemgås status på individuelle opgaver, og om der var forekommet nogle problemer, der kunne være svære at løse. Herefter kunne scrum-boardet opdateres med evt. nye opgaver. Det gav et godt overblik, og alle havde altid adgang til at kunne se, hvad der kunne laves som det næste, når en opgave var løst. Det blev dog valgt i SW gruppen ikke at bruge scrum-boardet under implementeringsfasen, da der blev holdt daglige møder, og det ikke føltes som en nødvendighed at skulle holde styr på alle opgaverne vha. Scrum, når der kun var 3 personer i teamet. Der blev dog fortsat holdt fast på de daglige scrum-møder.

7.2.4 Versionsstyring

Versionerhistorik på dokumenter i projetdokumentationen er blevet opdateret løbende bla. i forbindelse med kommentarer fra vejleder og reviews. Væsentlige ændringer i fx design har givet anledning til versions-ændring, hvilket hjælper med at holde styr på hvilke ændringer projektet har gennemgået.

7.2.5 Reviews

De reviews der er modtaget igennem projektet, [**?**] og [**?**], har været en stor hjælp til retning og tilføjelser til dokumentationen. De afgivne reviews har været med til at give ideer til ændringer af dokumentation og eget projekt. De afleverede review er rettet med henblik på, hvad der er svært at forstå eller dårligt beskrevet. Dette gør reviewet objektivt. Der medtages ikke forslag til, hvordan man kunne ændre projektet, da det ikke er reviewernes opgave at komme med løsninger til modtagers problemer. Ved modtagelse af review er der holdt en neutralt tilgang. Fokus er lagt på at få så meget som muligt ud af de kommentarer, der modtages. Disse kommentarer har herefter kunnet diskuteres på et efterfølgende internt møde.

7.3 Specifikation og Analyse

OBS: Stjålet fra PRJ3

Dette afsnit omhandler specifikation og analyse i forbindelse med projektets opstillede krav, samt bearbejdelsen og tankerne bag udarbejdelsen af kravspecifikationen (se **??** **??** side **??** i projektdokumentationen).

Der blev foretaget undersøgelser af de forskellige krav, der blev opstillet til projektet, for at sikre at kravene faktisk kunne opfyldes. Der blev efterfølgende diskuteret hvorvidt forskellige sensorer og aktuatorer skulle implementeres i systemet. Det resulterede i, at systemet skulle indeholde en temperatur-, jordfugt-, luftfugt- og en lysintensitetssensor.

Til styring af systemet blev det valgt at bruge DevKit8000 som embedded system, og PSoC 4 Pioneer Kits til at styre hardwaren med. Det blev bestemt at bruge en PSoC Master, som skulle have forbindelse til DevKit8000 igennem UART. UART blev valgt som kommunikationsvej, da der allerede var kendskab til UART. Valget af UART gjorde fejlfinding nemt, da det var muligt at teste ved brug af en PC. PC'en kunne læse, hvad der blev sendt, og skrive tilbage ved brug af tastaturet på PCen. Til kommunikation mellem PSoCs blev det besluttet at bruge I^2C jf. projektoplægget. Beslutningen om at buskommunikationen skulle være af typen I^2C blev taget på baggrund af flere forskellige faktorer. Modsat SPI er I^2C i stand til at sende data over relativt lange afstande. I^2C interfacet har desuden indbygget sikkerhed i standarden.

Til regulering af temperatur i drivhuset blev det valgt, at et vindue skulle kunne åbne og lukke, og fire ventilatorer skulle udskifte luften i drivhuset. Formålet med disse er at køle drivhuset ned. Der blev til opvarmning valgt en glødepære, som en simpel måde at opvarme drivhuset på.

Efter de fleste overordnede hardware beslutninger var blevet taget, blev der lavet en overordnet plan for den generelle funktionalitet og hvordan den grafiske brugerflade skulle se ud.

Der kunne efter de generelle beslutninger, skrives use case diagrammer over de ønskede processer og funktionaliteter, og derved give et godt overblik til at opstille endelige krav. Disse krav blev inddelt i funktionelle og ikke-funktionelle krav, således at de valgte arbejdsmodeller blev fulgt, og det var muligt at opstille en endelig kravspecifikation under processen. Dette endte ud i, at en accepttestspecifikation (se afsnit **??** **??** side **??** i projektdokumentationen) var mulig at udarbejde, dermed blev V-modellen fulgt.

7.4 Systemarkitektur

OBS: Stjålet fra PRJ3

I systemarkitekturen beskrives grænseflader for systemet og hvilke blokke det består af. Til at beskrive dette er der anvendt en række BDD'er og IBD'er. Nedenfor er de vigtigste af disse vist. For mere detaljeret beskrivelse se afsnit **??** **??** på side **??** i projektdokumentationen.

her var et overordnet bdd for system

På Figur **??** kan der skabes et overblik over systemet og hvilke underblokke det består af. Det kan ses, at systemet består af syv underblokke, bl.a. PSoC Master, DevKit8000 mm. I blokken System, der består af alle de andre underblokke, vises de porte som hele systemet har, dvs. grænsefladen til omverdenen.

her var et IBD over signaler

På Figur **??** vises alle signalerne i systemet, dvs. alle spændingsforsyninger og referencer er udeladt for overskuelighedens skyld. For at beskrive de interne signaler, tages der udgangspunkt i DevKit8000. DevKit8000 spørger løbende PSoC Master om temperatur, luftfugtighed, lysintensitet samt jordfugtighed over UART, denne er detaljeret beskrevet på side **??** i projektdokumentationen. Kommunikationen mellem PSoC Master, Jordfugt, Temp/Luftfugtighed, Lyssensor og Aktuator foregår over en I^2C bus. Via denne kommunikationsvej kan PSoC Master'en efterspørge alle sensorværdierne og aktivere aktuatorer, hvis DevKit8000 ønsker at regulere klimaet i drivhuset med varmelegemet, vinduet og/eller blæserne. Der kan ses en signalbeskrivelse, hvor alle signaler mellem hver blok er beskrevet i Tabel **??** på side **??** i dokumentationen.

her var et UML diagram for au2green

På Figur **??** ses et UML klassediagram, som viser relationer mellem klasserne på DevKit8000. Der anvendes to relationstyper; komposition og association. Domain klassen datalog har til opgave at gemme de sensordata som monitor opsamler, jf. Listing **??** på side **??** i dokumentationen. Regulatoren anvender denne information og bruger den til at afgøre, om forholdene i drivhuset er som ønsket. DevKit8000 er som angivet en controller klasse, og derfor binder den de øvrige klasser sammen og har den overordnede styring.

her var en figur over menuer

Menuoversigten, der ses på Figur **??**, gives et samlet overblik over hvordan de forskellige menuer tilgås igennem systemet. Systemet viser altid hovedmenuen, når systemet starter op. Herfra er det muligt at få et overblik over drivhusets aktuelle klima. Hovedmenuen viser desuden fem knapper, hvor man kan få adgang til undermenuerne: virtuel drivhus-, historik-, plantedatabase-, systemlog- og konfigurationsmenu.

7.5 Hardware Design

Svissen svassen

7.6 Software Design

7.6.1 GUI

I arbejdet med GUI'en var det første der skulle findes ud af, hvilket sprog og udviklingsmiljø softwaren skulle skrives i. Henrik og Karsten gik begge i gang med at undersøge hver sit. Henrik undersøgte mulighederne i C# og visual studio. Han fik hurtigt en GUI op og stå, hvori der kunne modtages videostream. Karsten gik i gang med C++ og Qt creator, her tog det dog længere tid at finde en løsning til at modtage videostream, men det var et kendt sprog og det var nemt at designe det grafiske. På et møde samlede begge deres erfaringer og det blev i gruppen besluttet at køre videre med Qt. Beslutningen blev argumenteret med at vi senere ville få faget ISU, hvor vi ville lære om tråde og deres kommunikation indbyrdes i C++. Det ville derfor være en fordel at kunne drage nytte af det i projektet. Henrik havde desuden fundet et bibliotek til at håndtere Xbox360-controlleren skrevet i C++. Dette ville ikke kunne inkluderes i C#. I selve arbejdet med GUI'en var det som før beskrevet en udfordring at finde en løsning til at modtage videostream i. I Qt creator er der en indbygget webbrowser som kan trækkes ind i den grafiske del af GUI'en. Denne startes nemt i constructoren af koden, hvor der kan gives en URL med ved start. Men desværre virkede det ikke med et videostream og det blev derfor undersøgt om det var muligt at tilføje en tilføjelse til browseren, som det kendes fra fx Chrome eller Firefox, osv. Dette var muligt, men blev hurtigt så kompleks at det ville være nemmere at bruge open source biblioteker fra VLC mediaplayer. Selvom at denne løsning umiddelbart virkede simplere, viste det sig dog at være ligeså tidskrævende. Der blev fundet flere guides på nettet, men disse var alle fyldt med fejl, da der enten var døde links eller henvisninger til biblioteker som ikke længere var tilgængelig. Det var derfor en sammensætning af dem alle som der gjorde det muligt at komme videre. I dokumentationens litteraturliste findes linket [**?**] der er brugt som udgangspunkt. Da arbejdet med GUI'en startede før faget ISU, var der på det tidspunkt ikke nogen der viste noget om tråde. Det tog derfor et stykke tid at finde ud af hvordan disse skulle implementeres og hvordan der sendes signaler fra den ene tråd til den anden. Der blev fundet en bog på nettet om hvordan man programmerer i QT [**?**] s. 13-37 & 381-396. fra denne er der fundet inspiration til at løse problemet med tråde samt signals og slots. Da projektet var ved at nærme sig aflevering var der et problem som der manglede en løsning på. Problemet er at når der er oprettet forbindelse til bilen og controlleren er forbundet, ligger der to TCP-forbindelser i hver sin tråd at kører. Hvis bilen kommer uden for rækkevidde vil GUI'en miste forbindelsen til bilen og GUI'en crasher. Dette skyldes at når en TCP-socket kører i en tråd, kan den ikke sende signaler til hovedvinduet som ellers skal sørge for at der ikke længere sendes data når forbindelsen er tabt. Programmet crasher derfor når den skriver til en forbindelse som ikke længere eksisterer.Problemet er også beskrevet i dokumentationen.

7.6.2 Video stream fra bilen

Sammen med arbejdet på GUI'en blev der arbejdet på Pi'en med at få et kamera installeret og et video-stream op at stå. Først blev der lånt et kamera på skolen som det viste sig at der ikke var skrevet en driver til. Derfor blev det valgt at købe et Raspberry-pi-rev-2.0 Kamera som der både var skrevet driver til samt det passede i kamera porten på Pi'en. Programmet motion blev installeret på Pi'en men det viste sig nu at motion kun kunne tilgå enheder belliggende i folderen /**dev**/. Der skulle derfor findes på en anden løsning. En af dem var at installere en modificeret version af motion [**?**] som godt kunne tilgå kameraet. Desværre var der ingen af disse versioner som der kunne bruges, da flere biblioteker som motion skulle bruge, ikke længere eksisterede eller havde skiftet navn. Selv om at pakkerne godt kunne findes under nogle nye navne, kunne motion stadig ikke finde dem. Der blev fundet frem til at der kunne installeres en virtuel driver [**?**] som gør at kameraet nu

findes i /**dev**/. Den oprindelige verison af motion som downloades fra apt-get kan derfor godt finde kameraet og oprette et video-stream.

7.7 Hardware Implementering

Write me

7.8 Software Implementering

write me

7.9 Resultater og Diskussion

OBS: Stjålet fra PRJ3

Overordnet er alle "skal" krav blevet opfyldt for AutoGreen projektet. Det vil sige, at systemet kan overvåge temperaturen, samt regulere den på baggrund af måleværdier. Systemet giver mulighed for at vælge om varmelegeme og/eller blæsere skal være aktive under regulering, og AutoGreen indeholder en grafisk brugerflade udviklet i QT. AutoGreen indeholder desuden nogle "bør" funktionaliteter, da brugeren har mulighed for at overvåge op til seks planters jordfugtigthed i hovedmenuen.

Lysintensitet- og luftfugtighedssensorer er blevet droppet sent i forløbet, da det viste sig at der var problemer med at få dem til at fungere. Systemet har en fungerende datalog over alle måleværdier, dog er der ikke implemeneteret en grafisk fremstilling af dem. Ligeledes er plantedatabasen ikke implementeret. Systemloggen virker, men kan kun vise sidste hændelse, dvs. den er ikke implementeret færdig. System opfylder kun et enkelt "kan" krav, nemlig mulighed for tilslutning til et automatisk vandingssystem, ellers opfylder AutoGreen ikke flere "kan" krav.

En af de ting som virker overraskende godt i AutoGreen, er selve reguleringen af temperaturen. Vi havde forventet at få problemer med at kunne regulere temperaturen med en præcision på +/- 2 °C, men det er på grænsen til at det lader sig gøre med +/- 0,5 °C, og så er det pludselig opløsningen på temperatursensoren, der sætter begrænsningen.

Det gode resultat grunder i et sammenfald af flere ting. For det første er aktuatorerne passende dimensioneret i forhold til drivhusets størrelse. Der ligger ikke dybe tanker og en masse beregninger bag dette. Der var monteret fire ventilatorer i drivhuset da vi overtog det, hvilket syntes en smule voldomt; derfor kører de med en duty cycle på 50% for at undgå overshoot i forbindelse med køling. De 50% var intet mere end et gæt, som viste sig at være fornuftigt. Vi indkøbte i starten af forløbet 3 stk. 100W glødepærer til at bruge som varmelegeme, men det viste sig hurtigt, at en enkelt var passende, for at undgå overshoot i forbindelse med opvarmning.

Der var fra begyndelsen desuden lagt op til at udvide med PWM styring og PID-regulering af varmlegeme og ventilation under hhv. opvarmning og afkøling af drivhuset. Dette blev bla. pga. tidsnød ikke implementeret, så vi regnede fx med at varmelegemet ville komme til at stå og tænde og slukke, når temperaturen i drivhuset nåede det ønskede nivaeu. Dette er også tilfældet, men det sker meget langsommere end vi havde forventet, da glødepæren ikke bliver kold i det samme øjeblik den slukkes; derfor falder temperaturen kun langsomt.

UART kommunikationen mellem DevKit8000 og PSoC Master kom desværre til at fungere dårligere end forventet. Der er en del fejlkommunikation, som primært kommer til udtryk ved udfald på jordfugtsensorerne. Vi mener dette kan skyldes problemer med timing i SW på hhv. DevKit8000 og PSoC Master, da meget af koden er interruptbaseret, der afvikles flere strenge samtidigt og der skal ventes på retursvar, når der spørges efter en sensorværdi eller sendes en kommando til en aktuator. Problemet kan også skyldes decideret fejl på selve den fysiske UART kommunikation. Problemet kan muligvis afhjælpes ved at anvende skærmet kabel mellem DevKit8000 og PSoC Master, eller mere sandsynligt ved at anvende en komplet UART med alle 9 forbindelser i stedet for AutoGreen's mere skrabede model, der kun indeholder Tx, Rx og en reference. Alternativt skal design af SW på PSoC Master og/eller DevKit8000 skrives helt om. Under alle omstændigheder ligger her et oplagt udviklingspotentiale for systemet.

7.10 Udviklingsværktøjer

OBS: Stjålet fra PRJ3

I dette afsnit vil der blive gennemgået de forskellige udviklingsværktøjer, som er blevet anvendt under dette projekts design-, implementerings- og integrationsproces.

7.10.1 PSoC Creator

Til programmering af projektets PSoC 4 Pioneer Kits blev PSoC Creator udnyttet. Programmet er anvendt til kodning og debugging af PSoCs under implementering af software. Dette udviklingsmiljø har et stort bibliotek af klargjorte komponenter og kode hertil, hvilket gør processen af programmering af PSoCs effektiv.
I dette projekt er der valgt at bruge PSoC 4 processoren på de tre Pioneer Kits, det er dog muligt at bruge PSoC 5 processoren i stedet, hvis dette ønskes, da denne processor har mange flere funktionaliteter. Grunden til at PSoC 5 komponenten ikke er valgt i dette projekt er, at der ikke har været brug for den funktionalitet PSoC 5'eren har, frem for PSoC 4'eren. Ved at bruge PSoC 4, kan PSoC 5 processoren ligeledes bruges til at debugge direkte på et PSoC 4 projekt, hvilket har gjort fejlfinding af PSoCs effektiv.

7.10.2 Multisim

Gruppen har valgt Multisim til design af hardware. Styrkerne ved Multisim er at skabe et overblik og muligheden for at simulere forskellige hardware moduler, med de ønskede komponenter, fra et rigt bibliotek. Svaghederne ved Multisim er, at det nogle gange ikke er muligt simulere et kredsløb korrekt samt i værste tilfælde, at Multisim ikke har mulighed for at simulere en specifik hardware komponent. I dette projekt er Multisim anvendt under design og implementeringsprocessen for hardware.

7.10.3 Ultiboard

Ultiboard blev brugt til at tegne og konstruere printplade layouts. Ultiboard kan, i forbindelse med Multisim, uddrage de komponenter der findes i designs fra Multisim og konstruere printplade layouts ud fra disse.

7.10.4 QT Creator

QT Creator er brugt til designe GUI'en til DevKit8000. Programmet har en del funktionalitet, der gør at de GUI programmer man kan lave i programmet også kan køre på andre OS platforme fx Windows eller Linux. Dette er dog også problematisk, da denne funktionalitet kommer på bekostning af den normale implementering. Dette ses i deres interne primitive typer, som fx er string kaldet en Qstring, så alle normale strings skal castes til Qstring for at kunne bruges i deres UI klasser.

7.10.5 Git

Git er et versionstyringsværktøj til at vedligeholde kildekode. Git er et stærkere værktøj end SVN, der blev brugt i sidste semester projekt. Det er dog mere kompliceret at bruge pga. de flere valgmuliger som medfølger. Som repository host er der valgt Github grundet stabilitet, da gruppen oplevede problemer med RiouxSVN sidste semester.

7.11 Opnåede Erfaringer

7.12 Fremtidigt Arbejde

OBS: Stjålet fra PRJ3

Umiddelbart er de resterende krav, som ikke blev implementeret i projektforløbet, oplagte kandidater til fremtidigt arbejde. Dette inkluderer måling af lysintensitet og luftfugtighed, mulighed for at tilføje planter i det virtuelle drivhus, interaktion med plantedatabase, grafer for samtlige måledata, e-mail notifikationer samt fuld systemlog. For at e-mail notifikationer skal fungere ville det også være nødvendigt at implementere internetadgang på DevKit8000, hertil kan det tilføjes at mulighed for trådløst netværk ville være oplagt, da systemet skal sidde i brugerens hjem og ikke nødvendigvis være i nærheden af router el. lign., og man ville slippe for at trække ethernet kabel til sit drivhus. Man kunne også overveje en eller anden form for trådløs kommunikation mellem DevKit8000 og PSoC 4 Master; derved undgår man at brugerfladen skal placeres i drivhuset.

Udover disse opgaver er oplagte muligheder integration med et vandingssystem, så drivhuset er selvstyrende mht. vanding. Dette kunne gøres ved at åbne/lukke magnetventiler for hver plante i drivhuset. For at det ville fungere, bør der også implementeres en form for kalibrering af vandtilførslen i forhold til systemet, så en eventuel vandingsalgoritme ville kunne justere vandmængden fornuftigt.

Under projektforløbet har gruppen indset, at PSoC Master-blokkens funktionaliteter kunne integreres i DevKit8000 og herved spare en hel hardwareblok, da denne har mulighed for I^2C direkte på kittet. Der er ligeleds også fordele i at lade den digitale filtrering foregå på DevKit8000, da denne har langt flere resourcer tilgængelige i forhold til digital filtrering i software end PSoC 4. En anden mulighed ville være at lade den digitale signalbehandling foregå på PSoC 5, der også forefindes på PSoC 4 Pioneer Kit. Denne er dog upraktisk at udvikle på, da den ikke tilbyder samme debugging funktionaliteter som PSoC 4 processoren.

Gruppen har under forløbet også luftet tanken om at implementere en mobilapp til monitorering og interaktion med systemet, denne ville fx kunne implementeres til Android og iOS. En sådan app ville kunne give brugeren information og mulighed for interaktion med systemet når brugeren ikke er hjemme i en længere periode, som fx på ferie. Brugeren ville ligeledes også kunne anvende appen udelukkende i stedet for touchskærmen på DevKit8000.

Med hensyn til selve hardwaren i drivhuset ville det være oplagt at integrere aktuator og jordfugtmåler på én PSoC 4 og placere denne på samme print som strømforsyning. Dette ville reducere prisen på produktet væsentligt samt skabe mere stabilitet ift. udefrakommende påvirkninger (elektromagnetisk stråling og mekanisk påvirkning). Ligeledes ville det være en fordel at indpakke alle hardwareenheder i passende kabinetter.

8 Konklusion

OBS: Stjålet fra PRJ3

Hvis man kun ser på afsnit **??** **??** på side **??** i projektdokumentationen, kan det umiddelbart se ud som de opstillede krav for projektet langt fra er opfyldt, men hvis man sammenholder accepttesten med MoSCoW prioriteringen i afsnit **??** **??** på side **??** i projektdokumentationen, ses det, at de vigtigste krav er opfyldt.

At de mange uopfyldte krav overhovedet blev formuleret i de tidlige faser af projektarbejdet, var med fuldt overlæg. Gruppen ønskede at arbejde med et stort og realistisk projekt, og ville hellere skære arbejdsopgaver væk undervejs end løbe tør for stof at arbejde med. Der er gennem projektarbejdet løbende lavet bagudgående rettelser i dokumentationen efterhånden som arbejdet skred frem, men gruppen har valgt at fastholde alle krav i projektformuleringen.

En stor del af de krav som ikke er opfyldt, er delvist opfyldt. Det skyldes at gruppen som udgangspunkt har sat ambitionerne højt, og så har vi skåret ned undervejs. Der er fx blevet arbejdet med både lysintensitets- og luftfugtighedssensorer, men komplikationer med disse betød, at de blev droppet sidst i forløbet. Der er dog designet og implementeret SW for sensorerne, så der kræves ikke meget mere arbejde for at kravene vedrørende dette er opfyldt. Af andre eksempler kan nævnes sytemlog og plantedatabase, der er delvist implementerede, men af hensyn til tidsplanen ikke blev gjort fuldstændig færdige.

Det implementerede systems største udviklingspotentiale ligger i UART kommunikationen mellem DevKit8000 og PSoC Mater, se 7.9 Resultater og Diskussion på side 23 for nærmere diskussion af dette.

I de sidste faser af projektarbejdet er der i gruppen internt blevet talt en del om, at det nærmest er ærgerligt at forløbet er slut. Der er mange funktionaliteter som kunne færdigimplementeres og/eller optimeres, hvis der havde været mere tid til rådighed.

Gruppen har undervejs været meget tilfreds med projektarbejdets forløb; de erfaringer gruppen gjorde sig på sidste semester har gavnet forløbet, og gruppen har formået at udvikle sig yderligere. Der er ingen tvivl om at der er store fordele ved at arbejde sammen i en "gammel" gruppe, der ikke først skal til at lære hinanden at kende både socialt og fagligt.

Gruppen er meget tilfreds med det realiserede system, men der er bred enighed om at gruppens største styrke ligger i planlægning og koordinering af arbejdet. Der blev - som på sidste semester - lagt en stram tidsplan fra start; der var lagt op til en periode på tre uger til skrivning af denne rapport. Tidsplanen kom - som forventet - til at skride undervejs, men gruppen som helhed har undervejs formået at have overblik over arbejdet og rette i tidsplanen og kravene for projektet. Derved har vi undgået at skulle lave makværk og lappeløsninger i slutningen af forløbet.

Litteraturliste

[1] Timll Technic Inc: *DevKit8000 brugermanual*.
"Bilag 001 - DevKit8000 User Manual En". 2009.