



3. SEMESTER PROJEKT
RAPPORT

UDVIKLING AF ET BLODTRYKMÅLESYSTEM
Projektrapport

Navn	AU ID	Studienummer
Caroline Kaagaard Dahl Laursen	AU572444	201611025
Nicolai Bæch	AU580049	201704646
Mathias Egsgaard	AU590400	201705031
Thea Plenus Kjeldahl Kristensen	AU577124	201707180
Sarah Krohn Fenger	AU577425	201707931
Mikkel Rugholm Boisen	AU578833	201708119
Kajene Elakanathan	AU594051	201710472

VEJLEDER: SAMUEL ALBERG THRYSSØE

DATO: 19/12-2018

ANTAL SIDER: XX

ANTAL ANSLAG: XX

Resumé

Formålet med dette projekt er at designe og udvikle et invasivt blodtryksmålesystem, der består af en brugervenlig grænseflade. Projektet omfatter også arbejdet med at have et system, der skal kunne kalibrere, kontinuerligt vise blodtryk, forstyrret hjertefrekvens. Systemet skal kunne alarmere hvis blodtrykket stiger eller falder. Derudover består programmet af et digitalt filter til filtrering af blodtrykket.

Gruppen besluttede sig for at designe et system, der passer ind i en operationsstue. Denne brugssituation har haft stor indflydelse på gruppernes beslutningstagning om design og funktioner samt hvordan arbejdet er prioriteret. Brugeren af blodtrykssystemet, som er den sundhedsfaglig personale, kan også via grænsefladen gemme og hente data fra en privat database. Arbejdet har været prioriteret baseret på MoSCoW-modellen [X]. Denne prototype opfylder derfor de foreskrevne must-have kriterier vedrørende: farve størrelse, størrelse af værdier og mm.

Abstract

The purpose of this project is to design and develop an invasive blood pressure measurement system, that consists of a user-friendly interface. The project also includes the work of having a system that should be able to calibrate, continuously display blood pressure and measure heart rate. The system will be able to alert with an alarm if the blood pressure rises or falls. In addition, the program consists of a digital filter for filtering the blood pressure.

The group decided to aim at designing a system that fits inside an operating theater. This user-situation has made a big impact on the groups' decision-making regarding design and functions, as well as how the work has been prioritized. The user of the blood pressure system, who is the healthcare staff, can through the interface save and retrieve data from a private database. Work has been prioritized based on the MoSCoW-model [X]. This prototype therefor fulfills the prescribed must-have criteria regarding: color size, size of values and ect.

Indholdsfortegnelse

	Side
Kapitel 1 Ordliste	6
Kapitel 2 Forord	7
Kapitel 3 Indledning	8
Kapitel 4 Problemformulering	9
Kapitel 5 Teori	10
Kapitel 6 Udviklingsproces	12
6.1 Metode	12
6.1.1 Samarbejdsaftale	13
6.1.2 Udviklingsforløb	13
6.1.3 Arbejdsfordeling	14
6.2 Programmer	15
Kapitel 7 Systembeskrivelse	16
7.1 Systemoversigt	16
7.2 Systembeskrivelse	16
7.2.1 Alarmer	17
Kapitel 8 Krav	18
8.1 Krav	18
8.2 Use Case diagram	19
8.3 Aktørbeskrivelser	19
8.4 Use Case beskrivelser	20
8.4.1 Use Case 1: Mål og vis blodtryk og puls	20
8.4.2 Use Case 2: Justér grænseværdier	20
8.4.3 Use Case 3: Alarmering	20
8.4.4 Use Case 4: Gem data	20
8.4.5 Use Case 5: Kalibrering	21
8.5 Afgrænsning	21
Kapitel 9 Arkitektur	22
9.1 Hardwarearkitektur	22
9.1.1 BDD	22
9.1.2 IBD	23
9.2 Softwarearkitektur	24
9.2.1 Overordnet domænemodel	24

9.2.2 Domænemodel for software	24
Kapitel 10 Design	28
10.1 Hardwaredesign	28
10.1.1 Forstærker	28
10.1.2 Subtractor	29
10.1.3 Anti-aliaserings filter	30
10.1.4 Printplade	30
10.2 Softwaredesign	31
Kapitel 11 Implementering og test	34
11.1 Hardwareimplementering	34
11.2 Softwareimplementering	35
11.3 Test af hardware	36
11.3.1 Modultest	36
11.4 Test af software	39
Kapitel 12 Resultater	40
12.1 Resultater	40
12.2 Diskussion af resultater	42
Kapitel 13 Konklusion	43
Kapitel 14 Fremtidigt arbejde	44
Kapitel 15 Bilag	46
Litteratur	47

1 Ordliste

Ord	Forklaring
AV-klap	Atrioventrikulærklap
GUI	Graphical User Interface
Ext	Extension
VS	Microsoft Visual Studio
AD-converter	Analog-Digital Converter
CPR	Centrale personregister
MoSCoW	Must have, should have, could have, won't have
FURPS+	Functionality, usability, reliability, performance, supportability
BDD	Block Definition Diagram
IBD	Internal Block Diagram

Tabel 1.1. Ordliste

2 Forord

Denne projektrapport er udarbejdet af gruppe 5 bestående af: Kajene Elankanathan, Thea Plenus Kjeldahl Kristensen, Nicolai Bæch, Sarah Krohn Fenger, Mikkel Rugholm Boisen, Mathias Egsgaard og Caroline Kaagaard Dahl Laursen. Vi er alle studerende på diplomingeniøruddannelsen sundhedsteknologi på 3. semester.

Til projektet har vi haft vejleder Samuel Alberg Thrysøe. Samuel har været ansvarlig for den generelle projektvejledning, herunder alle vejledermøder. Alle spørgsmål vedrørende projektet som helhed og spørgsmål til processen er besvaret af Samuel.

Derudover har vi haft mulighed for vejledning ugentligt til både software- og hardware. Thomas Nielsen har været ansvarlig for hardwarevejledningen og Jesper Rosholm Tørresøe har været ansvarlig for softwarevejledningen. Vi har i gruppen også fået hjælp af Lars Mortensen når de andre vejledere ikke har været tilgængelige.

Det færdige projekt til aflevering vil bestå af en projektrapport samt en række bilag. Projektrapporten og opstillingen af bilag er udført efter vejledningen "Vejledning til udfærdigelse af projektrapporter", som findes på studieportalen.

Projektet afleveres inden d. 19-12-2018 kl 12.00. Projektet forsvares til en gruppeeksamen der finder sted tirsdag d. 22-01-2019 kl. 09.00.

3 Indledning

Vores fornemste opgave som sundhedsteknologer er at lave udstyr, som kan hjælpe fagpersoner i sundhedssektoren. Især på sygehuse er der brug for mange sundhedsteknologier, som vi skal være med til at udvikle og gøre bedre. I daglig klinisk praksis er der ofte behov for kontinuert at monitorere patienters blodtryk invasivt, i særdeleshed på intensive afdelinger samt operationsstuer, hvor blodtrykket er en vigtig parameter til monitorering af deres helbredstilstand. Formålet med dette projekt har været at udvikle et system, der kan opfylde netop dette behov. Vi har valgt at arbejde med projektet i forhold til den valgte brugssituation på en operationsstue. Da systemet skal kunne bruges på operationsstuer, stiller dette selvfølgelig mange krav, da det er livsnødvendigt, at systemet er funktionelt og intuitivt at bruge. Systemet består både af hardware og software, som i sammenspil kan måle, processere og visualisere blodtryk samt puls på en computerskærm. Monitorering af blodtrykket invasivt bruges på operationsstuer og intensive afdelinger verden over. Derfor er standarden ISO 60601-1-8:2007 udarbejdet, der omhandler generelle krav, prøvninger og vejledninger for alarmsystemer i elektromedicinsk udstyr. Vi har valgt at implementere uddrag af standarden ISO 60601-1-8: 2007. De dele af standarden, vi har valgt at implementere kan læses i dokumentet alarmbeskrivelse.

Projektet har givet os mulighed for at anvende vores viden fra fysiologi-kurser, elektronik-kurser, signalbehandlingskursus og ikke mindst programmeringskurser. Teoretisk viden fra disse kurser har dannet grundlag for, at vi har været i stand til at designe og udvikle systemet.

Når man skal måle blodtryk, skal man først og fremmest bruge en tryktransducer, som mäter signalet. Dette analoge signal skal behandles ved hjælp af hardware inden det bliver digitaliseret. Når signalet er digitaliseret, kan det bruges i en software applikation, som sørger for at lave beregninger på signalet, visualisere det, alarmere hvis der er pludselige ændringer i og kontinuerligt give et overblik over patientens status. Dette leder frem til den konkrete problemformulering, som vi har arbejdet ud fra.

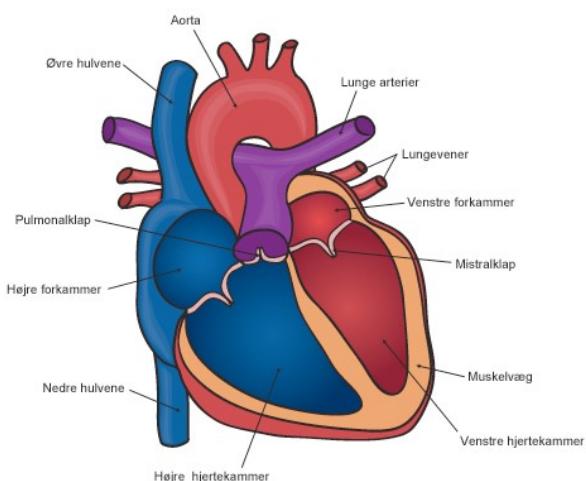
4 Problemformulering

Projektet har til formål at udvikle et invasivt blodtryksmålesystem, der kan tilsluttes patienters arterier via et væskefyldt kateter. Dette målesystem indeholder to elementer.

- Et elektronisk kredsløb, som forstærker signalet fra tryktransduceren og filtrerer det med et analogt filter for at undgå aliasering
- Et program til at vise blodtrykket som funktion af tiden. Programmet skal opfylde en række obligatoriske krav. Det skal:
 - Programmeres i C#
 - Kunne kalibrere blodtrykssignalet og foretage en nulpunktsjustering
 - Vise blodtrykssignalet kontinuert
 - Kunne lagre de målte data
 - Kunne filtrere blodtrykket i selve programmet via et digitalt filter, dette skal kunne slås til og fra (monitor = filtreret og afrundet signal/diagnose mode = råt signal med alle udsving)
 - Kunne afbilde systolisk/diastolisk blodtryk/middel blodtryk med tal
 - Kunne alarmere hvis blodtrykket overstiger indbyggede grænseværdier
 - Kunne måle og afbilde puls

5 Teori

Hjertet består af to halvdele og er adskilt af en kraftig skillevæg. Ud over de to hjertehalvdele består hjertet også af andre elementer, her vigtigt at nævne hjerteklapperne, som sørger for, at blodet kun kan løbe én vej. Hjertet har fire hjerteklapper: To AV-klapper, aortaklappen og pulmonalklappen. Dette ses også på figur 1. Hjerteklappernes åbning og lukning bestemmes af trykket på henholdsvis den ene og den anden side. Højre side af hjertet pumper blod ud i lungekredsløbet via lungearterierne, også kaldt det lille kredsløb, mens venstre side af hjertet pumper blod ud i legems kredsløbet via aorta, også kaldt det store kredsløb. Hjertets pumpefunktionen er en vigtig funktion, der har til hovedopgave at transportere blod rundt i kroppen.



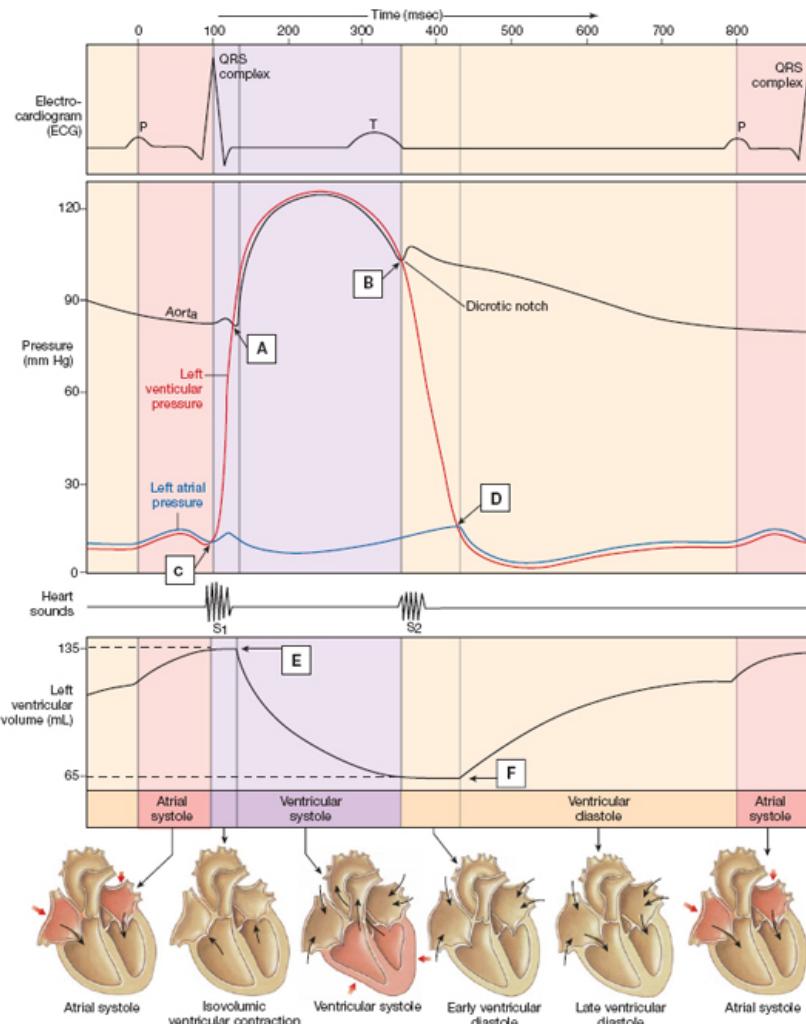
Figur 5.1. Figur af hjertets opbygning [1]

For at hjertet kan transportere blodet rundt til hele kroppen kræver det et tryk, der dannes ved, at hjertet trækker sig sammen med jævne mellemrum. Når hjertet slapper af, er blodtrykket lavest - denne fase i hjertets cyklus kaldes diastolen, deraf kommer det diastoliske blodtryk. Når hjertet i stedet kontraherer sig er blodtrykket højest - denne fase i hjertets cyklus kaldes systolen, deraf kommer det systoliske blodtryk.

På figur 2 ved punkt c ses det, at systolen starter, når trykket i ventriklerne overstiger trykket i arterierne. AV-klapperne lukkes for at forhindre tilbagestrømning af blodet, der derved vil løbe i den forkerte retning. Når trykket i venstre ventrikkel overstiger trykket i aorta åbnes aortaklappen og blodet vil strømme ud, dette ses ved punkt A. Ved punkt E ses det, at volumenet i ventriklerne falder idet blodet løber fra ventriklen til aorta, og videre ud i det store kredsløb som tidligere nævnt.

På figur 2 ved punkt d ses det, at diastolen starter, når trykket i ventriklerne er lavere end i

This diagram follows left heart and aortic pressures, left ventricular volume, and the ECG through one cardiac cycle. The boxed letters refer to Concept Checks 28-30.



Figur 5.2. Wiggers Diagram [2]

arterierne. Dette medfører, at den ene AV-klap åbnes, som det ses ved punkt b, så hjertet kan fyldes med blod. Under diastolen er aortaklappen lukket. Ved punkt F ses det, at volumenet i ventriklerne stiger, idet blodet løber fra atrierne til ventriklerne.

Systolen starter, når trykket i ventriklerne igen overstiger trykket i arterierne. AV-klapperne lukkes for at forhindre tilbagestrømning af blodet, der derved vil løbe i den forkerte retning. Når trykket i venstre ventrikkel overstiger trykket i aorta åbnes aortaklappen og blodet vil strømme ud.

Blodtrykket måles som det tryk, der er højere end det atmosfæriske tryk. Det angives normalvis i mm Hg

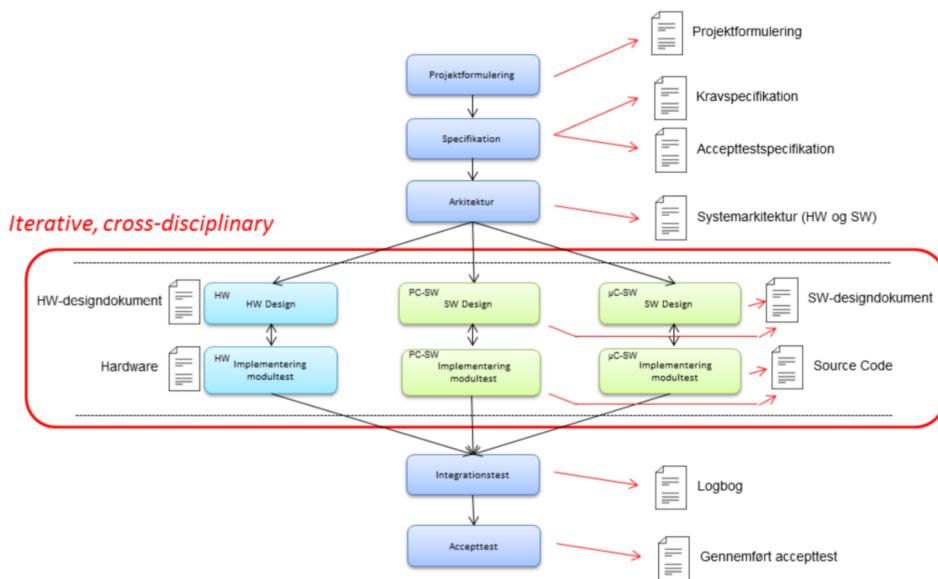
Blodtryk kan måles invasivt eller non-invasivt. Da det er givet i problemformuleringen, at vi skal vise blodtrykket som funktion af tiden, vil dette projekt omhandle et invasivt blodtryksmålesystem.

6 Udviklingsproces

6.1 Metode

I dette semesterprojekt har vi arbejdet ud fra ASE-udviklingsmodellen. Denne model er en kombination af vandfaldsmodellen og v-modellen. V-modellen kommer til udtryk ved specifikationen, hvor vi udformer krav i kravspecifikationen samtidig med, at vi udformer tests i acceptttestspezifikationen. Ved at gøre dette sideløbende sikrer vi, at de krav, der bliver stillet er testbare. Vandfaldsmodellen kommer til udtryk i design og implementeringsfasen. I denne fase forløber projektet i iterationer. Dette gør vi for kontinuerligt at kunne justere og optimere produktet ud fra tidlige iterationer.

Vi har igennem ASE-udviklingsmodellen benyttet os af review på kravspecifikationen, acceptttest, hardware arkitektur og software arkitektur. Reviewet er foregået med de resterende grupper på semesterprojektet. Begge grupper læste og kommenterede på det emne der skulle reviewes. Derefter mødtes grupperne til et møde, hvor eventuelle kommentarer blev forklaret og diskuteret. Grundet at reviewgrupperne på forhånd ved, hvad systemet omhandler, har vi derfor ikke kunne få et optimalt billede af, hvorvidt vores dokumentation er forståelig for en potentiel kunde. Dog har reviewene givet os et billede af, hvordan man kan udforme projektet på flere måder. Vi har fået feedback, der har hjulpet os til at spore os ind på den opsætning vi synes er den rette for vores system.



Figur 6.1. ASE-modellen [3]

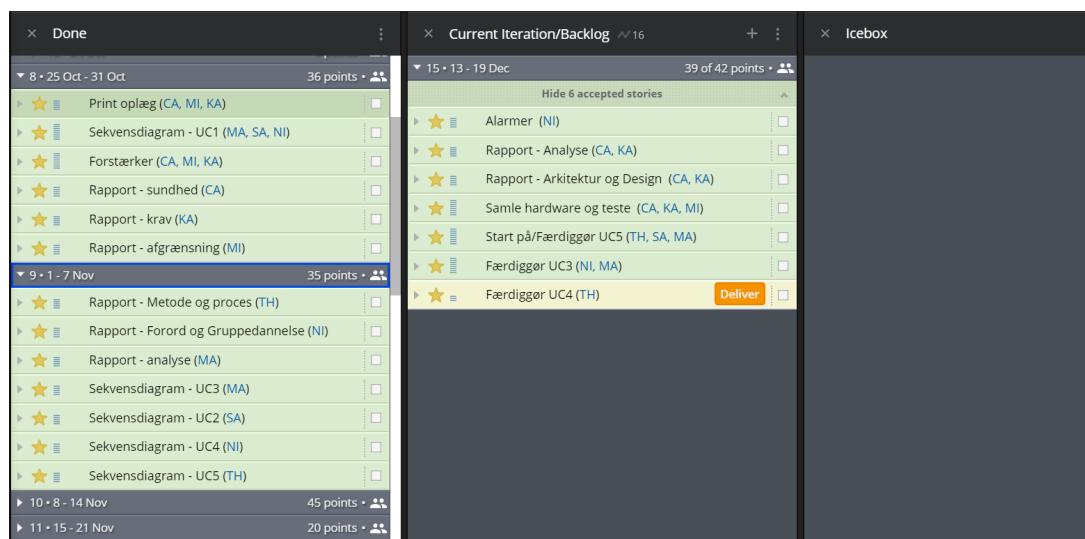
6.1.1 Samarbejdsaftale

Vi har anvendt en samarbejdsaftale, som er udarbejdet i fællesskab i gruppen. Formålet med at udarbejde en samarbejdsaftale er dels at få lavet en forventningsafstemning omkring ambitionsniveauet og arbejdsindsatsen for hvert gruppemedlem for projektet. Hvis de opstillede krav ikke blev opfyldt af et gruppemedlem, ville samarbejdskontrakten have været udfordret, og derfor ville der have været brug for et punkt i samarbejdskontrakten omkring konflikthåndtering. Derfor har vi valgt at formulere en procedure i forhold til hvad der skal ske, hvis der opstår en konflikt. Dog har vi ikke haft nogle konflikter i gruppen, og dermed har vores samarbejdskontrakt ikke været udfordret.

Derudover indeholder samarbejdskontrakten punkter omkring hvornår der afholdes gruppemøde og vejledermøde samt proceduren for afbud til disse. Dette kan ses i samarbejdskontrakten.

6.1.2 Udviklingsforløb

Som arbejdsmetoden i dette projekt har vi anvendt Scrum. Vi har valgt at fokusere på de dele af scrum, der har været relevant i forhold til vores projekt. Vi har valgt at have en product backlog og en sprint backlog. Derudover har vi haft rollerne scrum-master og scrum-team. Vores projektleder Sarah har fungeret som scrum-master. Denne rolle har indeholder opgaverne som ansvarlig for planlægning af sprint, review af sprint, ordstyring af de to ugentlige standup-møder. Hvert sprint har varet en uge. Derfor har vi én gang ugentligt holdt et gruppemøde. Gruppemødet blev sædvanligvis holdt onsdag morgen og til dette møde har vi lavet review af det forrige sprint og planlagt det næste sprint for den kommende uge. Ved review af sprintet har der været fokus på, hvad der er færdiggjort samt scrum-teamets oplevelse af, hvad der er gået godt og hvordan eventuelle problemer er løst. Ved planlægning af sprintet har fokus været, hvad der skal laves og hvad målet for den kommende uge er. Både review af sprintet og planlægning af næste sprint dokumenteres således, at det er tilgængeligt for scrum-teamet. Dokumentationen kan ses i bilag X.

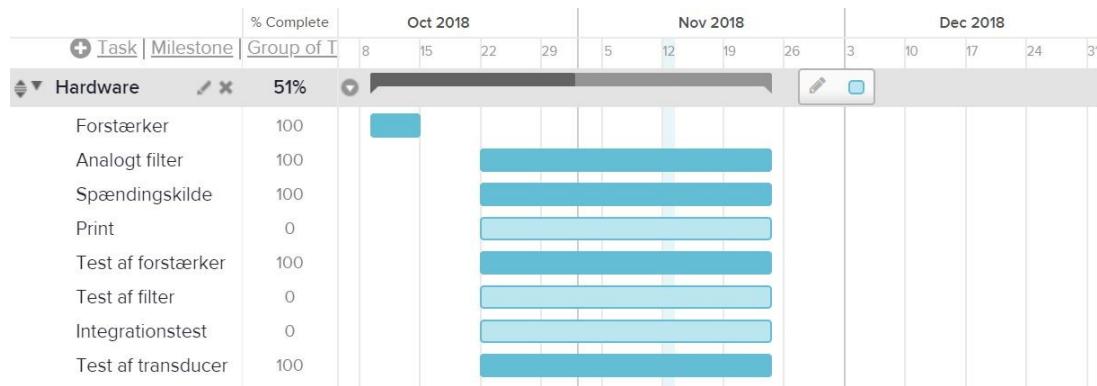


Figur 6.2. Screenshot af Pivotal Tracker

På figur 6.1.2 ses et screenshot af programmet Pivotal Tracker. Pivotal Tracker er et program, hvor et projekt kan inddeltes i mindre opgaver. Disse opgaver placeres i "icebox". Når sprintet planlægges, kigges der på de opgaver der allerede befinner sig i "icebox". Det vurderes, hvilke opgaver der skal startes på og hvorvidt der skal tilføjes flere opgaver til "icebox". Opgaverne

tildeles point alt efter, hvor lang tid det vurderes, at der skal bruges på opgaven. De opgaver der startes placerer sig automatisk i "current backlog". Ved review af sprint kigges der på, hvilke opgaver i "current backlog", der er færdige. Disse opgaver markeres herefter færdige, og flyttes dermed til "done".

Udover gruppemøder har vi to gange om ugen haft standup-møder. Formålet med standup-møder har været at få en status på, hvor langt teamet har været med de forskellige opgaver samt, om der har været problemer, og hvordan disse har kunne løses. Derudover har standup-møderne været med til at give Sarah, der har været projektleder og scrum master, et billede af om tidsplanen skulle opdateres. Til at konfigurere tidsplanen har vi brugt Team Gantt. Et eksempel på dette ses nedenfor.



Figur 6.3. Screenshot af TeamGantt

Sarah har stået for at opdatere tidsplanen, og har dermed kunne forholde sig til, hvornår teamet skulle bruge flere timer i projektet, så opgaverne blev fuldført. Udover opdatering af tidsplan og tovholder på sprint review samt sprint planlægning har Sarahs opgaver været mødeindkaldelse til gruppemøder samt referent til disse. I forhold til lederrollen har gruppen ikke givet Sarah nogle beføjelser. Sarahs opgaver som leder har primært været at være tovholder på projektet.

6.1.3 Arbejdsfordeling

Gruppen tog en beslutning om at fordele projektarbejdet i to teams; et softwareteam og et hardwareteam, da projektet er så stort at det ikke er muligt at være inde over det hele. Softwareteamet er bestående af Mathias, Nicolai og Thea, mens Hardwareteamet er bestående af Caroline, Kajene og Mikkel. Sarahs rolle er flyver, som skiftevis er med softwareteamet og hardwareteamet, alt afhængigt af hvor der er brug for ekstra hjælp. Vi fandt dog hurtigt ud af, at Software-gruppen havde mest brug for hjælp og derfor har Sarah været en del af denne siden uge 40.

Denne opdeling har fungeret udmærket, da den har givet mulighed for at specialisere sig, og komme dybere ned i stoffet. Dog har det også givet anledning til frustration i forhold til ikke at være helt inde over hvad det modsatte team arbejder med. De ugentlige standup-møder har bidraget til at begge teams får en forståelse for hvad der bliver arbejdet med i både hardware og software.

Projektarbejdet er også blevet uddelegeret, så hvert gruppemedlem har fået primære og sekundære områder, arbejdsfordelingen i detaljer ses i et skema i bilaget "arbejdsfordeling".

6.2 Programmer

Til projektet har vi brugt programmerne:

- LaTex
- Google Drive
- GitHub
- Visual Studio
- Multisim
- Ultiboard
- Waveforms
- Pivotal Tracker
- TeamGantt
- MatLab
- MathCad Prime 4.0
- EuroCircuits

Til hardware-delen har vi brugt MathCad Prime 4.0 til analyse-delen. I dette program har vi foretaget alle udregninger. Vi har brugt programmet Waveforms til realiseringen af vores byggede kredsløb - både på fumlebræt og på printpladen. Multisim og Ultiboard har vi brugt i forbindelse med vores design af vores printplade. Før printpladen blev sendt endeligt til print blev den testet gennem EuroCircuits, som også er leverandør af printpladen.

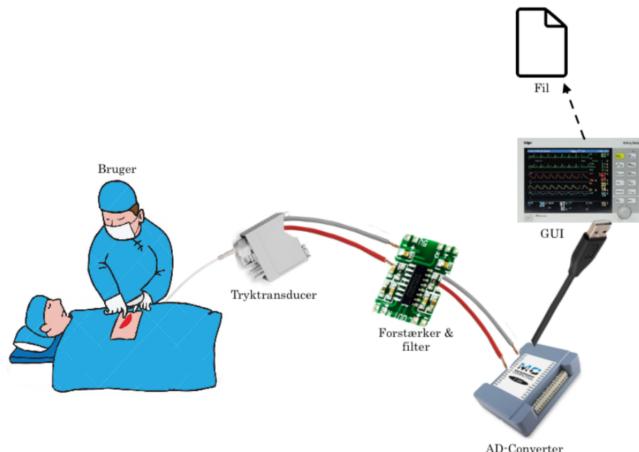
Vores software er skrevet i Microsoft Visual Studio i sproget C#. Det var et krav fra projektets vejledere at skrive koden i dette sprog. Vi har valgt at skrive koden til vores software i Microsoft Visual Studio da det er et program, vi har fået undervisning i på 1. og 2. semester. Som tilføjelse til dette har vi valgt at implementere GitHub. Dette har muliggjort, at software gruppen har kunne arbejde på samme projekt på hver deres computer. Dette har optimeret software gruppens arbejde, så projektet var samlet fra start. Til software-delen har vi brugt MatLab til at lave alarmer.

Vi har også brugt GitHub i forbindelse med programmet LaTex. Dette har vi brugt fordi, at alle på den måde har kunnet skrive deres afsnit af rapporten og derefter synkronisere vha. GitHub, således at de resterende medlemmer i gruppen har alle afsnit. Før vi valgte at kaste os ud i at lære at bruge LaTex brugte vi Google Docs under Google Drive til at skrive vores dokumenter. Her havde vi alle mulighed for at skrive, se og rette alle dokumenter og dette har fungeret godt som en start til projektet. På Google Drive har vi endvidere haft en samlet mappe til alle relevante dokumenter i forbindelse med projektet.

Til selve styringen af projektet i forbindelse med scrum har vi brugt programmerne Pivotal Tracker og TeamGantt. Disse to programmer er beskrevet lidt nærmere i forrige afsnit [6.1.2](#) på [side 13](#).

7 Systembeskrivelse

7.1 Systemoversigt



Figur 7.1. Systemoversigt

7.2 Systembeskrivelse

På figur 7.1 ses systemoversigten for vores blodtrykmålesystem, der har til formål at kunne måle og vise en patients blodtryk og puls på en brugergrænseflade.

Systemet består af:

- Tryktransducer
- Forstærker
- Subtraktor
- Anti-aliaseringsfilter
- AD-Converter (NI DAQ 6009 USB)
- Computer
- Bluetooth højtalere
- Skærm
- Brugergrænseflade

Brugergrænsefladen er bygget op som et Graphical User Interface (GUI) hvorpå det målte blodtryk visualiseres kontinuert i form af en graf. Det systoliske-, diastoliske-, median blodtryk og puls vises i form af tal. Brugergrænsefladen består af knapper, der har forskellige funktioner. For nærmere beskrivelse af brugergrænsefladens funktioner henvises der til dokumentet kravsspecifikation.

7.2.1 Alarmer

Systemet skal i henhold til standard 60601-1-8 vedrørende alarmering kunne alarmere ved eventuelle fejl eller ændringer. I forhold til brugssituationen af blodtrykmålesystemet vurderes det at systemet skal kunne alarmere, når der er fald eller stigning i blodtryk eller puls uden for de justerbare grænseværdier. Alarmen er en high priority, som alarmerer visuelt og auditivt. For krav til alarmen henvises til dokumentet kravspecifikation afsnit 1.12 og dokumentet for alarmspecifikationerne, som er udarbejdet ud fra standard ISO 60601-1-8: 2017.

8 Krav

8.1 Krav

For ethvert system er der opstillet en række funktionelle og ikke-funktionelle krav for at sikre at kunden og udvikleren er enige om, hvilke funktionaliteter systemet skal have samt systemets kapacitet og betingelser.

For dette system er der tilknyttet fem aktører; to primære og tre sekundære. Den ene primære aktør er en bruger, som betjener blodtryksmåleren. Den anden primære aktør er det tekniske personale, som skal foretage kalibreringen af systemet. De sekundære aktører er blodtrykssensoren, som er et interface for patienten som er koblet til systemet, bluetooth højtaleren, der bruges til at afspille alarmlydene, samt storskærmen, der skal bruges til at vise brugergrænsefladen meget større, så den er let at se for alle på operationsstuen.

Aktørerne interagerer med systemet, hvilket opstiller 5 use cases (funktionelle krav). UC1 er hovedscenariet idet, der måles og vises blodtryk og puls. UC2 omhandler om at justér grænseværdier for puls samt systolisk- og diastolisk blodtryk. UC3 omhandler alarmeringen ift. ændringer i blodtryk samt puls eller ved fejl. UC4 omhandler funktionen at gemme patientens CPR, de opsamlede blodtryksdata, digitalt filter status, kalibreringsdato samt dens værdi og alarmer iløbet af operationsstuen samt alarmbetingelserne iflg. uddraget af standard 60601-1-8 i en fil. UC5 omhandler kalibreringen af systemet. For yderligere detaljer se kravspecifikationen.

Når der opstilles en række use cases, opstilles der samtidig en række ikke-funktionelle krav, der skal overholdes for at use casene kan udføres optimalt. Den samlede oversigt over de ikke-funktionelle krav ses i kravspecifikationen. Her er der valgt at beskrive fire funktionelle krav der har fået prioriteringen must have ift. MoSCOW-modellen.

- Systemet skal have en brugergrænseflade
- Systemets GUI skal alarmere i henhold til punkt 1.12.4, hvis pulsen falder til under 60 slag i minuttet og overstiger 120 slag i minuttet.
- Systemet skal alarmere i henhold til punkt 12.1.4, hvis blodtrykket falder til under 60 diastolisk eller stiger til over 160 diastolisk.
- Diagrammet samt de viste værdier for puls skal kunne læses på op til 1,5 meters afstand af person uden synshandicap, eller som benytter korrekt korrigende midler, og må ikke indeholde farver som påvirker folk med farveblindhed.

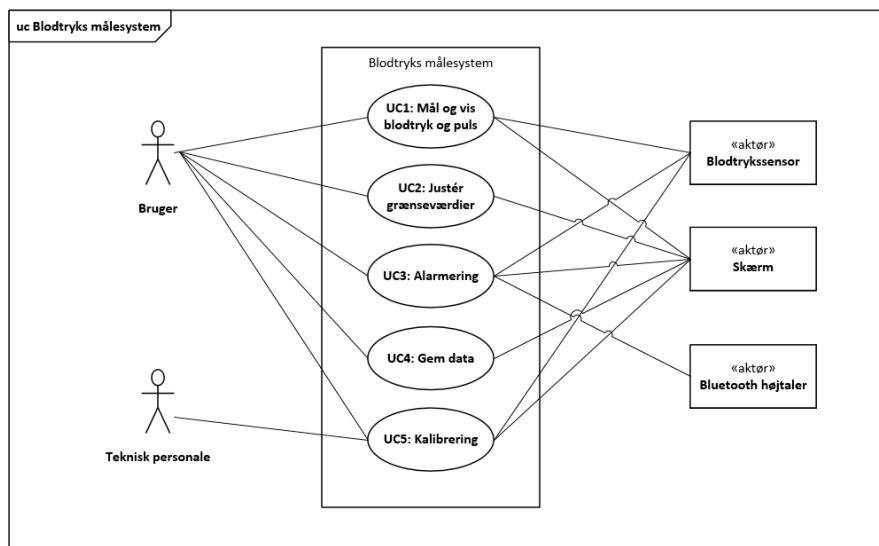
Uden det ikke-funktionelle krav vedrørende brugergrænsefladen ville udviklerne kunne vise de opsamlede data som de ville, og ikke som kunden ønsker det. I dokumentet kravspecifikation i bilaget ses flere ikke-funktionelle krav opstillet med fokus på brugergrænsefladen, så den er

beskrevet så specifikt som muligt.

De ikke-funktionelle krav, der henvender sig til alarmeringen er meget vigtig, da brugeren skal reagere hurtigt hvis patientens blodtryk eller puls falder eller stiger, da det kan være livstruende.

I henhold til brugssituationen i en operationsstue kan det ske, at brugeren måler patientens blodtryk og samtidig skal ordne noget ved underekstremiteterne, og dermed skal kunne læse værdien for puls og blodtryk på 1,5 meters afstand. Ligeledes skal der tages højde for at brugeren kan være farveblind, og dermed skal der benyttes andre farver end rød og grøn.

8.2 Use Case diagram



Figur 8.1. Use Case Diagram

8.3 Aktørbeskrivelser

Aktør	Type	Beskrivelse	Samtidige forekomster
Bruger	Primær	<p>Brugeren betjener blodtryks målesystemet.</p> <p>Brugeren nulpunktsjusterer, igangsætter og stopper måling. Brugeren er en sundhedsfaglig person, der arbejder på operationsstuen.</p>	<p>1. Da der kun er én blodtryksmåler i systemet, kan der kun være en bruger.</p>
Teknisk personale	Primær	<p>Teknisk personale er teknisk uddannet og skal foretage kalibreringen af systemet.</p>	<p>1. Da der kun er én blodtryksmåler i systemet, kan der kun være et teknisk personale</p>

Blodtryksensor	Sekundær	Blodtryksensoren er et interface for patienten som er koblet til systemet	1. Da der kun er én blodtryksmåler i systemet, kan der kun være en blodtryksensor
Bluetooth højtalер	Sekundær	Bluetooth højtaleren afspiller lyden for de alarmer der står i systembeskrivelsen.	1. Da der kun er én blodtryksmåler i systemet, kan der kun være en bluetooth højtalér
Skærm	Sekundær	Skærmens bruges til at vise brugergrænsefladen meget større end på computeren. Skærmens gör det nemt og overskueligt for hele personalet at følge med i blodtrykmålingen og de målte værdier af systolisk-, diastolisk-, median blodtryk og puls.	Flere. Da skærmens er uafhængig af antal blodtryksmåler i systemet, vil der kunne være flere skærme

Tabel 8.1. Aktørbeskrivelser

8.4 Use Case beskrivelser

Dette afsnit giver en kort beskrivelse af systemets Use Cases. For en yderligere og mere detaljeret beskrivelse af Use Cases henvises til dokumentet kravspecifikation afsnit xx.

8.4.1 Use Case 1: Mål og vis blodtryk og puls

Før en måling kan igangsættes skal systemet nulpunktjusteres. Patientens målte blodtryk visualiseres kontinuert i form af en graf på brugergrænsefladen. Systolisk-, diastolisk-, median blodtryk og puls vises i form af tal ligeledes på brugergrænsefladen.

8.4.2 Use Case 2: Justér grænseværdier

Før og under en måling er det muligt at justere grænseværdier for blodtryk og puls. Default værdier for grænseværdierne findes i systembeskrivelsen i bilaget kravspecifikation afsnit xx.

8.4.3 Use Case 3: Alarmering

Før og under en måling skal systemet kunne alarmere ved eventuelle fejl eller ændringer. Systemet alarmerer i henhold til standarden og prioriteres efter *high priority*, *medium priority* eller *low priority*.

8.4.4 Use Case 4: Gem data

Efter endt måling er det muligt at gemme patients blodtryksmåling, digitalt filter status og alarmer i en fil. Personale ID, patientens CPR-nummer samt dato og tid indtastes. Kendes patientens CPR-nummer ikke indtastes "000000-0000".

8.4.5 Use Case 5: Kalibrering

Kalibrering af systemet skal foretages af teknisk personale. Kalibreringen foregår én gang årligt.

8.5 Afgrænsning

I kravspecifikationen er der beskrevet de ikke-funktionelle krav vha. MoSCoW- modellen.

INDSÆT BILLEDE HER

I ovenstående tabel ses vores de MoSCoW krav, som er implementeret på vores system.

Vores fokus har været invasiv blodtryksmålingen, og derfor har vi valgt ikke at prioritere at implementere fx EKG-måling. Vores won't have krav lyder derfor således, "Systemet kan ikke vise andet end blodtryksmålingen samt angive puls, diastolisk/systolisk blodtryk (F)".

De softwaremæssige afgrænsninger i systemet er blandt andet, at der ikke kan hentes, samt åbnes en tidligere måling fra en fil. Herudover har vi valgt at gemme målingens værdier i en fil, frem for i en database. Ift. brugssituationen, ville det give god mening at benytte både en fil og en database, da data kan tilgås og ses fra flere enheder. Eftersom det at gemme i en fil er et must have krav, er det blevet prioriteret højere at opfylde dette end could have kravet om at gemme i en database. Yderligere er et could have krav, at systemet kunne have et lydsignal ved afsluttet blodtryksmåling, dette blev dog ikke prioriteret, da det kun var et could have krav. Derimod er det prioriteret, at systemet skal kunne beregne og vise middelblodtrykket på GUI'en. Dette er et must have krav, da man gennem en operation skal have en indikation på, hvordan blodtrykket for patienten ligger. Systemet er en prototype, hvor softwaren kræver en PC som kan køre Visual Studio, hvor det til fremtidigt arbejde ville være oplagt at lægge softwaren på en mikroprocessor.

Det er nødvendigt at man manuelt skal foretage en nulpunktsjustering inden hver måling igangsættes, for at sikre sig at man ikke mäter det atmosfæriske tryk med i beregningerne. Sker dette ville vores system ikke kunne registrere værdierne, da vi har bygget systemet efter at kunne håndtere et max tryk på 250 mmHg. Derfor er dette et must krav. Endnu et must krav er, at der årligt skal laves en kalibrering af systemet, for at sikre en vedvarende præcision af systemet. Dette foretages af en teknisk person, som sørger for at et kendt tryk svarer til en kendt spænding, således systemet mäter korrekt. Hardwaren er følsom overfor vand og støv, da der ikke er prioriteret at bygge en boks til at skjule printpladen. Derudover kræver hardwaren at Analog Discovery bruges som power supply.

Der er valgt en række værdier ifm. udviklingen af hardwaren, hvilket gør at den er begrænset i forhold til hvilke størrelser af værdier den kan håndtere. Forstærkeren kræver specifikke modstande, for at operationsforstærkeren skal forstærke op til de 4 V, som vi har valgt vores system skal kunne. Derudover skal systemet benytte en tryktransducer med en sensitivitet på $5,0 \mu\text{V}/(\text{V} \cdot \text{mmHg})$, da forstærkningsfaktoren er beregnet ud fra denne, og dermed er modstandsvalget afhængig af en tryktransducer med en sensitivitet på $5,0 \mu\text{V}/(\text{V} \cdot \text{mmHg})$.

9 Arkitektur

9.1 Hardwarearkitektur

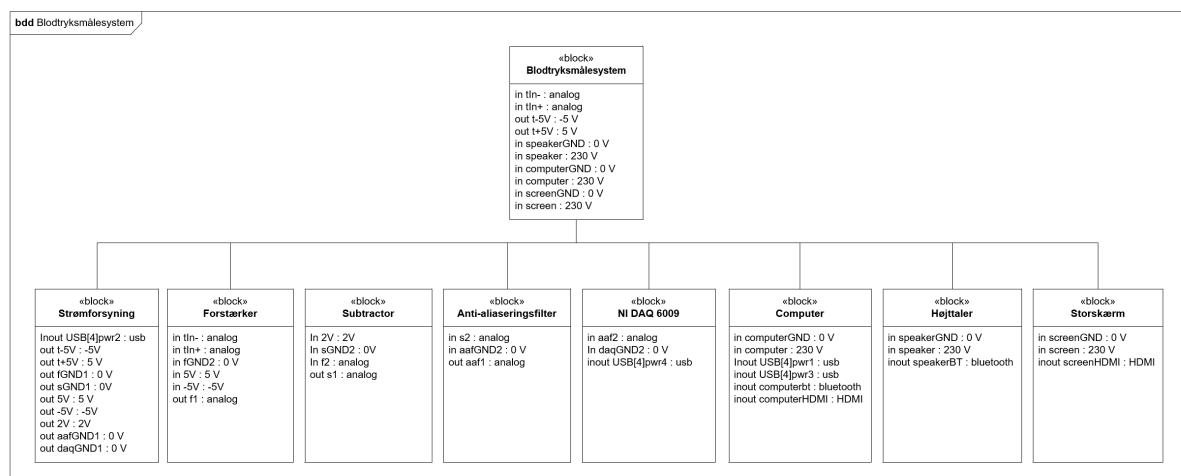
I den følgende sektion om hardwarearkitektur beskrives opbygningen af systemet ved brug af sysML. Hardwarearkitekturen indholder diagrammerne BDD og IBD samt en kort forklaring af disse. For ydeligere information om interaktionen i systemet henvises til dokumentet arkitektur og design for flere detaljer. Dette dokument indeholder bloktabel og signaltabel, der fortæller om alle blokke, porte og signaler, der ses i diagrammerne.

9.1.1 BDD

Et BDD viser opbygningen af et system med brug af blokke. Nedenstående diagram på figur 9.1 viser opbygningen af vores blodtryksmålesystem, der består af en række forskellige elementer.

Dette diagram er det endelige diagram for vores system. Første version af vores BDD indeholdte ikke en subtractor. Dette element er tilføjet i form af en blok mere på BDD'et. Derudover har vi ændret i vores valg af strømforsyning som nu udgøres af Analog Discovery. Dette har vi valgt fordi subtractoren bruger en 2V forsyning til at trække signalet ned med tilsvarende. Før dette valg blev truffet blev strømforsyningen udgjort af computeren, der sendte en 0-5V forsyning gennem systemet via en USB-forbindelse.

Portnavnene på diagrammet er endvidere ændret i forbindelse med designet af printpladen. Portnavnene er blevet forkortet og forenklet så det var mere overskueligt og så disse navne kunne stå tilsvarende på det endelige design af printpladen.



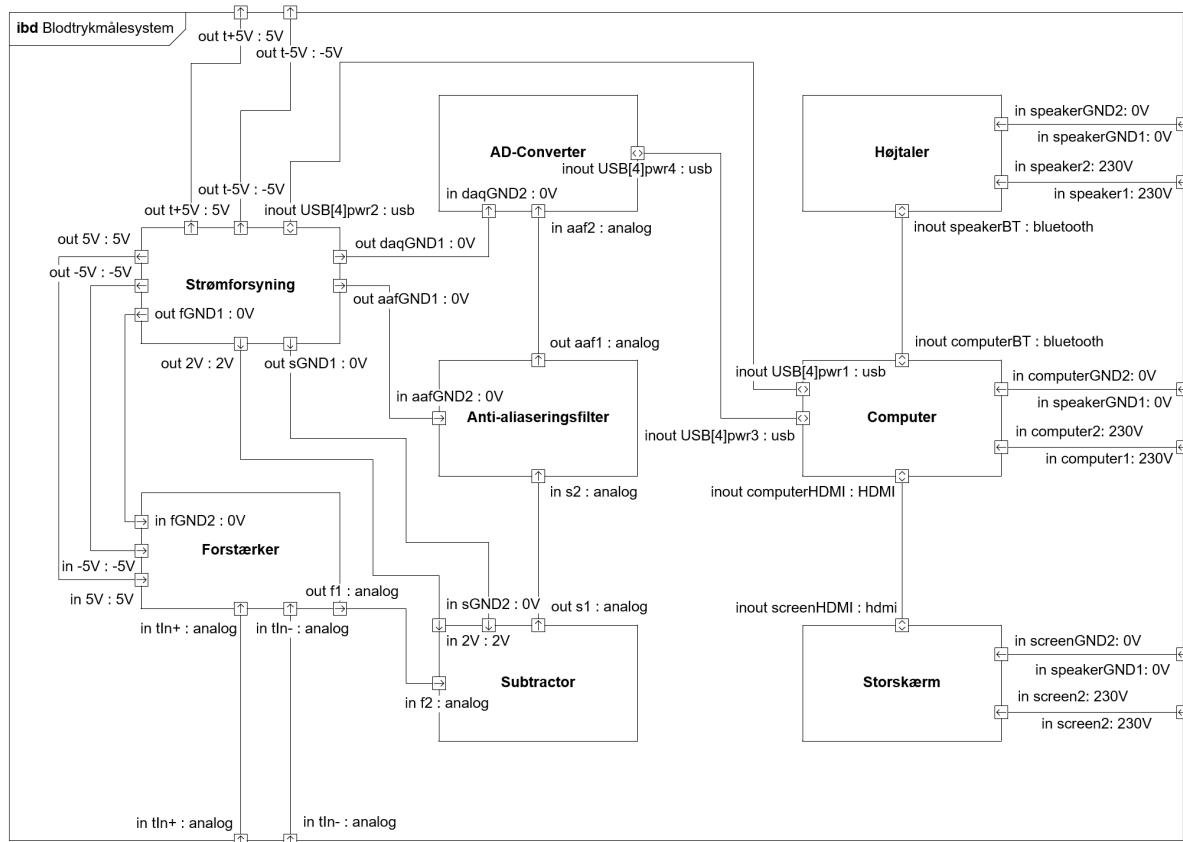
Figur 9.1. BDD Blodtrykmålesystem

På 9.1 på foregående side ses de forskellige elementer, der er en del af blodtrykmålesystemet. Den øverste blok ”Blodtryksmålesystem” viser systemets interaktioner med udefrakommende objekter. I denne blok findes tilslutningen til tryktransduceren, som ikke er en del af selve systemet. I blokken findes også tilslutningen af strøm til henholdsvis computer, højtalere og storskærm.

På diagrammet ses også otte blokke, der repræsenterer de forskellige dele af systemet. Systemet består altså af en strømforsyning, en forstærker, en subtractor, et filter, en AD-Converter, en computer, en højtalere og en storskærm. Strømforsyningen giver strøm til tryktransducer, forstærker og subtractor og udgøres af Analog Discovery. Forstærker, subtractor og filter er designet til systemet i forbindelse med projektet. Mere info om disse elementer i afsnit om design af hardware.

9.1.2 IBD

Et IBD viser en mere detaljeret opbygning af et system fordi alle forbindelser mellem blokkene, der findes i BDD’et er tegnet. På nedenstående figur 9.2 ses derved opbygningen af blodtryksmålesystemet inklusiv alle forbindelser mellem blokkene.



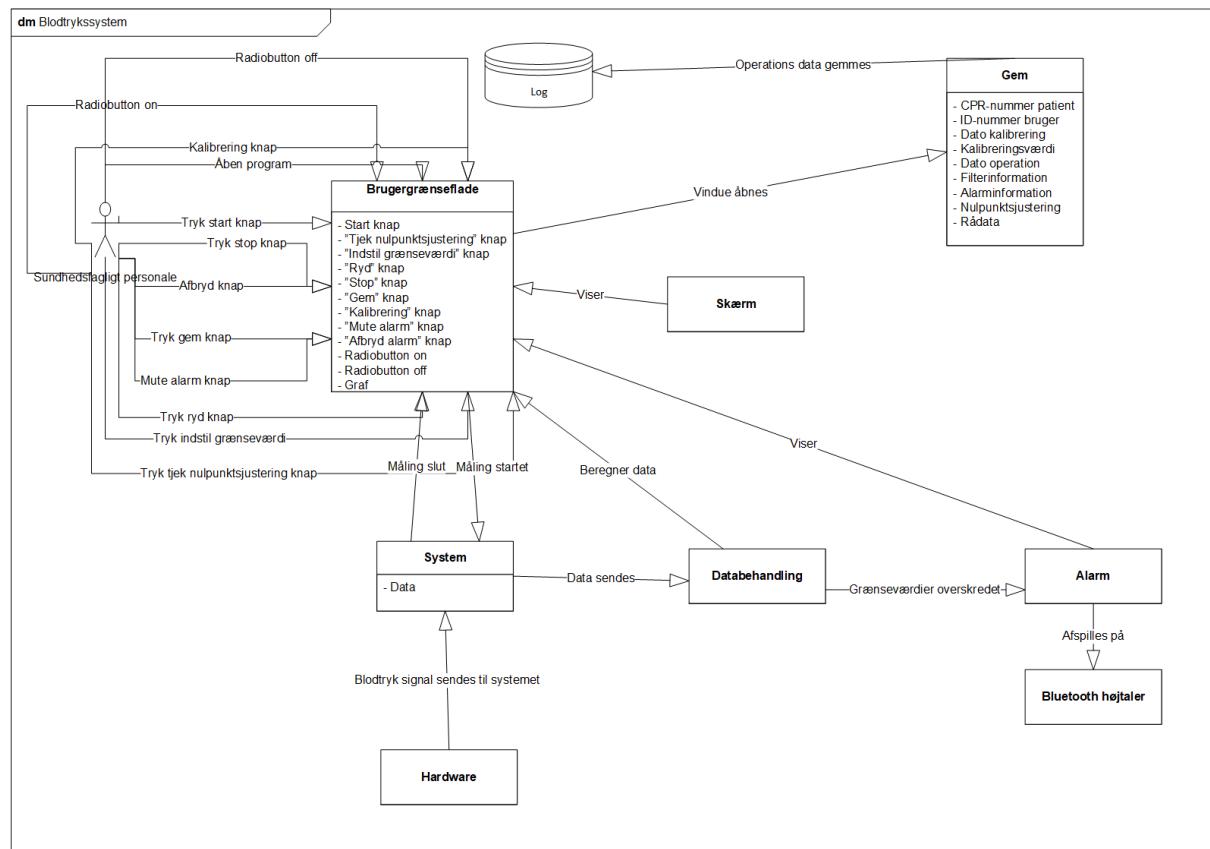
Figur 9.2. IBD Blodtrykmålesystem

Den yderste ramme omkring diagrammet illustrerer den øverste blok ”Blodtryksmålesystem” på figur 9.1 på foregående side. Her ses det tydeligt hvordan nogle af blokkene interagerer med objekter uden for selve systemet fordi forbindelserne er tegnet ud til rammen på diagrammet. Hver port har et unikt portnavn. IBD’et er med til at forbinde portnavnene fra BDD’et, så det giver et overblik over systemet som en helhed.

9.2 Softwarearkitektur

Dette afsnit beskriver softwarearkitekturen. For at give det fornødne overblik over softwarearkitekturen i vores system, har vi valgt at inkludere en overordnet domænemodel, en domænemodel for software samt et tomt klassediagram uden attributter og metoder. I afsnittet design findes sekvensdiagram og klassediagram for UC1. Diagrammer er med til at danne grundlaget for software designprocessen. Ved at man bryder systemet ned i mindre dele, giver det tilmed også et overblik over hvad der skal foregå i systemet, og dette gør det nemmere at finde frem til designløsninger. Nedenstående figur den overordnede domænemodel, som giver et overblik og hele systemet.

9.2.1 Overordnet domænemodel

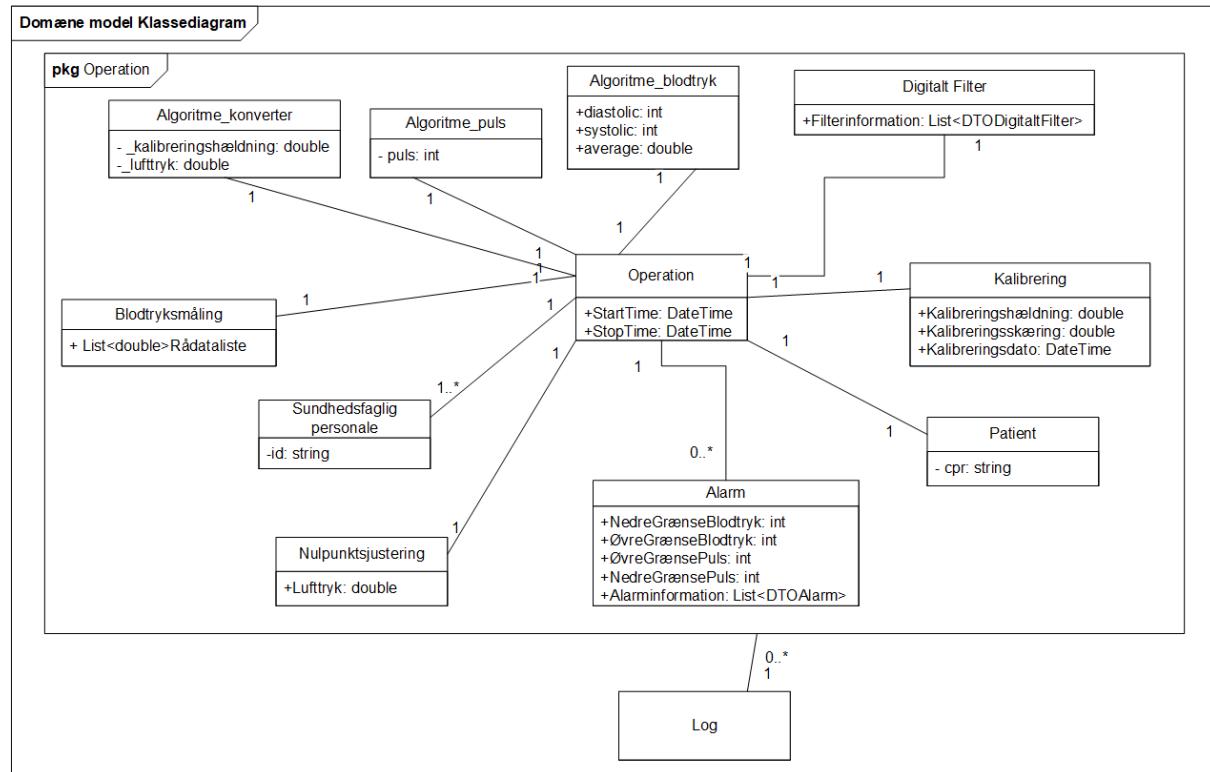


Figur 9.3. Overordnet domænemodel

Ovenstående figur 9.2.1 illustrerer den overordnede domænemodel, som viser både hardware og software. Domæneklasserne fandt vi efter en navneords analyse af vores fully dressed use cases, hvor vi fandt frem til brugergrænseflade, sundhedsfagligt personale, log, skærm, gem, system, databehandling, alarm og bluetooth højtalér. Log domænet er en tekstfil, hvor oplysninger fra operationen gemmes i.

9.2.2 Domænemodel for software

Ud fra vores kendskab til domænet, har vi lavet en software-domænemodel, hvor vi får et overblik over, hvilke domæneklasser vores software skal indeholde.

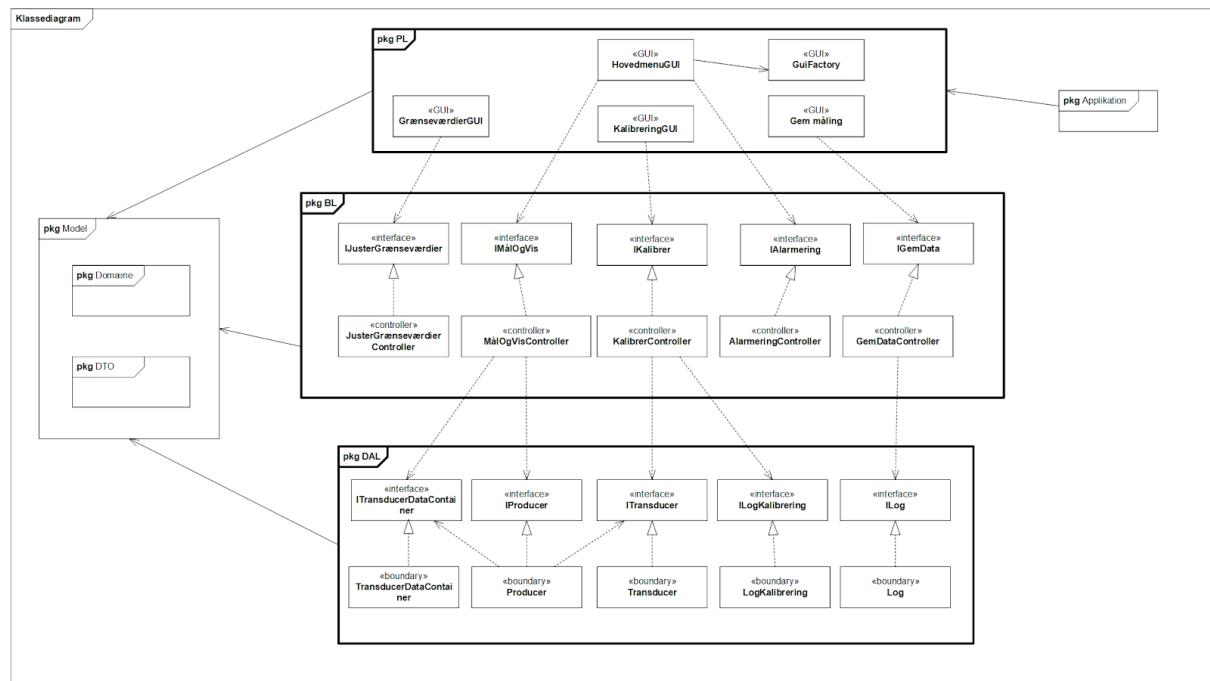


Figur 9.4. Domænenmodel for software

Ovenstående figur 9.2.2 viser systemets domæne for softwaren og hvilke klasser dette er bestående af. Herudover viser den relationen mellem de forskellige domæneklasser og operationen. For eksempel kan der ved 1 operation være 0 til mange alarmer, som er illustreret ved "0..*". Modellen viser også en oversigt over klassernes tilhørende attributter og typerne af disse. Typerne har vi rationaliseret os frem til ved viden om at vores målinger vil være decimaltal (doubles), vores beregner vil vi gerne have vist i heltal (int), id og cpr er hensigtmæssigt at gemme som en string, og så vi nogle start- og stoptidspunkter som er oplagte at lave som DateTime.

Ud fra vores use cases er vi kommet frem til, at ovenstående domæneklasser kan beskrive vores domæne tilstrækkeligt. Det gjorde vi, som tidligere nævnt, ved at lave en navnordsanalyse af vores fullydressed use cases. Ud fra denne fandt vi domæneklasserne: sundhedsfagligt personale, nulpunktsjustering, patient, alarm, kalibrering, digitalt filter, blodtryksmåling samt log. Ud fra disse gav det mening at samle det hele under klassen operation, som er vores brugssituations. Vi indså, at vi manglede klasser for vores databehandling. Dette delte vi op i tre klasser. En til pulsalgoritme, en til blodstryksalgoritme og en klasse til at konvertere fra volt til mmHg.

Dernæst lavede vi et overordnet tomt klassediagram, som er udformet på baggrund af vores use cases. Klasserne fra software domænenmodellen er en del af vores Model package, som indeholder domæneklasser og dtoklasser.



Figur 9.5. Klassediagram uden attributter og metoder

Klassediagrammet på figur 9.2.2 viser opbygningen af softwaren. Vi har valgt at bygge vores system op efter trelagsmodellen, med præsentationslag (PL), businesslag (BL) og datalag (DAL). Dette er en god opbygning af et større system, som skal indhente, behandle og vise data.

Klassediagrammet viser opbygningen af softwaren. Vi har valgt at bygge vores system op efter trelagsmodellen, med præsentationslag (PL), businesslag (BL) og datalag (DAL). Dette er en god opbygning af et større system, som skal indhente, behandle og vise data.

Klasserne fra software domænemodellen er en del af vores Domæne package, som ligger i vores Model package, der også indeholder en DTO package, som består af fem dtoklasser. En for alarm der skal bruges til at gemme alarmtype og tidspunkt for en alarm, en for blodtryksberegninger som bruges til at sende beregninger fra BL til PL, en for digitalt filter der skal bruges i forbindelse med logging af filterstatus i løbet af en måling, en DTO med alle properties der skal gemme i loggen og en for kalibrering. Sidst men ikke mindst er der en package kaldet Applikation, som står for at starte hele applikationen op.

Softwareen er designet ud fra paradigmet om lav kobling. Dette er først og fremmest gjort ved, at dele de forskellige use cases op i egne klasser med interfaces imellem for at opfylde open-closed principippet. Derudover er der brugt dependency injection, som ligeledes fremmer en lav kobling. Denne opbygning gør det også muligt at benytte scrum da hver use case har sin egen klasse i softwaren, og man dermed kan arbejde individuelt på hver sin use case. Desuden bliver det også lettere at teste og lokalisere eventuelle fejl. Derudover er det i fremtiden, lettere at genbruge softwaren i nye applikationer, samt at udvide uden at skulle lave alt for mange ændringer i koden.

Et knudepunkt i softwaren er klassen `HovedmenuGUI`, som har mange ansvar. Dette ansvar er forsøgt mindsret ved at lave en `Guifactory` klasse, som sørger for at oprette instanser af de tre andre GUI'er. Dette sikrer også, at det ville være nemt at tilføje en anden GUI, hvis softwaren skal udvides med en ny funktionalitet.

Kommunikationen mellem datalaget og logiklaget er designet ud fra producer-consumer principippet. Dette var en oplagt løsning, da vores måleenhed hele tiden producerer data, som skal bruges og behandles af controllerklasserne i logiklaget. En anden vigtig feature ved producer-consumer er, at en datakø er indbygget, som sørger for, at der ikke går måledata tabt mellem datalaget og logiklaget, som kører i hver sin tråd. Køen har vi realiseret ved at bruge AutoResetEvents, som forhindrer, at producertråden overskriver måledata, inden de er blevet consumed i logiklaget. På den måde undgås fejl i kommunikationen mellem datalaget og logiklaget.

Mellem logiklaget og præsentationslaget var det i første omgang planen at bruge observer pattern. Det viste sig dog, at dette ikke var særlig hensigtsmæssigt, da der er forskel på, hvor tit blodtryksgrafen og beregningerne på hovedmenuen skal opdateres. Dette gjorde det vanskeligt at bruge observer standardmønstret. Der var behov for en mere generisk løsning. Kommunikationen mellem logiklaget og præsentationslaget blev i stedet løst ved at bruge events, som på mange måder minder om observer, men er en mere fleksibel metode, der er indbygget i .NET frameworkt. Dette viste sig også at være en god løsning på kommunikation mellem alarmeringscontrolleren og hovedmenuen. Her blev der altså afgivet fra ASE-modellen, da der blev arbejdet iterativt med software arkitekturen. Det fleste ændringer har dog været mindre ændringer som fx tilføjelse af flere klasser i datalaget. Se dokumentet softwarearkitektur og design hvor den første version af klassediagrammet kan findes.

En anden udfordring var en samplefrekvens på 1000 Hz og en computerskærm, der opdaterer med 60 Hz. For at undgå en nedampling af signalet i softwaren, blev det besluttet, at grafen opdaterer med 100 punkter af gangen 10 gange i sekundet. Opdateringsfrekvensen på skærmene bliver således kun 10 Hz. Blodtryksgrafen opdaterer altså ikke helt så smooth, men dette blev besluttet som et fint kompromis, da der ikke mistes information i signalet på grund nedampling.

10 Design

10.1 Hardwaredesign

I det følgende afsnit vil de vigtigste overvejelser og udregninger for designet af systemets hardware fremtræde. En mere detaljeret begrundelse for valg findes i dokumentet analyse i bilaget. Dokumentet analyse indeholder også alle de beregninger der er foretaget i forbindelse med valg af forstærkning, filtertype og subtractor.

Vi har desuden valgt at stabilisere vores hardware med brug af afkoblingskondensatorer. Vi har små afkoblingskondensatorer på hver del, herunder forstærker, subtractor og filter. Vi har to store afkoblingskondensatorer på fores forsyningsspænding på henholdsvis +5V. Vi har valgt at bruge disse for at sikre vores hardwares overfølsomhed så meget som muligt mod støj.

10.1.1 Forstærker

Grundet at det signal, der udsendes gennem tryktransduceren er for småt til, at det kan analyseres og anvendes i vores system, skal signalet derfor forstærkes. Til projektet blev det givet, at signalet skulle forstærkes med en operationsforstærker af typen INA114. Denne type operationsforstærker har et variabelt gain, der afhænger af hvilken modstand, der forbindes til operationsforstærkeren.

For at vide hvor meget vi ønsker at forstærke signalet fra tryktransduceren er det vigtigt at kende til størrelsen af signalet. Mængden af forstærkning bestemmes derfor ud fra tryktransducerens følsomhed. Denne følsomhed findes i databladet for tryktransduceren. Vi udregner signalets størrelse fra transduceren i forhold til det maksimale blodtryk vi har valgt, at systemet skal kunne måle op til.

Tryktransducerens følsomhed:

$$V_{out} = 5 \cdot \frac{\mu V}{V \cdot mmHg}$$

$$V_{out} = 5 \cdot \frac{\mu V}{V \cdot mmHg} \cdot 10V \cdot 250mmHg = 12500\mu V = 12,5mV$$

Fordi vi kender følsomheden på tryktransduceren og ved, at vi ønsker at kunne forstærke blodtrykssignaler op til 250 mmHg og har en forsyningsspænding på -5 til 5V via Analog Discovery får vi ovenstående resultat.

Forstærkerens gain kan udregnes med følgende formel:

$$G = 1 + \frac{50k\Omega}{R_G}$$

Vi ønsker at forstærke signalet fra 12,5 mV op til 4V. Denne forstærkning er valgt for at kunne udnytte alle 14-bit på AD-Converteren.

Komponenterne har vi udregnet på følgende måde:

$$gain = \frac{4V}{V_{out}} = 320$$

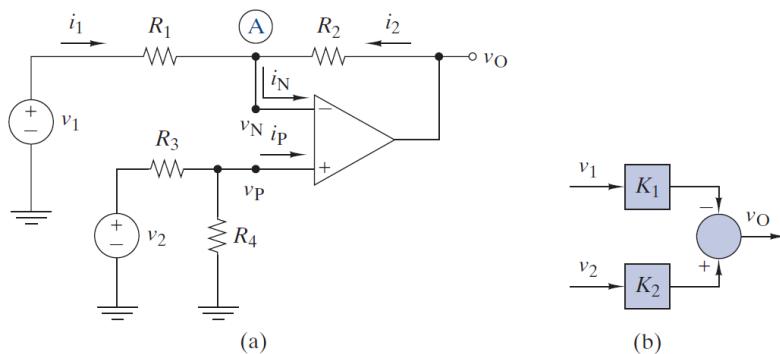
$$320 = 1 + \frac{50k\Omega}{R_G} \rightarrow R_G = \frac{5000}{319}$$

$$R_G = 156,74\Omega$$

Denne beregnede modstand sikrer at vi får en forstærkning på 320 gange.

10.1.2 Subtractor

For at benytte alle 14-bit på AD-Converten, NI-DAQ 6009, har vi i forbindelse med designet af forstærkeren besluttet at forstærke fra 0-4V. Alle tal vi får ind fra tryktransduceren vil derfor ligge i dette område. For at få vores indgangssignal til at passe med indgangene på AD-Converten og derved udnytte alle bit er det nødvendigt at nedjustere signalet. Vi bruger derfor en subtractor af typen OP27G, som er en standard operationsforstærker, der findes i laboratoriet. Subtractoren justerer vores signal så det går fra -2V til 2V i stedet for 0-4V.



Figur 10.1. Kredsløb for subtractor/Differential Amplifier

Til subtractoren bruger vi fire ens modstande. Dette gør, at de fire modstande divideret med hinanden giver 1.

$$\frac{R3}{R1} = \frac{R4}{R2} = 1$$

Ved brug af superposition har vi fået et udtryk for V_O .

$$V_O = V_2 - V_1$$

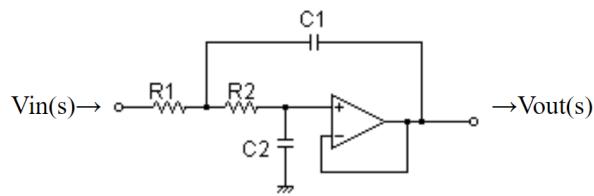
V_2 er input i subtractoren og V_1 er spændingen vi ønsker at nedjustere vores signal med.

$$V_O = 4V - 2V = 2V$$

Alle udregninger vedrørende subtractoren findes ligeledes i dokumentet analyse i bilaget.

10.1.3 Anti-aliaserings filter

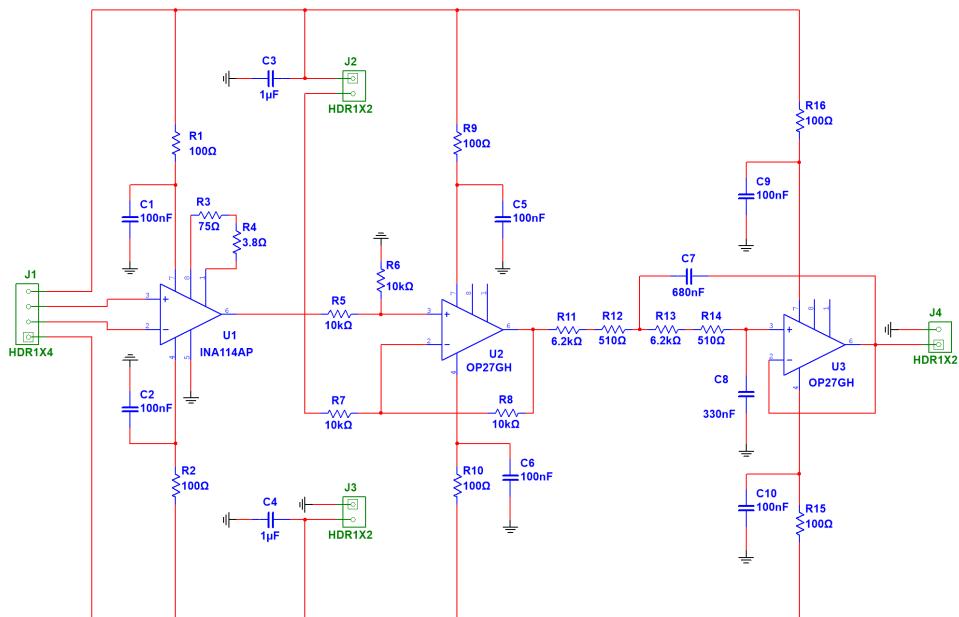
For at fjerne støj er det nødvendigt at filtrere signalet. Til systemet har vi derfor valgt at benytte et aktivt 2. ordens lavpasfilter af typen Sallen Key. Alt i alt ønskes signalet dæmpet med 90 dB , men fordi signalet allerede dæmpes 70 dB er det kun nødvendigt at dæmpe med 20 dB yderligere. Dette kan klares med et 1. ordens filter men for at være på den sikre side, har vi valgt at benytte et 2. ordens filter, der dæmper med 40 dB pr. dekade. Alt dokumentation omkring valg af filtertype, beregninger og begrundelse for skift at filter undervejs i projektet er beskrevet nærmere i dokumentet analyse i bilaget. Nedenfor ses opbygningen af et Sallen Key filter. Vi bruger igen en operationsforstærker af typen OP27G til at bygge filteret.



Figur 10.2. Sallen Key: Aktivt 2. ordens lavpasfilter

10.1.4 Printplade

Før printpladen kan designes er det nødvendigt at opbygge systemet i Multisim. Multisim danner en simuleret version af det kredsløb, der undervejs er opbygget og testet på fumlebræt. Nedenfor vises det endelige kredsløb i Multisim, som også er brugt til at designe printpladen.



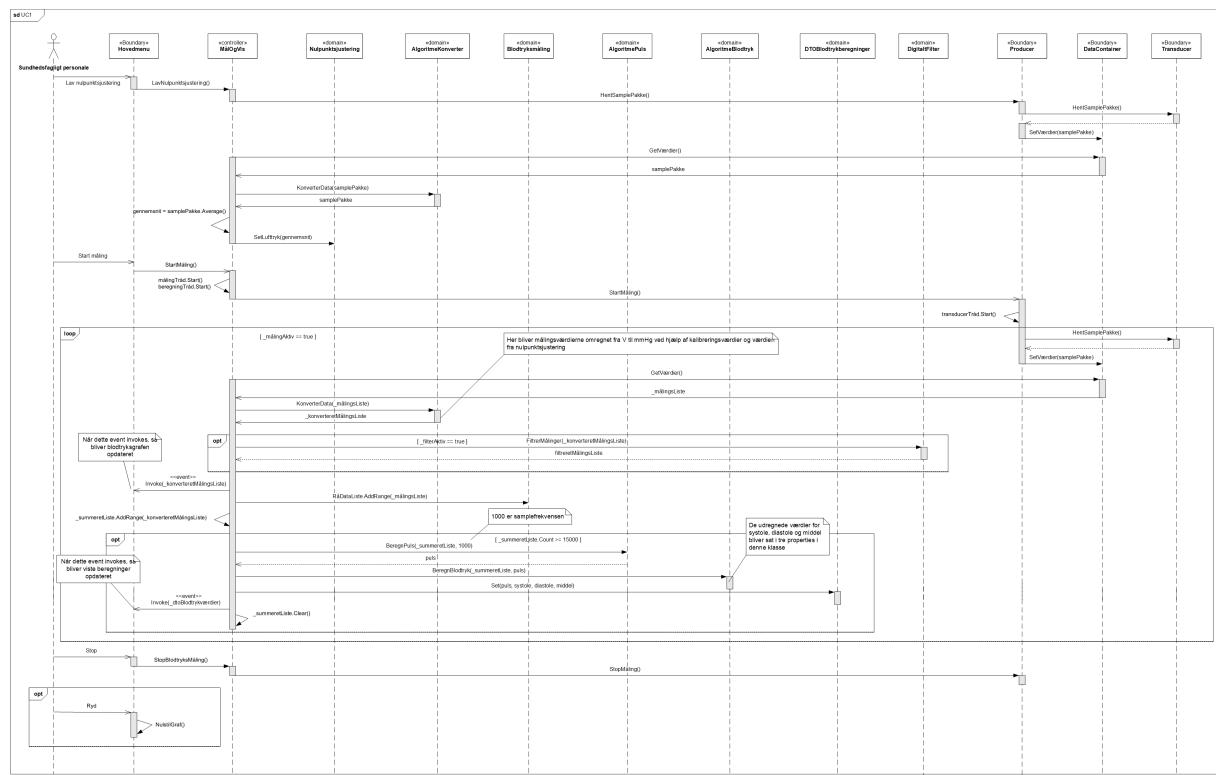
Figur 10.3. Kredsløbet i Multisim

En detaljeret beskrivelse af opbygningen af diagrammet, herunder designovervejelser, findes i dokumentet ?? i bilaget.

10.2 Softwaredesign

Dette afsnit beskriver designprocessen af softwaren. Afsnittet tager udgangspunkt i UC1, som er systemets vigtigste use case nemlig at måle og vise blodtryk. Her er der arbejdet iterativt, da det var nødvendigt at udvide softwaren mange gange i forhold til den oprindelige arkitektur. Nedenstående figur viser det endelige sekvensdiagram for UC1.

For større visning af figur henvises til bilaget.

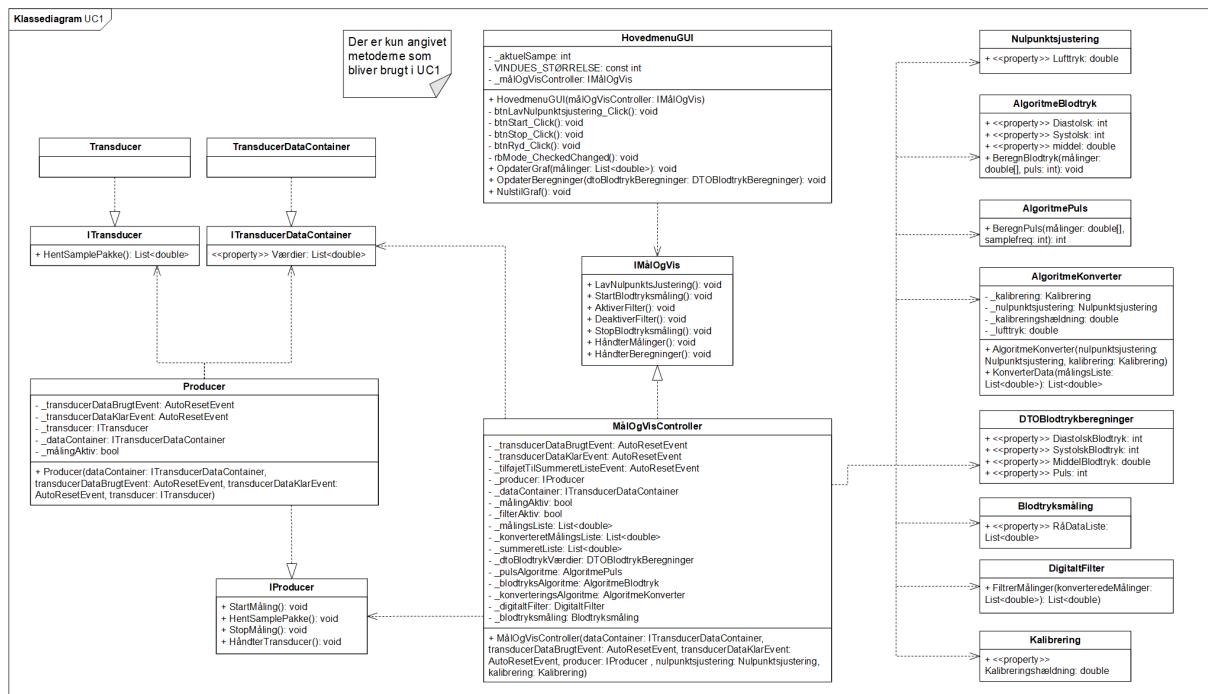


Figur 10.4. Sekvensdiagram for Use Case 1: Mål og vis blodtryk og puls

Sekvensdiagrammet viser interaktionen mellem de forskellige klasser og hvilke oplysninger der bliver sendt rundt. Den første del af diagrammet beskriver nulpunktsjusteringen, som sker inden en måling kan igangsættes. Den midterste del som er indrammet af et loop beskriver selve målingen og opdateringen af GUI'en. Der er benyttet tråde, hvilket ikke direkte kan illustreres på et sekvensdiagram. Som det er vist ovenstående, sker opdateringen af grafen inden opdateringen af beregningerne. I virkeligheden forløber opdatering af grafen, og opdatering af beregninger sideløbende i to tråde. Dette kunne været vist i et aktivitetsdiagram, men dette er ikke udarbejdet, da andet har været prioriteret højere.

Måling og beregninger sker indtil sundhedsfagligt personale vælger at stoppe målingen. Dette står beskrevet nederst i diagrammet. Ud fra sekvensdiagrammet er der udarbejdet et udfyldt klassediagram, som beskriver alle attributter og metoder der er nødvendige for at realisere Use Case 1. Klassediagrammet ses på nedenstående figur.

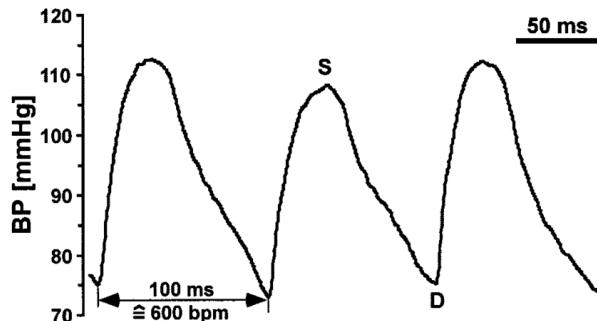
For større visning af figur henvises til bilaget.



Figur 10.5. Klassediagram udfyldt med attributter og metoder

Som der står på diagramnoten, så er der kun inkluderet metoder og attributter, som bruges i UC1. Diagrammet er forsøgt delt op i tre kolonner. Alle klasserne der ses til højre er domæneklasser, som en del af domæne package jf. Figur “Klassediagram for hele applikationen uden attributter og metoder”. I midterste kolonne er der HovedmenuGUI klassen, som er en del af PL, samt interface IMålOgVis og MålOgVisController, som er en del af BL. Det er underforstået, at klasser der implementerer et interface indeholder interfacelets metoder. Klasserne til venstre i diagrammet er en del af DAL. Ud fra klassediagrammet er det tydeligt, at UC1 er forholdsvis kompleks og indeholder rigtig mange afhængigheder. Dette gav sig til udtryk i test, da det viste sig at være meget sværere at lave unitests på denne use case end forventet. Dette er nærmere beskrevet i testafsnittet.

Puls



Figur 10.6. Puls

Et af vores must have krav er, at vi skal kunne vise systolis-, diastolisk-, middelblodtryk og puls på brugergrænsefladen. Derfor skal vores system kunne finde pulsen ud fra et blodtrykssignal. Dette kunne løses på to måder. Den ene mulighed var at måle tiden mellem to diastoler, da hjertet på denne tid har kontraheret sig og pumpet blod ud i det store kredsløb. Denne metode vil dog kun give en indikation om, hvad pulsen er, men vil ikke kunne give den nøjagtige. Den anden mulighed var at lave fourier transformation på blodtrykssignalet. Ved at lave fourier transformation vil vi flytte blodtrykssignalet fra tidsdomænet til frekvensdomænet. På frekvensspekteret vil vi kunne finde den måling med den højeste amplitude. Den måling med den højeste amplitude er den måling med mest energi i, og dermed den frekvens, der optræder flest gang i signalet. Denne måling har en frekvens, og ved at gange denne med 60, vil vi få vores puls. Vi har valgt at arbejde med denne metode i det, man beregner pulsen over en længere periode, og dermed vil det give en mere præcis indikation på pulsen.

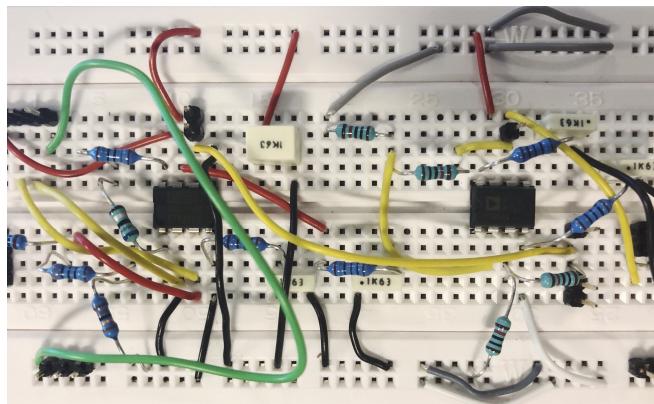
Den beregnede puls benytter vi til at beregne det systoliske-, diastoliske-, og middelblodtryk. Pulsen er med til at bestemme, hvor langt tilbage i blodtrykssignalet vi skal gå for at finde minimum, maximum og middelværdien. Minimum er det diastoliske, maximum er det systoliske og middelværdien er middelblodtrykket.

For at give et overblik til os selv om, hvordan beregninger af puls og blodtryksværdierne skulle foregå ville det have været en fordel, at vi havde lavet aktivitetsdiagrammer.

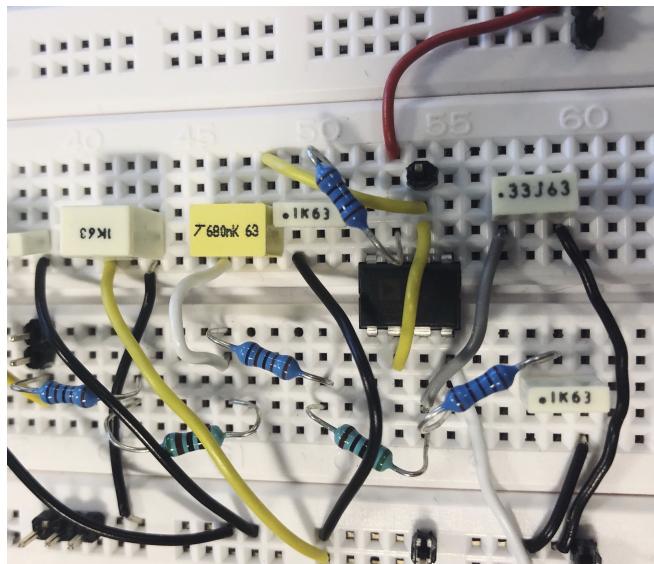
11 Implementering og test

11.1 Hardwareimplementering

Alle dele af projektets hardware er blevet analyseret og designet og derefter implementeret. Vi startede med at udregne komponenter til forstærkeren, som vi på forhånd vidste hvordan skulle opbygges. Efter beregningerne kunne forstærkeren opbygges på fumlebræt. Efter at hver del er blevet implementeret på fumlebrættet er det blevet modultestet. På denne måde er vi sikre på, at hvert delelement af projektets hardware virker som forventet. På fumlebrættet er forstærkeren, subtractoren og Sallen-Key filteret derved implementeret og modultestet enkeltvis for at se om de gav de forventede værdier. Efterfølgende blev det hele sat sammen til et kredsløb og systemet kunne derefter integrationstestes.



Figur 11.1. Opbygning af forstærker(til venstre) og subtractor (til højre) på fumlebræt



Figur 11.2. Opbygning af filter på fumlebræt

Efter integrationstesten blev kredsløbet tegnet i Multisim. Det tegnede kredsløb i Multisim blev til slut overført til ultiboard, hvor de tilhørende komponenter blev placeret på printpladen. Printet blev testet og sendt til EuroCircuits. De beregnede komponenter i analyse-delen er til slut loddet på den færdige printplade og systemet blev testet en sidste gang.

For kredsløbet i Multisim henvises til figur [10.3 på side 30](#). Nedenfor vises designet af printpladen i Ultiboard da det var klar til print samt printpladen, der er loddet med alle komponenter.

BILLEDE AF PRINTPLADE HER

11.2 Softwareimplementering

I implementeringsfasen startede gruppen ud med at lave et ”skelet” for programmeringen ud fra vores klassediagram, så det var opdelt og klar til at blive kodet efter 3 lags-modellen. Da vi blev klar til at gå i gang med at kode begav vi os til at uddelegerere forskellige programmeringsopgaver. Hvert medlem startede med at påtage sig nogle små opgaver, som var nogle basale ting. Efter udførslen af små-opgaverne blev der uddelegeret use cases, som man fik ansvaret for, at var klar og funktionel til deadline. Uddelegeringen af opgaverne skete ved brug af arbejdsmetoden scrum, se afsnit - Herefter blev de forskellige use cases sat sammen, og testet på tværs af lagene ud fra programmets funktionalitet, der er defineret i kravspecifikationen.

Hele vores system er skrevet i programmeringssproget C# og er skrevet vha. programmeringsprogrammet Visual Studios 2017. Vi har gennem 1., 2. og 3. semester skrevet i sproget C# i undervisningen, og derfor har det været oplagt at skrive i dette sprog. Alle i gruppen har endvidere downloadet programmet ”GitHub”, hvilket har gjort det muligt at alle gruppens medlemmer har kunne sidde og arbejde i det samme dokument. Dette har gjort programmeringsprocessen en hel del mere overskuelig idet vi får alle dele til at spille sammen med det samme, og ikke sætter det tilsammen til sidst, hvor der kan forkomme et store problemer.

11.3 Test af hardware

11.3.1 Modultest

Før at vi kunne teste det samlede system bestående af både hardware og software, måtte vi foretage nogle modultests af de enkelte dele i hardwaren. Forstærkeren, subtractoren samt filteret er blevet testet på fumlebrættet, og i forbindelse med disse test har vi skiftet mellem at bruge en spændingsdeler og transduceren tilsluttet vandsøjlen. Da testene var godkendt, kunne vi tilkoble alt hardwaren og lave integrationstest.

11.3.1.1 Forstærker

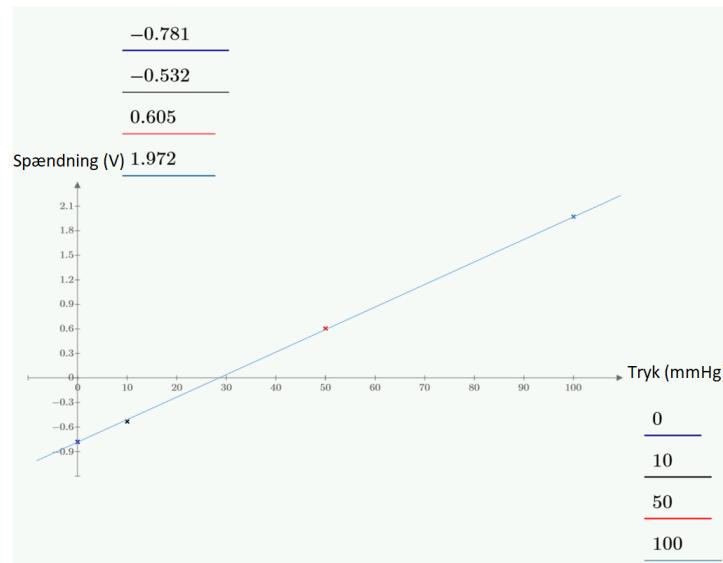
I testene for forstærkeren brugte vi en spændingsdeler, da Analog Discovery ikke kan levele lave nok spændinger uden alt for meget støj. Vi måtte have en lav spænding, da transducerens output tilsvarende er en lav spænding. Spændingsdeleren fungerede derfor som en erstatning af transduceren, hvor vi blot selv kunne bestemme hvilket type signal vi ville sende igennem forstærkeren.

Test med spændningsdeler

Vi ville teste at signalet blev forstærket 320 gange, hvilket vi gjorde ved at sende 4 V gennem spændingsdeleren samt forstærkeren, og målte på udgangen af forstærkeren. Her forventede vi at få et output på 4 V fra forstærkeren, da spændingsdelerens output var 320 gange mindre end de 4 V. Input til spændningsdeleren og output fra forstærkeren burde være det samme, hvilket stemte overens med vores faktiske resultater.

Test med transducer og vandsøje

I denne test erstattede vi spændingsdeleren med transduceren, som blev tilsluttet vandsøjlen. Der blev udført en række tests med forskellige tryk på transduceren, for at undersøge om signalet blev forstærket op 320 gange, samt teste lineariteten og nøjagtigheden af vores målinger. Nedenstående graf repræsenterer testresultaterne fra et tryk på 0, 10, 50 og 100 mmHg.



Figur 11.3. Modultest af forstærker på fumlebræt

På x-aksen er tryk i mmHg, og på y-asken er spænding i V. Det ses at der er en god lineær sammenhæng mellem tryk og spænding, samt at signalet er blevet forstærket. Da vi på vandsøjlen kun kan måle op til 100 mmHg antager vi at denne sammenhæng fortsætter kan følgende regnestykke opstilles:

$$\frac{250 \text{ mmHg}}{100 \text{ mmHg}} = 2,5$$

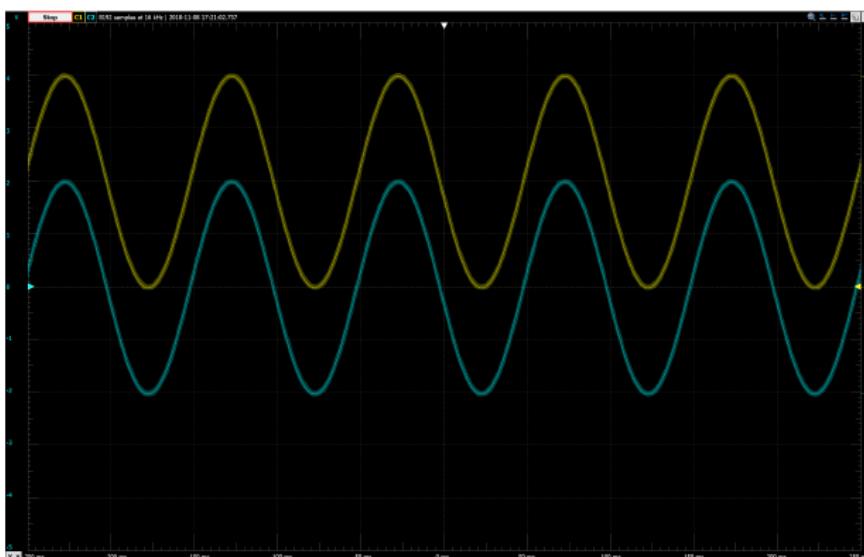
$$1,972V \cdot 2,5 = 4,93V$$

$$4,93V - 0,781V = 4,149V$$

Ud fra tidligere beregninger vil vi forvente at et tryk på 250 mmHg vil have en spænding på 4 V, hvilket stemmer godt overens med vores test. De specifikke måleresultater findes i bilag for test af hardwaren.

11.3.1.2 Subtractor

For at teste subtractoren brugte vi waveforms til at generere to signaler. Et DC-signal på 2 V, hvilket subtractoren bruger til at trække et signal ned med 2 V. Det andet signal var et sinussignal, som blev sendt gennem subtractoren. I testen forventede vi, at subtractoren ville sænke sinussignalet 2 V ned i dens amplitude. Vi sendte et sinussignal ind med et offset på 2 V, for at hæve signalet, som vi derefter vha. subtractoren kunne sænke. Vi mælte signalerne ved brug af waveforms scope funktion.

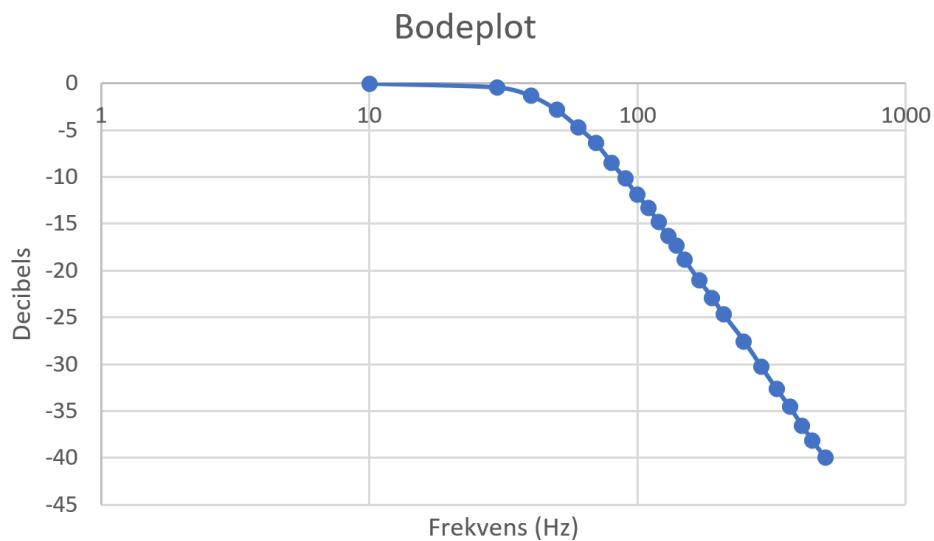


Figur 11.4. Test af subtractor

På ovenstående screenshots ses den gule kurve, som et sinussignal med en amplitude på 2 V og et offset på 2 V. Der benyttes et sinussignal, for at teste subtractorens nedeskalering af forskellige amplituder. Den blå kurve er outputtet fra subtractoren, hvilket har sænket det oprindelige sinussignal med 2 V. Subtractoren virkede derfor som den skulle, i og med at den kunne sænke signalet med de ønskede 2 V.

11.3.1.3 Filter

Da vi testede filteret, sendte vi et sinussignal ind på filteret med en amplitude på 2 V. For at undersøge filtrets dæmpning gav vi signalet 24 forskellige hertz i et range på 10 til 500 Hz. Resultaterne er påvist på nedenstående bodeplot.

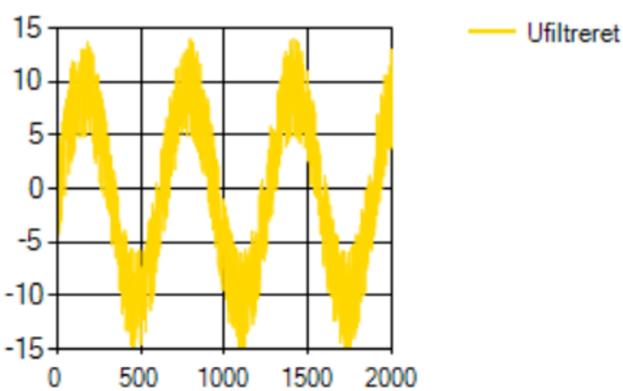


Figur 11.5. Test af filter

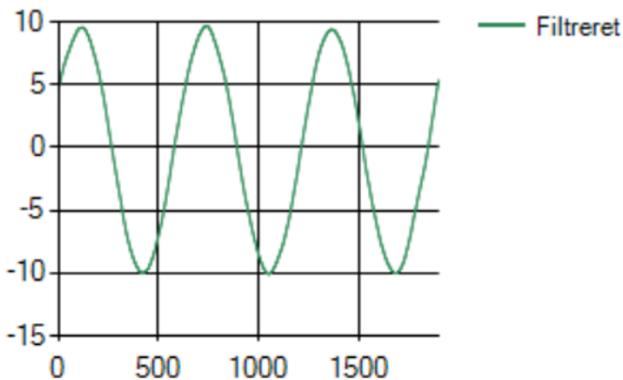
Det ses at signalet ved 500 Hz dæmper 40 dB pr dekade, hvilket stemmer overens med vores ventede resultat. De specifikke måleværdier kan findes i bilag for test af hardwaren. Kravet er at filteret skal dæmpe 20 dB over den dekade der er til rådighed, så i teorien har vi ikke brug for et 2. ordensfilter. Dog i tilfælde af at vi brugte 1. ordensfilter, skulle alt performe fuldstændigt perfekt, hvilket var en risiko vi ikke var villige at tage, og derfor benytter vi et 2. ordensfilter.

11.4 Test af software

Dette afsnit beskriver, hvordan softwaren er testet, samt hvordan dette er dokumenteret. Som udgangspunkt er ideen med softwaretest altid at lave unittest, integrationstest og til sidst systemtest. Som nævnt ganske kort i afsnittet software design, så viste det sig at være sværere at unitteste softwaren end forventet. Dette tyder på at selvom softwaren er designet ud fra ideen om lav kobling, så har dette ikke været opfyldt tilstrækkeligt. Følgende elementer er blevet unittestet: Pulsalgoritmen, blodtryksalgoritmen, det digitale filter og lineær regressionsberegning i forbindelse med kalibrering. Se dokumentet Softwaretest (eller unittest afhængig af hvad dokumentet hedder???) for dokumentation af disse tests. En af de udførte unittests er som nævnt test af det digitale filter. Dette blev gjort ved at lave et sinussignal med hvid støj, som blev sendt igennem filteret. Nedenstående figur viser sammenligning mellem det filtrede og ufiltrerede signal.



Figur 11.6. Test af digitalt filter: Ufiltreret



Figur 11.7. Test af digitalt filter: Filtreret

Som nævnt i indledningen var det meget svært at lave unittest på vores use cases. Det endte derfor med at langt det meste softwaretest har været integrationstest, hvor hele use casen er testet på én gang. Dette blev gjort ved hjælp af debuggeren i Visual Studio. Ved at sætte breakpoints i koden og følge de forskellige kald ned gennem koden kunne fejl lokaliseres på denne måde. Problemet ved at bruge debuggeren og ikke skrive en selvstændig testsuite er en mangelfuld dokumentation af disse tests. Dette giver sig til udtryk i softwaretest (eller hvad det hedder) dokumentet, som kun indeholder de beskrevne unittests og ikke integrationstest. Dette kom også til udtryk under udførelsen af accepttest, hvor flere tests ikke opførte sig som forventet, hvilket igen indikerer en utilstrækkelig systematisk tilgang til test.

12 Resultater

12.1 Resultater

Ud fra vores accepttest blev alle vores krav testet. Resultaterne giver et billede over hvilke funktioner som virkede optimalt, og hvilke der ikke gjorde.

Resultater for accepttest af Use Case 1: Mål og vis blodtryk og puls

Use Case 1: Mål og vis blodtryk og puls		Ext. 1: Digitalt filter vælges fra	
Test	Resultat	Test	Resultat
1.1	OK	1.2.1	OK
1.2	OK		
1.3	OK		
1.4	OK		

Resultater for accepttest af Use Case 2: Juster grænseværdier

Use Case 2: Juster grænseværdier		Ext. 2a: Systemet afviser justering	
Test	Resultat	Test	Resultat
2.1	OK	2.1.1	OK
2.2	OK	2.1.2	FAIL
2.3	OK	2.1.3	OK
2.4	OK	2.1.4	Ikke testet

Resultater for accepttest af Use Case 3: Alarmering

Use Case 3: Alarmering		Ext. 3a: Sundhedsfagligt personale muter alarmen	
Test	Resultat	Test	Resultat
(Hovedscenarie testet gennem andre Use Cases)		3.1.1	OK
		3.1.2	OK
		3.1.3	OK

Resultater for accepttest af Use Case 4: Gem data

Use Case 4: Gem data		Ext. 4a: Data ikke gemt	
Test	Resultat	Test	Resultat
4.1	OK	4.1.1	FAIL
4.2	FAIL	4.1.2	FAIL
4.3	FAIL		
4.4	FAIL		

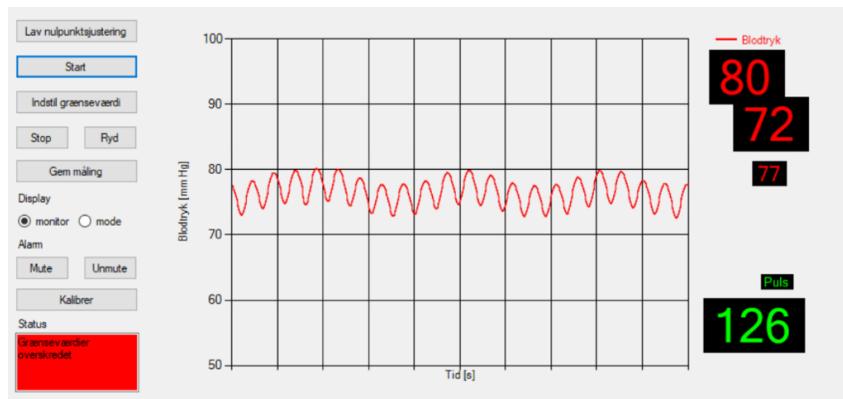
Resultater for accepttest af Use Case 4: Kalibrering

Use Case 4: Kalibrering	
Test	Resultat
5.1	OK
5.2	OK
5.3	FAIL
5.4	FAIL

Ud over de ovenstående tests af funktionelle krav er der også blevet lavet accepttests af de ikke-funktionelle krav. Resultaterne på testen ses her:

Resultater for ikke-funktionelle krav		
Krav nr.	Krav	Vurdering (OK/FAIL)
1	Den skærm der benyttes af operatøren er den skærm, hvor man kan interagere med systemet	OK
2	Den skærm, der benyttes af observatøren er den skærm, hvor man observerer grafen samt værdier for systolisk-, diastolisk-, middelblodtryk og puls.	Ikke testet
3	Systemets GUI skal indeholde elementerne beskrevet under systembeskrivelsen i dokumentet "Kravspecifikation" som findes i bilaget	OK
4	Systemets GUI skal kunne vise en graf inden for 5 sekunder	OK
5	Systemet skal kunne stoppe målingen inden for 5 sekunder	OK
6	Grafen samt de viste værdier skal kunne læses på op til 0,5 meters afstand. Farverne skal kunne skelnes af personer med farveblindhed	OK

Gruppen fik lov til at komme ud på Skejby sygehus og afprøve blodtryksmåleren på en gris. Forsøget gik ikke helt som forventet. Vi havde forventet nogle større udsving og har efterfølgende fundet ud af, at filteret havde to forkerte modstande på printpladen. Dette var skyld i en for hård filtrering af signalet og er efterfølgende blevet ændret.



Figur 12.1. Målt blodtryk på gris

12.2 Diskussion af resultater

Accepttesten gik udmærket, og vi fik valideret vores krav. Accepttesen bestod af 5 Use Cases, hvor hver Use Case indeholdt flere krav, der skulle testes. De fleste af kravene blev imødekommet, dog var der også flere som ikke gjorde. Kravene der ikke blev imødekommet skyldtes prioritering af forskellige arbejdsopgaver, hvor det var blevet nødvendigt at lægge arbejdsindsatsen ind på, at få alle "Must have" kravene opfyldt, som det første. Den første Use Case med "Mål og vis blodtryk og puls" gik efter planen, og opfyldte alle de stillede krav. I denne Use Case var det de mest grundlæggende dele af vores system, som blev testet og var derfor prioriteret højt ift. at skulle bestå alle testene. Det lykkedes os at få foretaget en nulpunktsjustering samt at få vist blodtryk og puls på en graf via brugergrænsefladen, hvor grafen både kunne vises med digitalt filter og uden.

Use Case 2 "Justering af grænseværdier" gik som vi havde forventet. Det lykkedes os at kunne justere forskellige grænseværdier, hvor der kom en fejlmeddeelse, hvis grænserne virkede usandsynlige høje/lave. En overskridning af grænseværdierne viste sig også fint i "statusboksen", som blinkede rødt og gav en alarmeringsbesked. Det lykkedes os dog ikke at få udført en ændring af grænseværdierne mens vi fik et signal ind, da testen ikke kunne udføres med signalet fra væskesøjlen. Væskesøjlen giver en puls på 0, og vil derfor gøre det umuligt for os at teste på, da der ikke kan justeres på pulsen. I stedet skulle der have været brugt en analog discovery til at sende et signal ind.

Use Case 3 "Alarmering" blev testet parallelt med andre use cases idet de aktiverede alarmerne i forskellige sammenhænge. Vi valgte derfor ikke at lave nogle specifikke test af alarmen, udover hvor alarmen mutes. Når alarmen var i gang auditivt spillede tonerne c e g – g C, hvilket var det vi havde forventet. Samtidigt blinkede "statusboksen" med en hertz på 2, og en duty cycle på 50

Det lykkedes os desværre ikke at gemme vores data, som planlagt i Use Case 4 "Gem data". Efter at have gået vores kode for "gem data" grundigt igennem står det nu klart for os, hvorfor data'en ikke bliver gemt. I koden bliver der oprettet nye instanser af hver klasse, hver gang der skal gemmes noget data. Dette medfører at data'en fra den klasse, som der gerne skulle gemmes fra ikke bliver overført til klassen, hvor vi gerne vil gemme det. Resultatet af det er at vi ikke får gemt noget data. Udover at have oprettet flere instanser af klasserne opstod der også problemer under serialisering af den gemte data.

Under testning af kalibreringen fandt vi ud af at vores system kalibrer korrekt, dog blev den lineære regression ikke udført korrekt.

Accepttesten forløb godt trods de fejl, som er opstået. Systemet opfyldte alle vores ikke-funktionelle krav, og de fleste funktionelle krav til systemet blev imødekommet. Accepttesten giver et billede af hvad vi har prioriteret højtest af krav.

13 Konklusion

Vi kan konkludere, at det har været muligt at skabe en GUI, som viser de data, som knytter sig til en invasiv blodtryksmåling. Herunder en graf samt angivelse af puls, systolisk-, diastolisk- og middelblodtryk. GUI'en indeholder derudover de ønskede funktioner, der har været muligt at implementere på baggrund af vores prioriteringer, dvs. kravene ift. at kunne mute og unmute en alarm, foretage nulpunktsjustering og kalibrering. Desværre lykkedes det os dog ikke at implementere en funktion, således blodtryksmålingen samt informationer omkring alarmer i løbet af operationen, digitalt filter information, den sundhedsfaglige persons id-nummer, patientens CPR-nummer samt start og stop tidspunkt for operationen blev gemt i en fil. Der er derfor rig mulighed for videre arbejde ift. denne funktion.

Hovedparten af de valg, som vi har truffet ift. design og prioritering af funktioner, er sket på baggrund af den tænkte brugssituation. På en operationsstue, er det vigtigt, at blodtryksgrafen, pulsen, systolisk-, diastolisk- og middelblodtryk er tydelige. Dog er det ikke nødvendigt, at de kan ses på lang afstand. Dels fordi der er to skærme i systemet, og fordi den sundhedsfaglige person ikke vil være på lang afstand til skærmen. Derudover er der taget højde for, at den sundhedsfaglige ikke har øjnene på skærmene hele tiden, hvilket er årsagen til, at vi har valgt at give systemet alarmlyde i henhold til ISO-standarden 60601-1-8, når enten pulsen eller blodtrykket falder eller stiger i forhold til de indtastede grænseværdier. På denne måde kan det sundhedsfaglige personale gøres opmærksom på en muligvis farlig situation.

Igennem processen, har vi som gruppe valgt at samarbejde om udarbejdelse af kravspecifikation og accepttest. Herved har hvert enkelt medlem hvert en del af de valg, som er blevet truffet og spillet en rolle ift. implementering og beskrivelse heraf. Til udarbejdelse af systemets hardware og software del, har gruppen delt sig i et software og hardware team for at skabe større effektivitet. For at de to teams har haft et billede af, hvad hinanden har været i gang med, har vi benyttet os af standup-møder to gange om ugen. Derudover har vi benyttet scrum til at få en struktur på de opgaver, der skulle løses, for at vi kunne udforme en prototype af vores system. Begge arbejdsformer samt udviklingsmodellen har skabt en fælles følelse af ejerskab over projektet og betydet, at alle i gruppen har haft mulighed for at udvikle sine kompetencer.

14 Fremtidigt arbejde

I fremtidige versioner af blodtryksmåleren er der dele af systemet, som ville være oplagt at lave forbedringer af både i henhold til funktionalitet og brugervenlighed.

Gemme-funktion

I vores program foregår processen med at gemme oplysninger fra operation til slut, hvilket er meget risikabelt idet at vi ikke imødekommer naturlige uforudsigelige fejl, såsom strømafbrydelse af systemet. Denne uheldige situation kan blive en mindre kritisk situation, hvis der i løbet af operation bliver gemt data hver 5. minut fremfor kun at gemme til sidst. Uover at det kunne være optimalt at gemme flere gange i løbet af en operation burde den fremtidige udgave også gøre det muligt at tilføje patienten til en database, hvor data'en kan blive gemt fremfor at gemme i en tekst fil. På den måde er alle målinger, der er blevet foretaget under operationen hele tiden koblet til patientens oplysninger. Ydermere ville det være ideelt, hvis der i programmet var mulighed for at indhente tidlige målinger en patient, når patienten er blevet tilføjet i systemet. Det vil gøre at man kan gå ind og analysere tidlige målinger. En sammenkobling med den elektroniske patientjournal ville gøre at alle målinger automatisk blev gemt i den pågældende patients journal og kunne være med til at optimere programmet.

Teknisk personale

Det tekniske personale har til opgave at udføre kalibreringen korrekt. Når der er tid til en kalibrering skal en fra det tekniske personale udføre kalibreringen uden at logge ind. For at optimerer denne proces ville det være godt, hvis der blev knyttet et ID til det tekniske personale. Dette vil gøre at det tekniske personale logger ind med et ID hver gang der skal foretages en kalibrering, og dermed kant tracke, hvem der har udført processen.

Brugergrænseflade

Ifølge standarderne 60601-1-8, så er der nogle krav til symbolerne der bruges på brugergrænsefladen. På vores brugergrænseflade benytter vi primært tekst i stedet for symboler, og det kunne derfor være en god idé i fremtiden at gøre brug af symboler i stedet for tekst.

Alt efter indretning af operationsrummet ville det være optimalt, hvis man skifte farven for baggrunden på brugergrænsefladen. Når systemet bliver anvendt om natten kunne det derfor være behageligt, for både patient og personalet at baggrunden er sort, da der vil være mindre lys fra skærmen. Der kunne med fordel vælges en lysere baggrund om dagen, så det er nemmere at se skærmen.

Hardware

Rent hardwaremæssigt kunne der laves nogle forbedringer i fremtiden. Først og fremmest kunne det være oplagt, hvis strømforsyningen kom fra computeren i stedet for at have Analog Discovery som strømforsyning. Dette vil gøre at systemet vil bestå af mindre dele, da Analog Discovery kan blive unødvendig idet computeren nu kan virke, som vores strømforsyning. For at det kan

lade sig gøre kræver det, at printpladen laves større, så der er plads til en spændingsdeler på printet. Spændingsdelen vil dele de 5 volt, som computeren afgiver så subtraktoren stadig får 2 volt. Vi har i vores printplade fået lavet 1 hul i hvert hjørne, så den er klar til at blive skruet fast inde i en boks, som vil have funktionen at beskytte vores printplade. Det kunne derfor være en smart ting at få udarbejdet i fremtiden.

15 Bilag

Til projektet hører en række bilag. Bilagene er inddelt i to hovedmapper, projekt og proces. Nedenfor ses bilagene nummereret. Trykkes der på et af nedenstående bilag åbnes bilaget såfremt det findes i PDF-format.

Projekt

- Bilag 1:** "Analyse"
- Bilag 2:** "INA114 datablad"
- Bilag 3:** "NI-DAQ 6009 datablad"
- Bilag 4:** "OP27G datablad"
- Bilag 5:** "Tryktransducer datablad"
- Bilag 6:** "Doxygen genereret dokumentation"
- Bilag 7:** "Kravsspecifikation"
- Bilag 8:** "Logbog"
- Bilag 10:** "Multisim"
- Bilag 11:** "Ultiboard"
- Bilag 12:** "Source code"
- Bilag 13:** "Modultest"
- Bilag 14:** "Integrationstest"
- Bilag 15:** "Accepttest"

Proces

- Bilag 16:** "Gantt-diagram"
- Bilag 17:** "Gruppemøder"
- Bilag 18:** "Procesbeskrivelse"
- Bilag 19:** "Samarbejds aftale"
- Bilag 20:** "Vejledermøder"

Litteratur

- [1] Aleris Hamlet. Sådan fungerer dit hjerte. URL <https://www.aleris.dk/vi-tilbyder/aleris-hamlet-hospitaler/hjertesygdomme/sadan-fungerer-dit-hjerte/>. Sidst besøgt d. 13-12-2018.
- [2] Chegg Study. Wiggers diagram. URL <https://www.chegg.com/homework-help/questions-and-answers/using-wiggers-diagram-calculate-following-heart-rate-b-stroke-volume-c-cardiac-output-d-me-q30345183>. Sidst besøgt d. 13-12-2018.
- [3] Vejledning til udfærdigelse af projektrapporter. Ase-modellen. URL http://studerende.au.dk/fileadmin/user_upload/Vejledning_til_udfaerdigelse_af_projektrapporter_v1.5.pdf. Sidst besøgt d. 13-12-2018.