Høgskulen
på Vestlandet

BACHELOROPPGAVE:

BO19E-31 Dataoverføring for automatisk integritetsvurdering av fleksible stigerør

Stein Rune Alvheim
Eivind Nilsson Førde
Øystein Riis Mathisen

31. mai. 2019

Dokumentkontroll

Rapportens tittel: BO19E-31 Dataoverføring for automatisk integritetsvurdering av fleksible stigerør	*Dato/Versjon* 31. mai. 2019/3.00
	Rapportnummer: B019E-31
Forfatter(e): Stein Rune Alvheim Eivind Nilsson Førde Øystein Riis Mathisen	*Studieretning:* HKOM16
	Antall sider m/vedlegg 53
Høgskolens veileder: Kjell Eivind Frøysa	*Gradering:* Åpen
Eventuelle Merknader: Vi tillater at oppgaven kan publiseres.	

Oppdragsgiver: 4Subsea AS	*Oppdragsgivers referanse:*
Oppdragsgivers kontaktperson(er) (inklusiv kontaktinformasjon): Henrik Tvedt (01.01.2019 – 30.04.2019) Magnus Alme Aasheim (magnus.alme.aasheim@4subsea.com)	

Revisjon	**Dato**	**Status**	**Utført av**
0.1	15.01.19	Første utkast	Stein
1.0	31.01.19	Forstudie satt inn	Øystein
1.1	01.05.19	Utkast 4.2.4 Innkapsling og 4.2.5 Av/På knapp	Stein
1.1	03.05.19	Startet på 4.2.5 Strømforsyning	Stein
1.1	03.05.19	Kap. 5 Testing	Eivind
1.1	03.05.19	4.1 Software	Øystein
1.1	03.05.19	Appendiks B	Øystein
1.2	15.05.19	4.3.6 Strømforsyning og Ap. D1	Stein
1.3	21.05.19	Kobling ++	Øystein
1.3	21.05.19	Kap. 6. Diskusjon	Stein
1.3	21.05.19	Kap. 4.2. Hardware	Stein
2.0	22.05.19	Utkast	Øystein og Stein
2.1	24.05.19	Retter opp skrivefeil/formateringsfeil	Øystein
2.2	25.05.19	Lagt til div	Øystein
2.3	30.05.19	Kap. 7. Konklusjon	Stein
3.0	31.05.19	Endelig versjon	Alle

Forord

Denne oppgaven er en videreføring av bacheloroppgave skrevet av Osmund Bjørkum Vereide og Vebjørn Aasheim i 2018. Oppgaven bygger på funn og erfaringer gjort fra tidligere oppgave og er en fortsettelse på arbeidet gjort i 2018 med noen modifikasjoner på bland annet kravspesifikasjoner.

Oppgaven er utformet av -og utføres på vegne av 4Subsea AS.

Vi ønsker å takke Henrik Tvedt og Magnus Alme Aasheim fra 4Subsea AS samt vår internveileder Kjell Eivind Frøysa ved Høgskulen på Vestlandet for gode råd og veiledning underveis i prosjektet.

Sammendrag

Oppgaven er en fortsettelse på fjorårets bacheloroppgave der man skal lage en løsning for dataoverføring av måledata fra «Portable Annulus Tester» (PAT), som er en enhet utviklet av 4Subsea for å utføre målinger for integritetsvurdering av fleksible stigerør.

Fra arbeidet gjort på tidligere bacheloroppgave er kravspesifikasjoner og prioriteringer noe endret. Det opprinnelige kravet om at utstyret skulle kunne brukes innenfor ex-sone er frafalt da det ville bli dyrt og urealistisk å få utstyret ex-godkjent innenfor scoopet av bacheloroppgaven. Oppgaven blir da å lage en komplett løsning som fungerer utenfor ex-sone eller eventuelt som en prototype der man senere vil realisere en løsning som kan fungerer innenfor ex-sone.

Videre så ønsket 4Subsea at primærmålet skulle være å realisere en løsning der dataoverføringen blir gjort over satellitt. Løsning for dataoverføring over Wi-Fi og 4G skulle være sekundærmål dersom det skulle være mulig å realisere alle løsningene innenfor tidsrammen vi har tilgjengelig.

Fra tidligere bachelor var løsningen de hadde kommet lengst med en løsning som baserer seg på en Raspberry Pi 3 B+ sammen med en RockBLOCK 9603 enhet for kommunikasjon med satellitt. De hadde laget en enkel løsning som fungerte, og utstyret var tilgjengelig. Derfor konkluderte vi i forstudiet sammen med 4Subsea at dette vil være den beste løsningen å fortsette på.

Oppgaven vår har vært å skrive kode for å håndtere kommunikasjon mellom HMI på PAT og Raspberry Pi samt kode for håndtering av seriell kommunikasjon mellom RockBLOCK og Raspberry Pi. Vi har møtt på en del problemer ved bruk av ferdige biblioteker for kode, og utfordringer med eldre versjon av .NET rammeverk på HMI i PAT.

Det har også vært en del arbeid knyttet til hardware. Vi har valgt å lage en 3D-modell av innkapsling, og det har vært en del mindre oppgaver som lodding, kobling og montering.

I oppgaven har vi realisert en enkel løsning for kommunikasjon mellom PAT og RockBLOCK sin server. Det er fremdeles en del som gjenstår for at løsningen skal være komplett samt at oppgaven har heller ikke tatt for seg realisering av løsning for Wi-Fi og 4G.

1 Innhold

1 Innledning

1.1 Oppdragsgiver

4Subsea ble grunnlagt i 2007 under navnet Færder Engineering AS før de i 2008 skiftet navn til 4Subsea AS [1]. De utvikler og leverer ingeniørtjenester samt service løsninger for kunder innen subsea og fornybar energi. Hovedsegmentet av tjenester som leveres er service løsninger for fleksible stigerør for olje og gass industri. Kundeporteføljen består av de fleste store operatørene innenfor olje og gass samt flere store leverandører av undervannsutstyr. Ved siden av å levere tjenester for olje og gass industri leverer de også løsninger for offshore vindpark installasjoner.

Hovedkontoret til 4Subsea AS ligger i Asker. Ved siden av dette har de også kontorer i Bergen, Kristiansand, Macaè og Rio de Janeiro. 4Subsea har også tilstedeværelse gjennom samarbeid med Ashtead Technology i byer som Aberdeen, Abu Dhabi, Halifax, Houston og Singapore [2].

4Subsea AS sikter på å være en pioner innen digitalisering av olje, gass og offshore vindparker. De leverer løsninger på en unik digital plattform der kundene får presentert måledata fra installasjonene sine på en oversiktlig måte sammen med rapporter og anbefalinger fra 4Subsea sine spesialister [3].

4Subsea fikk som de fleste andre aktører innen olje og gass merke nedgangstidene og gikk med 2,6 millioner i underskudd (før skatt) i 2016. De har i midlertidig klart å ta seg opp igjen i 2017 og endte med et overskudd på 883.000 før skatt [4].

1.2 Problemstilling

Oppgaven er en fortsettelse på arbeid påbegynt i en tidligere bacheloroppgave der det har vært sett på måter å muliggjøre kommunikasjon mellom 4Subsea sin Portable Annulus Tester (PAT) og deres Azure server. Vår oppgave blir å sette oss inn i funn og arbeid som er gjort for å kunne fortsette arbeidet ved å realisere en fungerende løsning. Etter erfaringer fra fjorårets bachelor ønsker oppdragsgiver å forandre noe på kravspesifikasjonene. I første omgang ønskes det at vi skal realisere en løsning som sender måledata fra PAT til Azure server over Iridium satellitt nettverket. Når dette er utført og dersom tiden strekker til ønsker oppdragsgiver at vi ser på løsninger som også kan sende data over Ethernet, Wi-Fi og 4G. Oppdragsgiver har også kommet frem til at det blir urealistisk å realisere denne løsningen innenfor ATEX sone 2. Gruppen vår trenger altså ikke å forholde seg til at utstyr skal kunne godkjennes for bruk innen ATEX sone.

2 Kravspesifikasjon

Oppdragsgiver ønsker at vi utvikler en kompakt portabel løsning bestående av maskinvare og programvare som muliggjør overføring av data via satellitt, 4G, Wi-Fi eller Ethernet. Løsningen skal være en enhet som pakkes i en kompakt innkapsling som skal være støt- og sprutsikker. Enheten må enkelt kunne kobles opp mot HMI på PAT når denne er flyttet ut av ATEX sone 2. Enheten skal styres via HMI og alle meldinger til bruker skal vises på HMI.

Primære mål:

- Muliggjøre overføring av mindre mengder data via satellitt.
  Det skal sendes et lite utdrag av måleresultatet, hvor størrelsen på dataen er på rundt 2kb.
- Pakkes i en portabel enhet der man skal ha fokus på størrelse og vekt.
  Utstyr blir pakket og sendt i en koffert til kunde, slik at størrelsen og vekt må være tilpasset dette.
- Kapsling ift. IP-grad bør være lik eller høyere enn IP54.
  Det vil si at kapslingen skal være støv beskyttet og ha beskyttelse mot sprut fra alle kanter som et minimum. Se appendiks F.2 IP-grad for mer utdypende informasjon.
- Robusthet
  Utstyret skal være robust og skal kunne forventes å tåle påkjenningene fra miljøet det skal benyttes i.
- Initieres fra HMI
  Bruker skal via touch display kunne velge om data skal sendes via satellitt, Ethernet, Wi-Fi, 4G eller hentes ut via USB.

Sekundære mål:

- Muliggjør 4G og Wi-Fi overføring av «større» mengde data.
  Her sendes hele måleresultatet over 4G eller Wi-Fi til 4Subsea sin Azure server.
- Muliggjør Ethernet overføring av «større» mengde data.
  Her sendes hele måleresultatet over Ethernet til 4Subsea sin Azure server.

3 Analyse av problemet

3.1 Utforming av mulige løsninger

Løsningsforslag baserer seg på funn og erfaringer gjort i bacheloroppgaven fra 2018, samt mindre endringer i kravspesifikasjoner fra oppdragsgiver. Fra fjorårets oppgave var det to forslag til mulige løsningen som vi ønsket å ta med videre i vårt prosjekt. Begge løsningene inneholder et datakort som kobles opp mot ett satellitt modem for kommunikasjon med Azure server via iridium satellitter. For begge løsningsforslagene vil enheten kobles opp mot HMI via Ethernetkabel og enhetene styres via GUI på HMI.

3.1.1 Raspberry Pi 3 Model B+ m/RockBLOCK 9603

Det ble i bachelor fra 2018 besluttet å gå til innkjøp av Raspberry Pi (RPi) og RockBLOCK 9603. Med disse komponentene har vi alt utstyr for å etablere en satellitt link og Wi-Fi forbindelse tilgjengelig. Det som mangler for å lage en komplett løsning er å gå til innkjøp av en 4G modul til RPi. Fordi RPi har eget minnekort vil den være avhengig av en sikker måte på å bli skrudd av etter bruk slik at minnekortet ikke blir korrupt. RPi har støtte for flere programmeringsspråk, men det vil være naturlig å gå for Python da det er et enkelt språk å lære seg, samt at det er passende fordi vi kun skal lage et script.

3.1.2 Adafruit Feather HUZZAH ESP8266 m/RockBLOCK 9603

Det ble også gått til innkjøp av Adafruit Feather med modul for Ethernet. Man brukte da RockBLOCK som allerede var kjøpt inn for å teste denne løsningen. Under testing fikk forrige gruppe problemer med antall bytes de kunne sende over seriell linken mellom RockBLOCK og Adafruit Feather. De fant ingen direkte løsning på problemet, slik at hvis man skal se videre på denne løsningen må man starte med feilsøking. Det ble derimot pekt på at de kunne være feil med Adafruit Feather, hvor løsning som ble presentert er at man gikk til innkjøp av en nyere modell. Adafruit er en Arduino enhet, noe som betyr at programmeringsspråket som er brukt er C++. Dette er noe alle i gruppen har brukt før, slik at det krever minimalt med opplæring av oss.

3.1.3 Vurderinger i forhold til verktøy og HW/SW komponenter

De ulike løsningene skissert ovenfor vil kreve kode skrevet i ulike programmeringsspråk. Det vil for oss være naturlig å bruke «Visual Studio 2017» ettersom det er en IDE vi kjenner godt samt at den støtter alle språkene vi trenger å skrive kode i, slik som Python, C# og C++. Uansett valg av løsning vil vi måtte benytte oss av et program for å gjøre oppslag i SQLite database med måledata lagret på HMI. Vi ble anbefalt av 4Subsea å bruke «DB Browser». For programmering og opplastning av kode for HMI må vi bruke iX Developer som er en proprietær programvare fra Beijer.

For innkapsling har vi valgt å lage en 3D-modell av boksen hvor utstyret skal monteres. For å designe 3D modell vil vi bruke «Autodesk Fusion360» som er et kraftig CAD verktøy for design av 3D-modeller. For å produsere en demomodell av innkapsling vil vi benytte oss av en «Creality Ender-3» 3D-printer som vi har tilgjengelig.

3.2 Konklusjon for valg av løsning

For løsning basert på Adafruit Feather må vi gjøre en del feilsøking for å finne ut om vi kan bruke kortet som allerede er kjøpt inn, eller om vi må se på nye modeller og bestille disse.

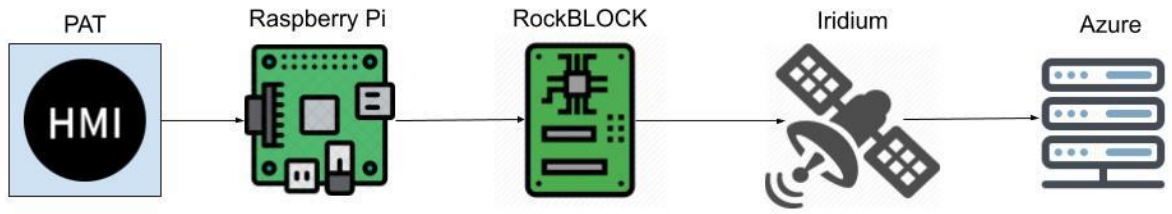
Vår anbefaling blir derfor å fortsette på arbeidet som er gjort med å realisere en løsning med Raspberry Pi Model B+ koblet til RockBLOCK 9603. Årsaken til at vi anbefaler å fortsette utviklingen av denne

løsningen er fordi dette var løsningen tidligere bachelorgruppe var kommet lengst i å realisere og fant mest lovende. For denne løsningen er også nødvendige komponenter foruten 4G modul bestilt og tilgjengelig slik at vi vil komme raskt i gang med å løse oppgaven.

4 Realisering av valgt løsning

I realiseringen vil vi presenterer en løsning for satellitt, fordi 4Subsea ønsker at det er dette vi skal prioritere. Under realiseringen av oppgaven ble det tatt hensyn til at det skulle være muligheter for Wi-Fi, 4G og Ethernet senere.

I Figur 1 nedenfor ser vi alle leddene i overføringen. Etter at data er sendt fra RockBLOCK enheten og til Iridium nettverket er det 4Subsea som håndterer dataen. Det vil si at de henter ut data som er sendt til RockBLOCK sin server og sender dette videre til 4Subsea sin Azure server. Det er ikke satt noe krav til format på dataen, men det er ønskelig at PAT enheten nummereres slik at man kan skille mellom de forskjellige enhetene.

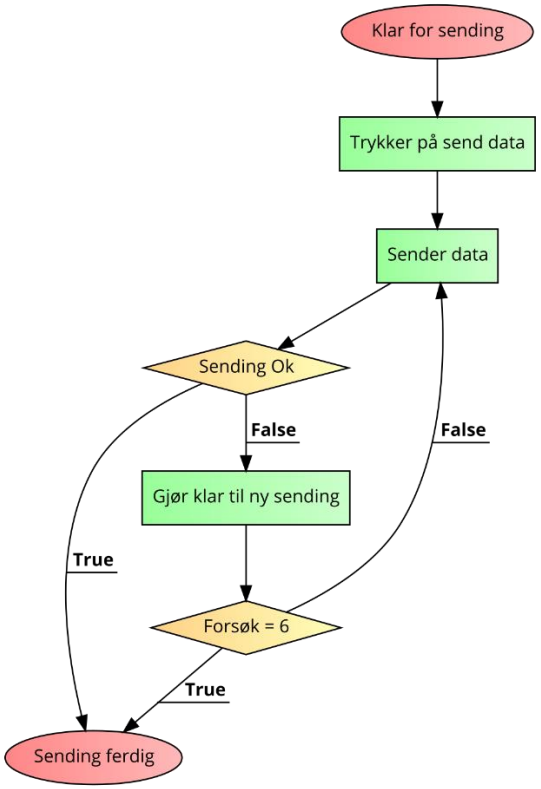


Figur 1 – En oversikt over alle leddene i systemet. Iridium består her av både satellitt, mottaker på bakken og en RockBLOCK server.

4.1 Virkemåte

I oppgaven ble det ikke gitt noen retningslinjer til ønsket virkemåte, annet enn det som kommer fram i kravspesifikasjonene. Slik at løsning vi leverer er basert på våre erfaringer og ønsker til løsning. I arbeidet med å lage utformingen av prosjektet ble det tidlig klart at vi måtte forholde oss til noen enkle punkter.

- Intuitivt for bruker
- Feilhåndtering
- Plug and play



Figur 2- Flowchart som viser virkemåten til programmet veldig forenklet.

I den ferdige løsningen vil RPi og RockBLOCK bli levert i en boks som skal kobles til HMI via en kabel. Etter at dette er koblet sammen må bruker vente i noen minutter fordi RockBLOCK må lade opp kondensatorer og RPi må starte opp operativsystemet og scriptet.

Når dette er klart kan bruker navigere seg fram til en side på HMI hvor det skal komme klart frem at det er snakk om satellittkommunikasjon (se Figur 3). På denne siden er det en knapp for start av sending og en boks med tilbakemeldinger om status på sendingen.

Figur 3 – Bilde av HMI hvor bruker kan starte en sending og få tilbakemeldinger om sendingen.

Når bruker trykker på knappen for å sende over satellitt vil den nødvendige dataen bli overført via satellitt. Bruker vil da få tilbakemeldinger i boksen om hvordan overføringen går, se appendiks F.1 for mer informasjon knyttet til disse tilbakemeldingene.

En overføring foregår ved at programmet vil prøve å sende inntil 6 ganger avhengig av om en sending er vellykket eller ikke. Dette fordi det kan være flere grunner til at en sending ikke er vellykket [5]. I de forskjellige forsøkene på en sending vil tiden mellom sendingen gå fra 0-30 sekunder til siste sending som vil vente 5 minutter før den prøver. Det vil si at hvis en komplett overføring ikke går igjennom vil det gå ca. 7 minutter før bruker kan prøve på nytt.

4.2 Software

Siden forrige gruppe valgte å bruke Ethernet med Socket som kommunikasjon mellom HMI og datakort, ble det naturlig for oss å se videre på arbeidet de startet med.

De valgte at HMI skulle være klienten og RPi skulle være server, noe vi var enig i at er den beste løsningen fordi det å gjøre motsatt vil kreve en del ekstra feilhåndtering og litt mer avansert scripting.

Fordi PAT enheten er noe 4Subsea leier ut, den er stor og de bruker programvare som krever lisens er det noe vi måtte sitte og jobbe med ut hos dem. Fordi dette var vanskelig i praksis ble det tidlig klart at å simulere en PC som HMI ble mye enklere for oss.

Det ble gjort ved at HMI ble simulert som en Windows Presentation Foundation (WPF) applikasjon, fordi HMI kjører som en WPF applikasjon. Vår ide var at dette enkelt skulle konverteres til fungerende kode på HMI med hjelp fra 4Subsea.

4.2.1 Raspberry Pi (server)

Utgangspunktet var et enkelt Python script hvor det var lagt inn socket kommunikasjon mot HMI. Scriptet tok imot en enkelt melding og sendte dette videre til den delen som håndterer kommunikasjonen med RockBLOCK. Vår oppgave ble da å videreutvikle og legge til disse punktene:

- Håndtering av flere linjer med data fra HMI
- Fra script til objektorientert (OOP)

Socket kommunikasjonen er oppnådd ved å bruke Transmission Control Protocol/Internet Protocol (TCP/IP).

Ved prosjektets slutt er alle punktene lagt til og hånderting av enkle feil er på plass. Det er også lagt til rette for at server kan ta imot tre forskjellige typer sendinger. Det er satellitt, 4G og Wi-Fi. Men det er kun lagt inn kode for satellitt løsning.

Det er ikke implementert noe form for feilhåndtering dersom det oppstår en feil som fører til at operativsystemet tvinger serveren til å avslutte. Dersom det skulle oppstå en feil må man starter RPi på nytt via start/stoppbryter.

4.2.2 HMI (klient)

På HMI var det det laget et enkelt C# program med socket kommunikasjon mot RPi. Her var det hardkodet inn en melding som ble sendt til satellitt. Vår oppgave ble da å videreutvikle og legge til disse punktene.

- Håndtering av flere linjer med data til RPi
- Lese data fra SQLite database
- Vise responsen fra RockBLOCK/RPi
- Tråder (thread)

Som det ble skrevet i innledningen til kapittelet simulerte via programmet som skal kjøre på HMI med en PC. Dette ble gjort ved å bruke Microsoft sitt rammeverk .NET 4.0 og nyere. I denne løsningen er alle punktene implementert og fungerer slik det er tenkt. Det er i denne løsningen lagt til rette for at man kan sende data via 4G eller Wi-Fi, men dette er ikke gjennomført og må løses ved neste prosjekt.

Utfordringene kom når vi skulle legge inn det ferdig utkastet til løsningen i iX Developer. iX Developer utgaven som er brukt er versjon 2.32, med rammeverket .NET Compact Framework (.NET CF). Slik at deler av koden ikke var kompatibel med rammeverket som kjører på HMI.

Det vil si at HMI koden må modifiseres til å være kompatibelt med .NET CF. Dette kombinert med at det har vært utfordrende å finne god dokumentasjonen for scripting på Beijer sitt utstyr gjør at det har vært vanskelig å finne en løsning.

Vi har ved avslutning av prosjektet klart å lage en fungerende løsning som leser data fra databasen og sender dette videre til RPi.

4.2.3 Raspberry Pi (RockBLOCK script)

Kommunikasjonen mot RockBLOCK foregår med en FTDI USB kabel [6]. For å håndtere kommunikasjonen med RockBLOCK har vi brukt et bibliotek som heter pySerial [7].

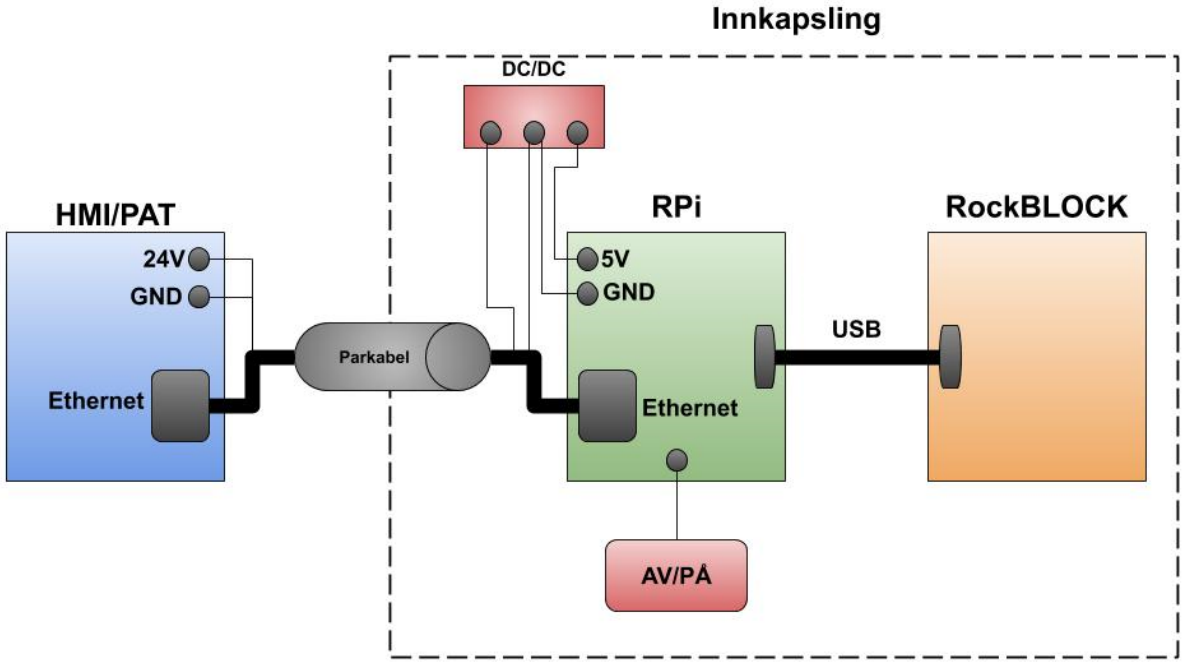
Fordi vi ønsker at kommunikasjonen mot RockBLOCK skal være automatisert er det tatt i bruk et bibliotek fra Makersnake [8]. Bibliotek er skrevet i Python 2, og ble tidlig en utfordring for oss. Vi ønsket å skrive i Python 3 fordi den offisielle støtten til Python 2 avsluttes 01.01.2020. Vi valgte derfor å skrive et helt nytt bibliotek for kommunikasjon med RockBLOCK i Python 3 med inspirasjon fra Makersnake.

I dette biblioteket er det lagt inn støtte for mange av funksjonene som er tilgjengelig i Iridium biblioteket [9]. Det er ikke lagt inn noen funksjoner som gjør at man kan motta meldinger fra satellitt, slik at dette må legges til senere ved behov.

Når data blir sendt til RockBLOCK skal det gjøres en sjekksum av teksten, som skal sendes til RockBLOCK for validering. Vi har hatt problemer med at sjekksummen vi sender og den som kalkuleres på RockBLOCK ikke er lik når det sendes større meldinger med tekst, slik at vi kan ende opp med å kun få sendt deler av dataen.

Vi har ved prosjektets slutt ikke klart å finne ut hva som gjør at det ikke går, slik at dette er noe man må se på videre.

4.3 Hardware



Figur 4 – Forenklet koblingsskjema som viser hvordan de forskjellige delene er koblet sammen.

HMI/PAT til RPi: Figur 4 viser at fra PAT får vi 24V og GND. Dette kobles til en DC/DC omformer, se kapitel 4.3.6 for videre utdyping. Fra omformeren går det GND og 5V videre til RPi. Ethernet og lederne til strømforsyning skal kombineres i en 5 pars skjermet kabel.

RPi til RockBLOCK: Siden dette er inne i en lukket boks blir det brukt FTDI/USB kabel som følger med RockBLOCK.

4.3.1 Raspberry

Raspberry Pi 3 Modell B+	
Prosessor:	64-bit SoC @ 1.4GHz
Minne:	1GB SDRAM
Lagring:	Micro SD-Kort
Kommunikasjon:	2.4 GHz og 5GHz wireless LAN
	Gigabit Ethernet RJ45
	Bluetoot 4.2, BLE
Tilkoblinger:	4 USB 2.0 Porter
	HDMI
	40-pin GPIO header
Strømforsyning:	5V/2.5A DC micro USB

Figur 5 - Raspberry Pi spesifikasjoner.

4.3.2 RockBLOCK

RockBLOCK 9603	
Kommunikasjon:	Iridium Modem, 9603 short burst transceiver
	Iridium Antenne, 1621Mhz tuned patch antenne
Tilkoblinger:	FTDI/USB
	SMA connector, for ekstern antenne
Strømforsyning:	5V/100mA Minimum

Figur 6 - RockBLOCK spesifikasjoner.

4.3.3 HMI

Beijer iX T7BR HMI	
Kommunikasjon:	Ethernet 1x10/100 RJ45
	Ethernet 1x10/100/1000 RJ45
Tilkoblinger:	2 Port 9pin DSUB
	2 USB 2.0 Port
Strømforsyning:	12V / 24V

*Oppkobling i PAT muliggjør kun kommunikasjon med RPi over Ethernet.

Figur 7 - iX T7BR HMI spesifikasjoner.

4.3.4 Innkapsling

Fra kravspesifikasjonen for oppgaven har vi at enheten vår skal pakkes i en portabel enhet, der vi må tenke på størrelse og vekt. Enheten skal være robust og ha en innkapsling med IP-grad som bør være høyere enn IP54 (Appendiks F.2 7F.2 IP-grad).

For at innkapslingen skal bli så liten som mulig, samt ha spesial tilpasset festeanordning for komponentene vi har benyttet, så har vi 3D-printet innkapslingen. Vi har valgt å printe innkapslingen i PLA plast da dette er lettest og rimeligst å jobbe med. PLA vil ikke møte kravene til robusthet og muligens heller ikke kravet til IP-grad, men er her kun benyttet for å lage en prototype. For en mer robust innkapsling kan man printe i andre plast typer eller eventuelt ved hjelp av 3D modellen formstøpe i plast for masseproduksjon.

For å lage 3D-modell benyttet vi programmet Fusion360 fra produsenten Autodesk. Dette er et kraftig 3D CAD/CAM Software som tilbyr gratis lisenser til studenter, oppstartsbedrifter og hobbybrukere [10]. 3D-modell av innkapslingen ble designet basert på skjematiske 2D tegninger med mål fra produsent av RPi og RockBLOCK enhetene samt en del måling med skyvelære.

Figur 8 Innkapsling utside m/logo.

Figur 9 Innkapsling innside.

4.3.5 Av/På Knapp

For å trygt kunne slå RPi av og på har vi laget en av/på -knapp. RPi blir levert uten dette antageligvis for å holde prisen nede. Årsaken til at vi trenger en trygg måte å stenge ned systemet på er fordi at dersom vi ikke avslutter og stenger ned operativsystemet på en korrekt måte kan vi risikere å få korrupt data på minnekortet eller i verste fall risikere at minnekortet blir ødelagt.

Man kunne selvsagt også valgt en software tilnærming til problemet med nedstenging der man kunne sendt en kommando fra HMI som ville fått RPi til å stenge ned på en trygg måte, men vi har heller valgt en fysisk knapp tilkoblet direkte til RPi slik at vi fremdeles har mulighet for trygg nedstenging selv om vi skulle miste kommunikasjon mellom RPi og HMI.

Måten denne knappen fungerer på er at det på RPi ved oppstart starter ett script som lytter etter knappetrykk. Når skriptet registrerer knappetrykket vil det kalle på en kommando som stenger ned systemet. Ferdig skript med kommandoer og fremgangsmåte har vi hentet fra en nettartikkel [11].

For oppstart av systemet benytter vi oss av en funksjon som er hardkodet i RPi der systemet vil starte opp dersom pin 5 og 6 kortsluttes på RPi. Vi har derfor valgt å koble knappen til disse pinsene slik at vi da kun trenger kode for nedstenging av operativsystemet.

Figur 10 - Strømbryter 1-polet med LED.

Knappen vi har valgt å bruke er en enkel trykknapp [12]. Denne knappen har innebygd LED slik at man ser om systemet er aktiv eller nedstengt. For koblingsskjema og fremgangsmåte henvises det til appendiks C.1 . Kildekode finnes i appendiks E.5 og appendiks E.6

4.3.6 Strømforsyning

For å hente strøm til utstyret vår har vi tilgjengelig strømforsyning via PAT'en. Denne leverer 24V, mens vi trenger 5V for å forsyne RPi. For å transformere spenningen ned til 5V har vi benyttet oss av TSR 1-2450 som er en DC/DC Step-Down Converter fra produsenten TRACOPOWER [13]. Denne fungerer slik at den tar innputt mellom 6.5V og 36V som den transformerer ned og gir i output på 5V. Får å benytte denne må vi lodde den på printkort sammen med kabling/tilkobling til RPi og HMI.

Figur 11 - TSR 1-2450 Step-Down Converter

For å spare plass i innkapslingen var tanken at vi skulle koble strømforsyningen Pin2 og GND på RPi. Etter å ha lest litt i dokumentasjonen og forum for RPi har vi funnet ut av at det ikke er anbefalt å gi strøm via GPIO pins da det ikke finnes noe form for overspenningsvern eller sikringer på disse tilkoblingene [14]. Fra databladet til TSR 1-2450 ser vi at strømforsyningen har innebygd filter og kortslutnings beskyttelse [13] så det ser ut til at det fremdeles kan være mulig å bruke GPIO pins til strømforsyning, men dersom vi velger å bruke disse bør vi først måle og være sikker på at strømforsyningen vår leverer stabilt over tid.

RockBLOCK modulen vil i dette oppsettet ikke få egen strømforsyning, men vil istedenfor få strøm via USB port på RPi. Minimum strøm RockBLOCK modulen trenger for å fungere er 100mA [15]. Vi ønsker å levere så mye effekt som mulig da det forventes at det vil ta lang tid å sende meldinger ettersom kondensatorene i RockBLOCK modulen trenger lengre tid for å lades etter hver melding. RockBLOCK 9603 kan maks trekke 450mA noe som er godt innenfor grensen til hva RPi kan levere.

Ettersom det fra strømforsyningen kun leveres 1A og anbefalt strømforsyning til RPi 3B+ er 2.5A vil det være en mulighet for at vi må strupe ned på hvor mye RPi leverer til RockBLOCK over USB, men dette vil være noe som må kartlegges og testes når strømforsyningen via PAT er komplett.

5 Testing av systemet

5.1 Generelt om tester i prosjektet

I prosjektet ble programvaren for de forskjellige delene i systemet utviklet hver for seg. Når disse delene ble satt sammen ble det nødvendig å kjøre tester. Dette ble gjort i tre forskjellige steg, som følger i kapitlene under.

5.2 Kommunikasjon mellom PC og RockBLOCK

I denne testen ønsket vi å lære om seriell kommunikasjon med RockBLOCK og hvordan man sender data til satellitt. For å gjennomføre testen brukte vi PuTTY [16], som er en emulator for seriell kommunikasjon. Etter at seriell kommunikasjonen var etablert så vi på hvilke AT-kommandoer vi skulle bruke og hvordan vi skulle sende de via seriell kabel (USB/UART kabel).

For å teste om data ble sendt til satellitt sendte vi «BlockROCK» som er en enkel tekst. Vi fikk respons fra RockBLOCK enheten om at sendingen var vellykket. Dette fikk vi bekreftet ved å logge inn på RockBLOCK (https://rockblock.rock7.com/Operations) server portalen. På denne siden loggføres sendinger og gir oss innsyn i data som blir mottatt på serveren. Dette fikk vi bekreftet ved å logge inn på RockBLOCK portalen [17]. På denne siden loggføres sendinger og gir oss innsyn i data som blir mottatt på serveren.

Date Time (UTC)	Device	Direction	Payload	Length (Bytes)	Credits Used	Status	Actions
27/May/2019 13:58:20	RockBLOCK 13645	MO	BlockRock	9	1	!	View
27/May/2019 13:56:19	RockBLOCK 13645	MO	BlockRock	9	1	!	View

Figur 12 - Utklipp fra RockBLOCK server portal som viser når data ble mottatt (Date Time), fra hvilken enhet (Device), om man sender eller mottar (Direction alltid «Message Out» i vår løsning), dataen (Payload), lengden på sendingen (Length), kreditter brukt (Credits Used), status på sendingen (Status) og mer detaljer rundt sendingen (Action).

5.3 Overføring av data fra RPi til RockBLOCK server

I denne testen ønsket vi å se på overføring av enkle data fra RPi til RockBLOCK server. For å teste dette brukte vi et enkelt Python script sammen med RockBLOCK biblioteket fra Makersnake [8].

Programmet sendte en kort tekst med data til RockBLOCK som videresendte dette til server via satellitt. Etter flere forsøk ble denne testen mislykket, og det gikk derfor en del tid på feilsøking. Problemet vi fikk er beskrevet i kapittel 4.2.3.

Etter at problemet ble rettet opp hadde vi utfordringer med at sendingene var uregelmessige fordi vi satt i et vindu i fjerde etasje med kun klar sikt sørover. Vi ble rådet til å flytte testene våre til et sted med bedre sikt fra sør til nord, og vi valgte derfor å ta med utstyret ut. Utstyret hadde nå klar sikt uten bygg, glassvinduer eller andre obstruksjoner som kunne forstyrre signalet. Etter flere forsøk utendørs, fikk vi etterhvert til å sende noe fra Raspberry til RockBLOCK server via satellitt.

5.4 Overføring av data fra HMI til RockBLOCK server.

Vi ønsket å teste hele løsningen for å se om systemet virket slik det skal. For å gjøre dette brukte vi PAT sammen med RPi og RockBLOCK.

For å gjennomføre dette ble det generert test data på HMI som ble lagt inn i databasen. Dette ble så forsøkt sendt fra HMI til RockBLOCK server. Disse forsøkene gikk ikke igjennom fordi vi hadde feil i koden på RPi som kommuniserte med RockBLOCK. Vi har ikke klart å finne noe løsning på dette, som vil si at det ikke har blitt kjørt en komplett test av hele systemet med database. Det vi har fått gjort er å sende data fra HMI til RockBLOCK server med enkle tekst meldinger.

6 Diskusjon

Ettersom oppgaven var en fortsettelse på arbeid fra tidligere bacheloroppgave så var forarbeidet i stor grad basert på å gjennomgå og sette seg inn i arbeidet som var gjort tidligere. For arbeidet med hoveddelen av oppgaven har vi hatt en bra flyt på arbeidet og progresjonen fulgte fremdriftsplanen ganske så bra i begynnelsen. Vi møtte på en del utfordringer ved bruk av ferdig bibliotek i Python for kommunikasjon mellom RPi og RockBLOCK som førte til at vi brukte en del unødvendig tid på feilsøking og løsning som ikke fungerte som planlagt.

Den største utfordringen i prosjektet som førte til at fremdriftsplan ikke har blitt holdt er utforutsette problemer for programmering av HMI ute hos 4Subsea. For å skrive kode til HMI har vi vært avhengig av å få tilgang til PC med lisens til iX Developer. Vi hadde planlagt å sitte hos 4Subsea i påskeuken for å jobbe med HMI, men møtte der på en del PC problemer slik at vi mistet de 3 dagene vi hadde planlagt å bruke ute hos dem. Når vi omsider fikk en fungerende PC med iX Developer så oppdaget vi at koden vi hadde skrevet på forhånd for HMI ikke kunne brukes da koden vi hadde skrevet benyttes seg av .NET 4.0 rammeverk, mens HMI kun hadde støtte for .NET CF. Når problemene rundt .NET rammeverk var utredet og håndtert møtte vi på en del utfordringer vedrørende «sjekksum» for kontroll av meldinger mellom Raspberry og RockBLOCK.

Disse uforutsette hendelsene har sammen ført til at vi ikke klarer å realisere en komplett ferdig løsning for dataoverføring via satellitt.

7 Konklusjon

1. **Målet med oppgaven:**
   Oppdragsgiver ønsket en løsning for overføring av måledata fra deres «Portable Annulus Tester» (PAT) til 4Subseas «Microsoft Azure» server. Det var ønskelig at enheten skulle kunne kommunisere mellom PAT og Azure server på flere mulige måter, alt ettersom hvilken teknologi som skulle være tilgjengelig på plasseringen til PAT.

   Fra kravspesifikasjonen har vi at:
   - Enhetens skal kunne overføre mindre mengder data via satellitt.
   - Enhet skal kunne overføre større mengder data via 4G.
   - Enhet skal kunne overføre større mengder data via Wi-Fi.
   - Enhet skal kunne overføre større mengder data via Ethernet.
   - Enhet skal pakkes i en robust innkapsling som skal ha IP-grad lik eller høyere enn IP54.

2. **Hva har vi fått til:**
   - TCP/IP kommunikasjon mellom HMI og Raspberry.
   - Seriell kommunikasjon mellom Raspberry og RockBLOCK.
   - Sende melding fra HMI til RockBLOCK server.
   - Innkapsling: Kan ved små justeringer i 3D-modell samt ved å printes eller støpes i et annet materiale møte krav til robusthet og IP.

3. **Anbefaling for videre arbeid:**
   - Bedre feilhåndtering i kode for kommunikasjon mellom HMI-RPi-RockBLOCK
   - Kode for å overføre data fra RockBLOCK server til 4Subsea sin Azure server
   - Løsning for 4G
   - Løsning for Wi-Fi
   - Løsning for Ethernet

I oppgaven har vi kommet et godt stykke på vei i å realisere en løsning for å sende måledata fra PAT over satellitt. For at satellittkommunikasjon skal fungere optimalt gjenstår det å skrive bedre kode for feilhåndtering samt finne årsak til -og utbedre problemer med «sjekksum» i kommunikasjonen mellom Raspberry og RockBLOCK enhet.

Oppgaven har ikke sett på løsninger for kommunikasjon over Wi-Fi og 4G. Slik vi har tenkt oppgaven videre, trenger man ikke å gjøre store endringer i koden for at HMI skal kunne ta i bruk Wi-Fi eller 4G. I programvaren har vi prøvd å tilrettelegge for implementering av en Wi-Fi løsning i den eksisterende koden. For 4G løsning må man bestille komponenter til RPi for å muliggjøre kommunikasjon via 4G.

Referanser

[1] Brønnøysundregistrene, «Kunngjøringer 4Subsea AS,» 11 Juli 2018. [Internett]. Available: https://w2.brreg.no/kunngjoring/hent_nr.jsp?orgnr=991711392. [Funnet 31 Januar 2019].

[2] 4Subsea, «4Subsea and Ashtead Technology Enter Strategic Partnership,» 01 Januar 2019. [Internett]. Available: https://www.4subsea.com/2019/01/4subsea-and-ashtead-technology-enter-strategic-partnership/. [Funnet 31 Januar 2019].

[3] 4Subsea, «Who We Are and What We Believe In,» [Internett]. Available: https://www.4subsea.com/subsea-systems/. [Funnet 31 Januar 2019].

[4] Proff, «Regnskap 4Subsea AS,» [Internett]. Available: https://www.proff.no/regnskap/4subsea-as/asker/tekniske-konsulenter/IGEFTEO01OU-1/. [Funnet 31 01 2019].

[5] Rock7, «Adaptive Retry,» 1 Januar 2014. [Internett]. Available: https://docs.rockblock.rock7.com/docs/adaptive-retry. [Funnet 22 Mai 2019].

[6] Future Technology Devices International Ltd, «TTL to USB Serial Converter Range of Cables,» 23 Mai 2016. [Internett]. Available: https://www.ftdichip.com/Support/Documents/DataSheets/Cables/DS_TTL-232R_CABLES.pdf. [Funnet 26 Mai 2019].

[7] C. Liechti, «pySerial,» 23 Juli 2017. [Internett]. Available: https://pyserial.readthedocs.io/en/latest/. [Funnet 3 April 2019].

[8] Makersnake, «Github,» 16 November 2016. [Internett]. Available: https://github.com/MakerSnake/pyRockBlock. [Funnet 16 Februar 2019].

[9] Iridium, «ISU AT Command Reference,» 22 Juni 2012. [Internett]. Available: http://www.rock7mobile.com/downloads/IRDM_ISU_ATCommandReferenceMAN0009_Rev2.0_ATCOMM_Oct2012.pdf. [Funnet 15 Mars 2019].

[10] AutoDesk, «Fusion 360 for hobbyists and makers,» autodesk.com, [Internett]. Available: https://www.autodesk.com/campaigns/fusion-360-for-hobbyists. [Funnet 01 Mai 2019].

[11] Tyler, «How to add a powe button to your Raspberry Pi,» [Internett]. Available: https://howchoo.com/g/mwnlytk3zmm/how-to-add-a-power-button-to-your-raspberry-pi. [Funnet 02 Mai 2019].

[12] «Strømbryter 1-polet fra/til Hvit belysning,» [Internett]. Available: https://www.kjell.com/no/produkter/elektro-og-verktoy/elektronikk/electromechanics/strombrytere/trykkstrombrytere/strombryter-1-polet-fra-til-hvit-belysning-p36142. [Funnet 02 Mai 2019].

[13] TRACO POWER, «www.tracopower.com,» [Internett]. Available: https://www.tracopower.com/products/tsr1.pdf. [Funnet 03 Mai 2019].

[14] The MagPi Magazine, «raspberrypi.org,» [Internett]. Available: https://www.raspberrypi.org/magpi/power-supply/. [Funnet 16 mai 2019].

[15] Rock Seven, «RockBLOCK 9603 - Developer guide,» 1 Juni 2017. [Internett]. Available: https://cdn.sparkfun.com/assets/6/d/4/c/a/RockBLOCK-9603-Developers-Guide_1.pdf. [Funnet 25 Mai 2019].

[16] S. Tatham, «PuTTY: a free SSH and Telnet client,» 26 Mars 2019. [Internett]. Available: https://www.chiark.greenend.org.uk/~sgtatham/putty/. [Funnet April 4 2019].

[17] Rock Seven, «Operations,» 25 Mai 2019. [Internett]. Available: https://rockblock.rock7.com/Operations. [Funnet 25 Mai 2019].

[18] Raspberry Pi Foundation, «Raspberry Pi 3 Model B+,» 25 Mai 2019. [Internett]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/ . [Funnet 25 Mai 2019].

[19] Rock Seven, «RockBLOCK 9603,» [Internett]. Available: http://www.rock7mobile.com/products-rockblock-9603. [Funnet 15 Mars 2019].

[20] Wikipedia, «IP-systemet,» 23 Desember 2017. [Internett]. Available: https://no.wikipedia.org/wiki/IP-systemet. [Funnet 25 Mai 2019].

[21] HiB, UiB, NHH, UiO og Nasjonalbiblioteket, «Søk og Skriv,» 12 12 2014. [Internett]. Available: http://sokogskriv.no/. [Funnet 12 12 2014].

[22] Iridium, «Iridium,» 31 Mai 2019. [Internett]. Available: https://www.iridium.com. [Funnet 31 Mai 2019].

Appendiks A Forkortelser og ordforklaringer

.NET CF	.NET Compact Framework. Framework for mobile enheter.
ATEX	ATmospheres EXplosibles. Direktiv for utstyr som benyttes i eksplosjonsfarlige områder
CAD	Computer Aided Design (Norsk: Dataassistert Konstruksjon)
CAM	Computer Aided Manufacturing (Norsk: Dataassistert Produksjon)
GPIO	General-Purpose input/output. I/O tilkoblinger på Raspberry Pi
GUI	Graphical User Interface. (Norsk: Grafisk brukergrensesnitt)
HMI	Human-Machine Interface
IDE	Integrated Development Environment. Norsk: Integrert utviklingsmiljø.
IP	Internet Protocol
IP-grad	International Protection, sier noe om kapslingsgrad til utstyret.
Iridium	Er en leverandør av satellitt tjenester.
Klient	En klient er en datamaskin i et nettverk som utnytter resursene på en server
OOP	Objektorientert programmering er et paradigme for programmering av datamaskiner.
PAT	Portable Annulus Tester
PLA	Poly Lactic Acid. Bioplast som brukes i blant annet 3D-Print
RPi	Raspberry Pi 3 B+
Script language	Et programmeringsspråk som blant annet brukes i forbindelse med utveksling av data mellom WWW og databaser. Med dette kan man skrive scripts (makroer) for å automatisere visse funksjoner
Server	Egentlig et program på en datamaskin koblet til Internet eller liknende. Programmet gjør at personer utenfra kan nå deler av det som finnes laget i datamaskinen, f.eks. websider eller nedlastbare filer
Sjekksum	En sjekksum er en kort kode som brukes til å sjekke integriteten av data, eller den matematiske funksjonen av dataene, algoritmen, som genererer koden, ofte kalt hash-funksjon.
SQL	Structured Query Language. Et programmeringsspråk for behandling av databaser.
SQLite	SQLite er et lite C-bibliotek som implementerer en frittstående SQL-databasemotor.
USB	Universal Serial Bus. Seriell databuss for å koble enheter til en datamaskin
TCP/IP	Transmission Control Protocol/Internet Protocol. En samling kommunikasjonsprotokoller benyttet på Internet, men også i mange lokale og andre datanettverk.
Tråder	Tråder i programmeringssammenheng er betegnelsen på flere deler av det samme programmet som kjører samtidig. Flere tråder kan i moderne datasystemer kjøres parallelt.
Wi-Fi	Trådløst lokalt datanett. Trådløst LAN (LAN = Local Area Netwrok) (engelsk Wireless LAN, WLAN)
WPF	Windows Presentation Foundation er et grafisk system fra Microsoft som generer brukergrensesnitt for Windows applikasjoner.

Appendiks B Prosjektledelse og styring

B.1 Prosjektorganisasjon

I arbeidet med prosjektet har det hele tiden vært klart hva som skal gjøres, og det har vært åpent i hvilken grad det enkelte medlem skulle bidra til arbeidet. Øystein har inntatt rollen som prosjektleder, med ansvar for kontakt med veileder. Stein har hatt ansvar for kontakt med oppdragsgiver. Prosjektet er delt opp i to delen software og hardware. Øystein har hatt hovedansvaret for software delen med hjelp fra Eivind, mens Stein har hatt ansvar for hardware delen.

B.2 Prosjektform

Vi har brukt en egendefinert form hvor hvert enkelt gruppemedlem har stått ganske fritt til å styre sin egen tid slik de ønsker. Vi har valgt denne formen fordi vi har liten/ingen kunnskap om andre prosjektmetoder og denne formen har fungert i tidligere prosjekter gruppen har hatt sammen. Arbeidet med forstudiet inngår i liten graden i resten av oppgaven fordi den baserer seg på bacheloroppgaven som ble levert høst 2018, hvor hovedfokuset var å finne ut hvordan type utstyr som kunne bli brukt i prosjektet og ikke en faktisk realisering av dette.

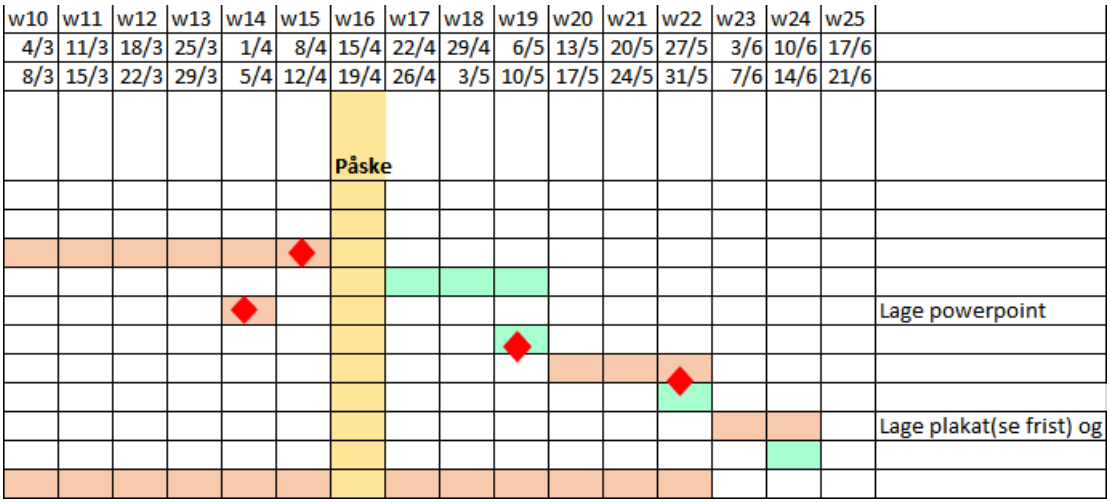
B.3 Fremdriftsplan

I fremdriftsplanene (Figur 13 og Figur 14) som ble laget til forstudiet var fokuset vårt å lage en overordnet plan som vi kunne følge frem til prosjektets slutt. Denne ble for lite detaljert og var vanskelig å jobbe ut ifra. Så det ble laget en ny versjon med samme plan, men de forskjellige delene ble splittet opp for å gjøre dette klarere, se Figur 15.

Vi holdt oss til fremdriftsplanen frem til uke 19. Etter dette ble det vanskelig å fortsette etter planen fordi kontaktperson hos oppdragsgiver sluttet, fridager stykket opp arbeidet og vi møtte utfordringer med implementasjonen på HMI.

					Uke	w2	w3	w4	w5	w6	w7	w8	w9
					Man	7/1	14/1	21/1	28/1	4/2	11/2	18/2	25/2
					Fre	11/1	18/1	25/1	1/2	8/2	15/2	22/2	1/3
#	Aktivitet	Start Dato	Slutt Dato	Fram drift	Ansvarlig								SU?
1	Forstudie arbeid	1/7	31/1	75 %	oss								
2	Forstudie innlevering/	31/1		75 %	oss								
3	Utvikle satellitt løsning			1 %	oss								
4	Utvikle 4G/wifi løsning			0 %	oss								
5	Midtveis presentasjon	1/4	5/4	0 %	oss								
6	Bachelor oppgave seminar/	10/5		0 %	oss								
7	Ferdigstille løsning/se på			0 %	oss								
8	Bachelor oppgave innlevering	31/5		0 %	oss								
9	Bachelor oppgave presentasjon	5/6	12/6	0 %	oss								
10	EXPO / Avslutningsfest	13/6		0 %	oss								
11	Møte med veileder			5 %	oss								

Figur 13 - Gantt forstuide del 1.



Figur 14 - Gantt forstuide del 2.

TASK NAME	START DATE	END DATE	DURATION (WORK DAYS)	TEAM MEMBER	PERCENT COMPLETE
Forstudie					
Forstudie arbeid	7.1.	31.1.	19	Alle	100 %
Første møte 4Subsea	11.1.	11.1.	1	Alle	100 %
Forstudie innlevering	31.1.	31.1.	1	Alle	100 %
Forstudie presentert for 4Subsea	15.2.	15.2.	1	Alle	100 %
Software utvikling					
Sette opp Raspberry Pi			0	Stein	100 %
Programmere database			0	Øystein	100 %
Programmere HMI client			0	Øystein	100 %
Programmere Raspberry Pi Server				Øystein	100 %
Programmere kode mot RockBlock				Øystein	100 %
Implementere kode på HMI			0	Alle	50 %
Hardware					
Hente utstyr hos 4Subsea			0	Stein/Eivind	100 %
Bestille antenne til RockBlock			0	Stein/Eivind	100 %
Bestille omformer til Raspberry Pi			0	Stein/Eivind	100 %
Se om vi må ha på/av knappp til			0	Stein/Eivind	100 %
Låne "koffert" til Expo			0	Stein/Eivind	0 %
Testing					
Teste HMI			0	Eivind	75 %
Test Raspberry Pi			0	Eivind	100 %
Teste satellitt overføring			0	Stein	100 %
Testet komplett løsning			0	Alle	75 %
Bachelor gruppe oppgaver					
Skrive på bachelor oppgave	1.1.	1.6.	109	Alle	70 %
Møte med veileder	1.1.	1.6.	109	Alle	75 %
Møte med bachelor gruppe	1.1.	12.6.	117	Alle	75 %
Møte med 4Subsea	1.1.	12.6.	117	Alle	100 %
Midtveis presentasjon	1.4.	5.4.	5	Eivind	100 %
Bachelor oppgave seminar	10.5.	10.5.	1	Eivind	100 %
Bachelor oppgave innlevering	1.6.	1.6.	0	Eivind	0 %
EXPO / Avslutningsfest	13.6.	13.6.	1	Stein/Eivind	0 %

Figur 15 - Gantt ny versjon som viser en mer detaljer oversikt over de forskjellige delene i prosjektet.

B.4 Risikoliste

Risiko	Sannsynlighet (1-10)	Alvorlighetsgrad (1-10)	Konsekvens	Tiltak
Sykdom	5	5	Arbeidstid går tapt	Den/de som er friste må ta ansvar for oppgaven til den/de syke
Personlige konflikter	2	7	Dårlig arbeidsmoral og fare for tapt arbeidstid	Ha klar rollefordeling og konflikthåndtering avtalt før prosjekt start
Tidsfrister	6	7	Rekker ikke å ferdigstille oppgaven	Sette realistiske mål
Kompabilitet	5	9	System delene fungerer ikke sammen	Planlegge hva man skal bruke i prosjektet
Oppdragsgiver/veileder utilgjengelig	5	2	Arbeidstid kan gå tapt	Gjøre avtaler i god tid
Kunnskaps mangel	9	2	At vi ikke får gjennomført prosjektet på ønsket måte	

Tabell 1 – Risikoliste

Appendiks C Brukerdokumentasjon

C.1 Av/På Knapp

Installasjon

Når man er i mappen PowerButton.py er lagret i skriv følgende:

```
1. sudo mv PowerButton.py /usr/local/bin/
2. sudo chmod +x /usr/local/bin/PowerButton.py
```

Når man er i mappen PowerButton.sh er lagret i skriv:

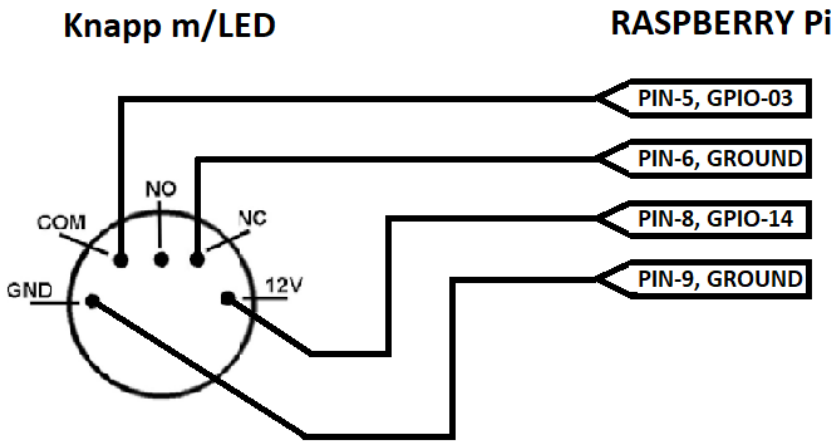
```
1. sudo mv PowerButton.sh /etc/init.d/
2. sudo chmod +x /etc/init.d/PowerButton.sh
```

Nå kan vi registrere scriptet til å kjøre ved boot med følgende kommando:

```
1. sudo update-rc.d PowerButton.sh defaults
```

For at LED skal virke må vi enable GPIO serial port. Dette gjøres i config filen.
Åpne: /boot/config.txt
Skriv i fil og lagre den:

```
1. enable_uart=1
```

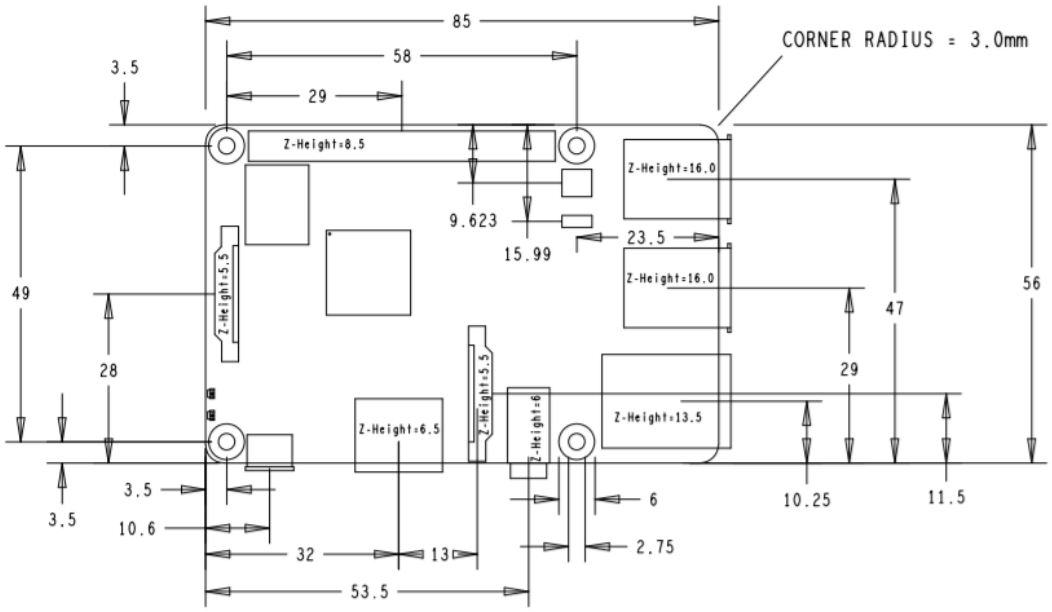


Figur 16 - Koblingsskjema knapp m/LED

Appendiks D Øvrige diagrammer

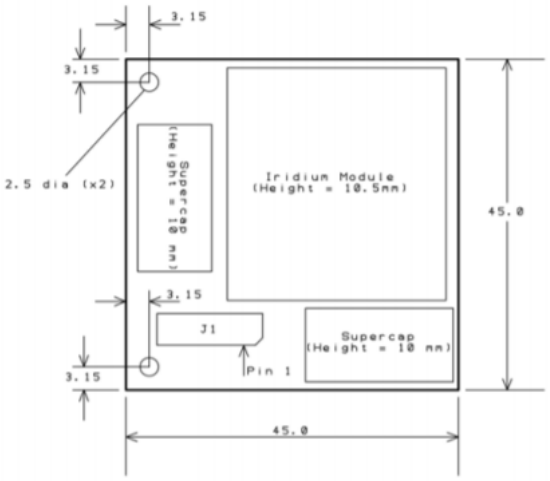
D.1 Tekniske tegninger brukt i design av 3D-modell for innkapsling

Raspberry Pi 3 Model B+



Figur 17 - Fysiske spesifikasjoner Raspberry Pi [18]

RockBLOCK 9603



Figur 18 - Fysiske spesifikasjoner RockBLOCK 9603 [19]

Appendiks E **Kildekode**

E.1 HMI-kode (C#)

Kode som er lagt inn på iX Developer som et script.

```
1.  //-------------------------------------------------------------
2.  // Press F1 to get help about using script.
3.  // To access an object that is not located in the current class, start the call with
     Globals.
4.  // When using events and timers be cautious not to generate memoryleaks,
5.  // please see the help for more information.
6.  //-------------------------------------------------------------
7.  namespace Neo.ApplicationFramework.Generated
8.  {
9.      using System.Windows.Forms;
10.     using System;
11.     using System.Drawing;
12.     using Neo.ApplicationFramework.Tools;
13.     using Neo.ApplicationFramework.Common.Graphics.Logic;
14.     using Neo.ApplicationFramework.Controls;
15.     using Neo.ApplicationFramework.Interfaces;
16.
17.     using System.Threading;
18.     using System.Net;
19.     using System.Net.Sockets;
20.     using System.IO;
21.     using System.Text;
22.     using System.Collections.Generic;
23.     using System.Data.SQLite;
24.
25.     public partial class Screen1
26.     {
27.         string hmi_name = "HMI_2019";
28.         void Button_Click(System.Object sender, System.EventArgs e)
29.         {
30.             Satellite satellite = new Satellite(ListBox, hmi_name);
31.             TcpCommunication tcpCommunication = new TcpCommunication(ListBox, satell
    ite);
32.             Thread thread = new Thread(tcpCommunication.TcpHandler);
33.             thread.Start();
34.         }
35.     }
36.
37.     // Handling the satellite communication
38.     class Satellite
39.     {
40.         private readonly ListBox _listBox;
41.         private readonly MoStatus _moStatus;
42.         private readonly Database _database;
43.
44.         private readonly string _hmi_name;
45.
46.         public Satellite(ListBox listBox, string hmi_name)
47.         {
48.             _listBox = listBox;
49.             _moStatus = new MoStatus();
50.             _database = new Database(_listBox);
51.             _hmi_name = hmi_name;
52.         }
53.
54.         public void HandleCommunication(byte[] byteBuffer, Socket socket)
55.         {
56.             ReadFromDatabase();
57.             ReceiveRockblockStatus(byteBuffer, socket);
58.             SendDataToRaspberry(socket);
```

```
59.             ReceiveSendStatus(byteBuffer, socket);
60.         }
61.
62.         private void UpdateListView(string item)
63.         {
64.             _listBox.Items.Add(item);
65.         }
66.
67.         private void ReadFromDatabase()
68.         {
69.             try
70.             {
71.                 _database.ReadDataForSatellitte(_hmi_name);
72.                 UpdateListView("Data loaded from the database!");
73.             }
74.             catch (SQLiteException sqLiteExp)
75.             {
76.                 UpdateListView(sqLiteExp.Message);
77.             }
78.             catch (Exception exception)
79.             {
80.                 UpdateListView(exception.Message);
81.             }
82.         }
83.
84.         private void ReceiveSendStatus(byte[] bytes, Socket socket)
85.         {
86.             var response = "";
87.             while (response != "END_OF_SENDING")
88.             {
89.                 var bytesRec = socket.Receive(bytes);
90.                 response = Encoding.UTF8.GetString(bytes, 0, bytesRec);
91.
92.                 if (response.StartsWith("MoStatus"))
93.                 {
94.                     var attemptResponse = response.Substring(response.LastIndexOf(':
    ') + 1, 1);
95.                     UpdateListView("Attempt: " + attemptResponse);
96.                     var moResponse = response.Substring(response.IndexOf(':', 0) + 1
    , 2);
97.                     //UpdateListView("Status: " + _moStatus.FindValue(Convert.ToInt1
    6(moResponse)));
98.                     UpdateListView("Status: ");
99.                 }
100.
101.                if (response != "END_OF_SENDING" & !response.StartsWith("MoStatus"))

102.                {
103.                    UpdateListView(response);
104.                }
105.            }
106.            UpdateListView("Sending is done!");
107.        }
108.
109.        private void SendDataToRaspberry(Socket socket)
110.        {
111.            foreach (var line in _database.DbStringList)
112.            {
113.                byte[] msgBytes = Encoding.UTF8.GetBytes(line);
114.                socket.Send(msgBytes, line.Length, SocketFlags.None);
115.                Thread.Sleep(50); // To make sure it dosent read blank lines(msgByte
    s is empty).
116.            }
117.            var end_of_line = "END";
118.            socket.Send(Encoding.UTF8.GetBytes(end_of_line));
119.            UpdateListView("Data transfered to Raspberry");
```

```
120.         }
121.
122.         private void ReceiveRockblockStatus(byte[] bytes, Socket socket)
123.         {
124.             var rbIsOk = socket.Receive(bytes);
125.             var response = Encoding.UTF8.GetString(bytes, 0, rbIsOk);
126.             UpdateListView("Connecting to the rockblock: " + response);
127.             if (response != "OK")
128.             {
129.                 UpdateListView("Connecting to the rockblock failed");
130.             }
131.             else
132.             {
133.                 UpdateListView("Connecting to the rockblock succeeded");
134.             }
135.         }
136.     }
137.     public class TcpCommunication
138.     {
139.         private IPAddress _ipAddress;
140.         private IPEndPoint _remoteEp;
141.         private const int ByteBuffer = 2048;
142.         private const int PortNumber = 5005;
143.         private readonly ListBox _listBox;
144.         private readonly byte[] _bytes;
145.         private readonly string _clientIp;
146.         private readonly object _sender;
147.
148.         public TcpCommunication(ListBox listBox, object transferObject)
149.         {
150.             _listBox = listBox;
151.             _bytes = new byte[ByteBuffer];
152.             _clientIp = "10.4.86.100";
153.             _sender = transferObject;
154.         }
155.
156.         public void TcpHandler()
157.         {
158.             try
159.             {
160.                 using (Socket socket = ConnectSocket(_clientIp, PortNumber))
161.                 {
162.                     var typeOfSending = Encoding.UTF8.GetBytes(_sender.GetType().Nam
    e);
163.                     socket.Send(typeOfSending, typeOfSending.Length, SocketFlags.Non
    e);
164.
165.                     Satellite satellite = _sender as Satellite;
166.                     if (satellite != null)
167.                     {
168.                         satellite.HandleCommunication(_bytes, socket);
169.                     }
170.                 }
171.             }
172.             catch (ArgumentNullException argumentNullExp)
173.             {
174.                 UpdateListView(argumentNullExp.Message);
175.             }
176.             catch (SocketException socketExp)
177.             {
178.                 UpdateListView(socketExp.Message);
179.             }
180.             catch (Exception exp)
181.             {
182.                 UpdateListView(exp.Message);
183.             }
```

```
184.         }
185.
186.         private Socket ConnectSocket(string server, int port)
187.         {
188.             _ipAddress = IPAddress.Parse(server);
189.             Socket tempSocket = new Socket(_ipAddress.AddressFamily, SocketType.Stre
    am, ProtocolType.Tcp);
190.             _ipAddress = IPAddress.Parse(server);
191.             _remoteEp = new IPEndPoint(_ipAddress, port);
192.             tempSocket.Connect(_remoteEp);
193.             if (tempSocket.Connected)
194.             {
195.                 var s = tempSocket;
196.                 return s;
197.             }
198.             return null;
199.         }
200.
201.         private void UpdateListView(string item)
202.         {
203.             _listBox.Items.Add(item);
204.         }
205.     }
206.     public class Database
207.     {
208.         public List<string> DbStringList = new List<string>();
209.         private const string Version = "Version=3";
210.         private SQLiteConnection dbConnection;
211.         private ListBox _listBox;
212.
213.         public Database(ListBox listBox)
214.         {
215.             _listBox = listBox;
216.         }
217.
218.         public List<string> ReturnDbStringList()
219.         {
220.             return DbStringList;
221.         }
222.
223.         private void UpdateListView(string item)
224.         {
225.             _listBox.Items.Add(item);
226.         }
227.
228.         public void ReadDataForSatellitte(string _hmi_name)
229.         {
230.             //DbStringList.Add(_hmi_name + "-");
231.             using (dbConnection = new SQLiteConnection(GetConnectionString()))
232.             {
233.                 dbConnection.Open();
234.                 var command = dbConnection.CreateCommand();
235.                 command.CommandText = "SELECT * FROM FVM_results";
236.                 var reader = command.ExecuteReader();
237.                 var toList = "";
238.                 while (reader.Read())
239.                 {
240.                     List<string> itteratorList = Itterator();
241.                     foreach (string line in itteratorList)
242.                     {
243.                             toList += reader[line] + " ";
244.                     }
245.                     string temp = toList.Trim();
246.                     DbStringList.Add(temp + "END_OF_LINE");
247.                     toList = "";
248.                     temp = "";
```

```
249.                 }
250.             }
251.         }
252.
253.         private List<string> Itterator ()
254.         {
255.             List<string> tempList = new List<string>();
256.             tempList.Add("Id");
257.             tempList.Add("Time");
258.             tempList.Add("Test_Name");
259.             tempList.Add("Free_Volume_litres");
260.             tempList.Add("Vent_Port_Flow_Capacity_l_per_min");
261.             tempList.Add("Test_Duration_minutes");
262.             tempList.Add("Test_Pressure_barg");
263.             tempList.Add("FVMongoing");
264.             tempList.Add("FVMqualityIndex");
265.             tempList.Add("Final_pressure_barg");
266.             tempList.Add("UnitID");
267.             return tempList;
268.         }
269.
270.         private static string GetConnectionString()
271.         {
272.             string excecutingPath = Path.GetDirectoryName(System.Reflection.Assembly
    .GetExecutingAssembly().ManifestModule.FullyQualifiedName);
273.             return string.Format("data source={0}", Path.Combine(excecutingPath, "Da
    tabase.db"));
274.         }
275.     }
276.     public class MoStatus
277.     {
278.         private Dictionary<int, string> Mostatus = new Dictionary<int, string>();
279.         private readonly List<string> _tempList = new List<string>();
280.         private FileStream _file;
281.
282.         public MoStatus()
283.         {
284.             GetValuesFromFile();
285.             AddValues();
286.         }
287.
288.         public Dictionary<int, string> ReturnMoStatus()
289.         {
290.             return Mostatus;
291.         }
292.
293.         public void GetValuesFromFile()
294.         {
295.             try
296.             {   //
297.                 _file = File.Open("MoValues.txt", FileMode.Open);
298.                 using (var streamReader = new StreamReader(_file, Encoding.UTF8))
299.                 {
300.                     string line;
301.                     while ((line = streamReader.ReadLine()) != null)
302.                     {
303.                         _tempList.Add(line);
304.                     }
305.                 }
306.             }
307.             catch (FileNotFoundException e)
308.             {
309.                 Console.WriteLine(e);
310.             }
311.             catch (Exception e)
312.             {
```

```
313.                 Console.WriteLine(e);
314.             }
315.         }
316.         public void AddValues()
317.         {
318.             foreach (var line in _tempList)
319.             {
320.                 var arraySplit = line.Split('-');
321.                 Mostatus.Add(Convert.ToInt16(arraySplit[0]), arraySplit[1]);
322.             }
323.         }
324.
325.         public string FindValue(int keyValue)
326.         {
327.             return Mostatus[keyValue];
328.         }
329.     }
330. }
```

E.2 Raspberry-kode server (Python)

Koden som håndterer socket kommunikasjonen.

```
1.  """Module for the TCP communication with HMI"""
2.  import socket
3.  from satellite import Satellite
4.  from wifi import Wifi
5.  from long_term_evolution import LongTermEvolution
6.
7.
8.  class Server:
9.    """Class for starting the server"""
10.   def __init__(self, tcp_ip, tcp_port):
11.     self.server_socket = None
12.     self.tcp_ip = tcp_ip
13.     self.tcp_port = tcp_port
14.     self.buffer_size = 256
15.
16.   def open_socket(self):
17.     """Opens a socket object."""
18.     try:
19.       print("Socket started")
20.       # with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as self.server_socket:
21.       self.server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
22.       self.server_socket.bind((self.tcp_ip, self.tcp_port))
23.       self.server_socket.listen(1)
24.     except OSError as msg:
25.       print("OS error: {0}".format(msg))
26.
27.   def accept_client(self):
28.     """Wait for a client to connect and sends messages both ways."""
29.     while True:
30.       conn, address = self.server_socket.accept()
31.       with conn:
32.         data_recv = conn.recv(self.buffer_size)
33.         transfer_object = data_recv.decode('utf-8')
34.         if transfer_object == "Satellite":
35.           print("Received: " + str(transfer_object))
36.           sat = Satellite(conn)
37.           sat.handle_satellite_communication()
38.         if transfer_object == "Wifi":
39.           pass
40.         if transfer_object == "LongTermEvolution":
```

```
            pass

if __name__ == "__main__":
    S = Server(tcp_ip="10.4.86.100", tcp_port=5005)
    S.open_socket()
    S.accept_client()
```

E.3 Raspberry-kode satellitt (Python)

Koden som håndterer satellittkommunikasjonen.

```
"""Handles the communication with rockblock and HMI"""
import time
import serial
from contextlib import contextmanager
from rock_block_message import RockBlockMessage


class Satellite():
    """Satellite class"""
    def __init__(self, socket):
        self.socket_conn = socket
        self.rockblock_conn = None
        self.buffer_size = 256
        self.string_array = []

    def handle_satellite_communication(self):
        """Handles the communication with rockblock and HMI"""
        try:
            with self.start_rb_serial() as self.rockblock_conn:
                print("Serial started. Rockblock_conn: " + str(self.rockblock_conn))

                if self.check_connection():
                    self.receive_data_from_hmi()
                    self.send_data_to_rockblock()
                    self.rockblock_response(b"END_OF_SENDING")
                    print("Sending is done!")
                else:
                    pass
        except serial.SerialException:
            pass

    def receive_data_from_hmi(self):
        """Receives data from the HMI."""
        end_of_line = "END"
        while True:
            data = self.socket_conn.recv(self.buffer_size)
            if data.decode('utf-8') == end_of_line:
                break
            self.string_array.append(data.decode('utf-8'))

    def send_data_to_rockblock(self):
        """Clears the MO buffer and sends data from HMI to rockblock."""
        self.rockblock_conn.clear_buffer()
        self.rockblock_conn.message_buffer(self.string_array)

    def rockblock_response(self, response):
        """Sends status updates to the HMI."""
        try:
            self.socket_conn.sendall(response)
            time.sleep(5)
        except OSError as msg:
            print(msg)


    def check_connection(self):
```

```
55.         """Checks if the rockblock is reachable."""
56.         if self.rockblock_conn.ping():
57.             print("Ping OK")
58.             self.socket_conn.sendall(b"OK")
59.             return True
60.         print("Ping failed")
61.         self.socket_conn.sendall(b"ERROR")
62.         return False
63.
64.     def check_signal_strength(self):
65.         """Checks the signal strength."""
66.         if self.rockblock_conn.check_signal() > 0:
67.             self.socket_conn.sendall(b"OK")
68.             return True
69.         self.socket_conn.sendall(b"ERROR")
70.         return False
71.
72.     @contextmanager
73.     def start_rb_serial(self):
74.         """Starts the serial with rockblock."""
75.         rockblock_conn = RockBlockMessage(self, self.get_port())
76.         yield rockblock_conn
77.         rockblock_conn.close_serial()
78.
79.     @staticmethod
80.     def get_port():
81.         """Find the port rockblock is connected to and connect to it."""
82.         ports = RockBlockMessage.list_ports()
83.         print(ports)
84.         return ports[0]
```

E.4 Raspberry-kode RockBLOCK script (Python)

Scriptet som håndterer kommunikasjonen med RockBLOCK.

```
1.  """Handling the communication with the rockblock unit"""
2.  import random
3.  import time
4.  import sys
5.  import glob
6.  import serial
7.
8.
9.  class RockBlockMessage:
10.     """A class for handling rockblock communication."""
11.     def __init__(self, server, port_id):
12.         self.server = server
13.         self.port_id = port_id
14.         self.serial_conn = serial.Serial(port_id, 19200, timeout=30)
15.
16.     def ping(self):
17.         """Checks connectivity to the rockblock."""
18.         command = "AT"
19.         self.serial_conn.write(b"AT\r")
20.         if self.serial_conn.readline().strip().decode("utf-8") == command:
21.             if self.serial_conn.readline().strip().decode("utf-8") == "OK":
22.                 return True
23.         return False
24.
25.     def send(self, msg):
26.         """Adds a message to the rockblock buffer."""
27.         ready_to_send = False
28.         command = "AT+SBDWB=" + str(len(msg))
29.         self.serial_conn.write(bytes(command + "\r", 'utf-8'))
```

```
30.         if self.serial_conn.readline().strip().decode("utf-8") == command:
31.             if self.serial_conn.readline().strip().decode("utf-8") == "READY":
32.                 # This is the checksum control and where we have an error.
33.                 # Think the error only happends when b return two hex's.
34.                 checksum = sum(ord(c) for c in msg)
35.                 self.serial_conn.write(bytes(msg, 'utf-8'))
36.                 a = checksum >> 8
37.                 b = checksum & 0xFF
38.                 self.serial_conn.write(bytes(chr(a), 'utf-8'))
39.                 check_last = bytes(chr(b), 'utf-8')
40.                 if len(check_last) > 1:
41.                     check_last = check_last[0:1]
42.                 self.serial_conn.write(check_last)
43.                 temp = self.serial_conn.readline().strip().decode("utf-
    8")   # BLANK
44.                 temp2 = self.serial_conn.readline().strip().decode("utf-8")  # 0
45.                 if temp2 == "0":
46.                     self.server.rockblock_response(b"Message added to buffer")
47.                     ready_to_send = True
48.                 self.serial_conn.readline().strip().decode("utf-8")  # BLANK
49.                 self.serial_conn.readline().strip().decode("utf-8")  # OK
50.                 if ready_to_send:
51.                     self.message_session()
52.             else:
53.                 self.server.rockblock_response(b"Rockblock not ready for another mes
    sage")
54.         else:
55.             self.server.rockblock_response(b"Couldn't send message to rockblock")
56.
57.     def message_buffer(self, msg):
58.         """Takes a message and splits it i lesser pieces before forwarding"""
59.         temp = ""
60.         for line in msg:
61.             if len(temp) + len(line) < 340:
62.                 temp += line
63.             elif len(temp) + len(line) > 340:
64.                 self.send(temp)
65.                 temp = line
66.                 continue
67.         self.send(temp)
68.         #self.send(line)
69.         #if len(temp) + len(line) < 340:
70.         #        temp +=
71.         #
72.         #    print("Message_buffer: " + line)
73.         #    temp = line
74.         #    self.send(temp)
75.         #while len(msg) > 340:
76.         #    firstpart = msg[:340]
77.         #    self.send(firstpart)
78.         #    msg = msg[len(firstpart):len(msg)]
79.         #self.send(msg)
80.
81.     def check_signal(self):
82.         """Sending a request for the signal strength."""
83.         command = "AT+CSQ"
84.         self.serial_conn.write(b"AT+CSQ\r")
85.         if self.serial_conn.readline().strip().decode("utf-8") == command:
86.             response = self.serial_conn.readline().strip().decode("utf-8")
87.             if response.find("+CSQ") >= 0:
88.                 self.serial_conn.readline().strip()  # BLANK
89.                 self.serial_conn.readline().strip()  # OK
90.                 if len(response) == 6:
91.                     return int(response[5])
92.         return False
93.
```

```
94.     def message_session(self):
95.         """Sends the message from buffer to satellite."""
96.         attempt = 1
97.         max_attempt = 7
98.         while attempt < max_attempt:
99.             command = "AT+SBDIX"
100.             self.serial_conn.write(b"AT+SBDIX\r")
101.             if self.serial_conn.readline().strip().decode("utf-8") == command:
102.                 response = self.serial_conn.readline().strip().decode("utf-8")
103.                 print("Message_session - Response one: " + str(response))
104.                 self.serial_conn.readline().strip().decode("utf-8")  # BLANK
105.                 if response.find("+SBDIX") >= 0:
106.                     self.serial_conn.readline()  # BLANK
107.                     self.serial_conn.readline()  # Ok
108.                     response = response.replace("+SBDIX: ", "")
109.                     parts = response.split(",")
110.                     mo_status = int(parts[0])
111.                     # mo_msn = int(parts[1])
112.                     # mt_status = int(parts[2])
113.                     # mt_msn = int(parts[3])
114.                     # mt_length = int(parts[4])
115.                     # mt_queued = int(parts[5])
116.                     if mo_status <= 2:
117.                         self.__mo_status_ok_response(attempt, mo_status)
118.                         break
119.                     self.__mo_status_failed_response(attempt, mo_status)
120.                     attempt = self.__handle_attempts(attempt)
121.
122.     def __mo_status_failed_response(self, attempt, mo_status):
123.         """Add docstring"""
124.         print("MoStatus: " + str(mo_status) + ", Attempt: " + str(attempt))
125.         self.server.rockblock_response(bytes("MoStatus:"
126.                                              + str(mo_status) + " Attempt:" + str(at
    tempt), 'utf-8'))
127.
128.     def __mo_status_ok_response(self, attempt, mo_status):
129.         """Add docstring"""
130.         print("MoStatus: " + str(mo_status) + ", Attempt: " + str(attempt))
131.         self.server.rockblock_response(bytes("MoStatus:"
132.                                              + str(mo_status) + " Attempt:" + str(at
    tempt), 'utf-8'))
133.         self.clear_buffer()
134.
135.     def __handle_attempts(self, attempt):
136.         """Add docstring"""
137.         if attempt == 6:
138.             self.server.rockblock_response(b"Sending failed")
139.             print("Attempt(6): " + str(attempt))
140.             attempt += 1
141.             self.clear_buffer()
142.         if attempt == 5:
143.             print("Attempt(5): " + str(attempt))
144.             attempt += 1
145.             time.sleep(5)  # 5 min delay = 300
146.         if attempt in (3, 4):
147.             print("Attempt(3/4): " + str(attempt))
148.             #rnd = random.randint(0, 30)  # 0 - 30 sec delay
149.             attempt += 1
150.             time.sleep(5)
151.         if attempt in (1, 2):
152.             print("Attempt(1/2): " + str(attempt))
153.             rnd = random.randint(0, 5)  # 0 - 5 sec delay
154.             attempt += 1
155.             time.sleep(rnd)
156.         return attempt
157.
```

```
158.     def setup(self):
159.         """The initial setup. Run once."""
160.         pass
161.
162.     def flush(self):
163.         """Flushes the MO before power off"""
164.         pass
165.
166.     def flow_control(self):
167.         """Disables the flow control"""
168.         pass
169.
170.     def clear_buffer(self):
171.         """Clears the MO/MT buffer
172.
173.         # 0 - Clear the mobile originated (MO) buffer
174.         # 1 - Clear the mobile terminated (MT) buffer
175.         # 2 - Clear the MO and MT buffers
176.         """
177.         command = "AT+SBDD0"
178.         self.serial_conn.write(b"AT+SBDD0\r")
179.         if self.serial_conn.readline().strip().decode("utf-8") == command:
180.             if self.serial_conn.readline().strip().decode('utf-8') == "0":
181.                 self.serial_conn.readline()  # BLANK
182.                 if self.serial_conn.readline().strip().decode('utf-8') == "OK":
183.                     self.server.rockblock_response(b"Clearing buffer: Ok")
184.                     return
185.         self.server.rockblock_response(b"Clearing buffer: Failed")
186.         return
187.
188.     def close_serial(self):
189.         """Closes the serial connection with the rockblock"""
190.         self.serial_conn.close()
191.
192.     @staticmethod
193.     def list_ports():
194.         """Returns a list with the port rockblock is connected."""
195.         if sys.platform.startswith('win'):
196.             ports = ['COM' + str(i + 1) for i in range(256)]
197.         elif sys.platform.startswith('linux') or sys.platform.startswith('cygwin'):
198.             ports = glob.glob('/dev/tty[A-Za-z]*')
199.         elif sys.platform.startswith('darwin'):
200.             ports = glob.glob('/dev/tty.*')
201.         result = []
202.         for port in ports:
203.             try:
204.                 serial_conn = serial.Serial(port)
205.                 serial_conn.close()
206.                 result.append(port)
207.             except (OSError, serial.SerialException):
208.                 pass
209.         return result
210.
211.     def sbd_loe(self):
212.         command = 'AT+SBDLOE'
213.         self.serial_conn.write(bytes(command + "\r", 'utf-8'))
214.         for i in range(3):
215.             if i == 1:
216.                 respons = self.serial_conn.readline().strip().decode("utf-8")
217.                 print(respons)
218.             self.serial_conn.readline().strip().decode("utf-8")
219.         status = respons[8]
220.         if status == "0":
221.             tid = respons[10:]
```

```
222.             self.server.rockblock_response(bytes("Maintenance time remaining: " + st
    r(tid), 'utf-8'))
223.             print("Time remaining: " + tid)
```

E.5 Raspberry-kode på/av knapp (Python)

Koden som håndterer av/på knappen til Raspberry.

```
1.  #!/usr/bin/env python
2.
3.
4.  import RPi.GPIO as GPIO
5.  import subprocess
6.
7.
8.  GPIO.setmode(GPIO.BCM)
9.  GPIO.setup(3, GPIO.IN, pull_up_down=GPIO.PUD_UP)
10. GPIO.wait_for_edge(3, GPIO.FALLING)
11.
12. subprocess.call(['shutdown', '-h', 'now'], shell=False)
```

E.6 Raspberry-kode på/av knapp (Shell)

Koden som håndterer av/på knappen til Raspberry.

```
1.  #! /bin/sh
2.
3.  ### BEGIN INIT INFO
4.  # Provides:        PowerButton.py
5.  # Required-Start:   $remote_fs $syslog
6.  # Required-Stop:    $remote_fs $syslog
7.  # Default-Start:    2 3 4 5
8.  # Default-Stop:     0 1 6
9.  ### END INIT INFO
10.
11. # If you want a command to always run, put it here
12.
13. # Carry out specific functions when asked to by the system
14. case "$1" in
15.   start)
16.     echo "Starting PowerButton.py"
17.     /usr/local/bin/PowerButton.py &
18.     ;;
19.   stop)
20.     echo "Stopping PowerButton.py"
21.     pkill -f /usr/local/bin/PowerButton.py
22.     ;;
23.   *)
24.     echo "Usage: /etc/init.d/PowerButton.sh {start|stop}"
25.     exit 1
26.     ;;
27. esac
28.
29. exit 0
```

Appendiks F Diverse

F.1 MoStatus – Overføringsstatus

Disse meldingene sier noe om status til overføringen, som bruker får tilbakemelding om på HMI [15].

Number	MO session status
1	MO message, if any, transferred successfully, but the MT message in the queue was too big to be transferred.
2	MO message, if any, transferred successfully, but the requested Location Update was not accepted.
3..8	Reserved, but indicate MO session success if used.
10	GSS reported that the call did not complete in the allowed time.
11	MO message queue at the GSS is full.
12	MO message has too many segments.
13	GSS reported that the session did not complete.
14	Invalid segment size.
15	Access is denied.
16	ISU has been locked and may not make SBD calls (see +CULK command).
17	Gateway not responding (local session timeout).
18	Connection lost (RF drop).
19	Link failure (A protocol error caused termination of the call).
20..31	Reserved, but indicate failure if used.
32	No network service, unable to initiate call.
33	Antenna fault, unable to initiate call.
34	Radio is disabled, unable to initiate call.
35	ISU is busy, unable to initiate call.
36	Try later, must wait 3 minutes since last registration.
37	SBD service is temporarily disabled.
38	Try later, traffic management period (see +SBDLOE command)
39..63	Reserved, but indicate failure if used.
64	Band violation (attempt to transmit outside permitted frequency band).
65	PLL lock failure; hardware error during attempted transmit.

Tabell 2 - Mo status

F.2 IP-grad

Kapslingsgraden angis med bokstavene «IP» etterfulgt av to sifre (tabellene nedenfor viser sifrenes betydning). Dersom det ikke finnes noe krav til utstyrets beskyttelse mot noen av kategoriene erstattes sifferet med «X» [20].

Første siffer

Siffer	Beskyttelse av utstyret mot faste partikler	Beskyttelse av personer mot farlige deler med
0	—Ingen beskyttelse	Ingen beskyttelse
1	Større enn 50 mm	Store kroppsdeler, for eksempel en håndflate
2	Større enn 12,5 mm	Finger, 12,5 mm diameter
3	Større enn 2,5 mm	Verktøy, 2,5 mm diameter
4	Større enn 1 mm	Tråd, 1 mm diameter
5	Støvbeskyttet	Beskytter mot støv som kan skade produkt/innmat.
6	Støvtett	Komplett beskyttelse mot støvgjennomtregning.

Tabell 3 - Første siffer

Andre siffer

Siffer	Beskyttelse mot inntrengning av vann	Detaljer
0	Ingen beskyttelse	Ingen beskyttelse
1	Vertikale drypp	Vertikale drypp skal ikke ha skadelig virkning
2	Vertikale drypp ved helning opp til 15°	Vertikale drypp skal ikke ha skadelig virkning når utstyret har helningsvinkel opp til 15° i forhold til vertikallinjen
3	Dusj/regn	Dusj/regn mot kapsling i en vinkel opp til 60° i forhold til vertikallinjen skal ikke ha skadelig virkning
4	Sprut fra alle kanter	Sprut (kraftig dusj/regn) mot kapsling fra alle kanter skal ikke ha skadelig virkning
5	Spyling fra alle kanter	Spyling (fra slange/dyse) mot kapsling fra alle kanter skal ikke ha skadelig virkning
6	Kraftig spyling fra alle kanter	Kraftig spyling mot kapsling fra alle kanter skal ikke ha skadelig virkning
7	Kortvarig neddykking i vann	Ingen skadelig virkning ved kortvarig neddykking i vann (15 til 100 cm i inntil 30 minutter)
8	Varig neddykking	Ingen skadelig virkning ved varig neddykking i vann under betingelser spesifisert gjennom avtale mellom kunde og produsent av utstyret

Tabell 4 - Andre siffer

Appendiks G Timelister

Øystein				
Dato	**Start**	**Slutt**	**Timer**	**Hva ble gjort**
Forstudie				
09.01.2019	14:00	16:00	2,00	Møte
11.01.2019	09:00	15:00	6,00	Møte med 4Subsea
14.01.2019	10:00	13:00	3,00	Møte med veileder
15.01.2019	14:00	16:00	2,00	Forstudie / kravspesifikasjoner
18.01.2019	09:00	12:00	3,00	Forstudie
22.01.2019	14:00	16:00	2,00	Forstudie
23.01.2019				SYK
24.01.2019				SYK
25.01.2019				SYK
26.01.2019				SYK
27.01.2019				SYK
28.01.2019				SYK
29.01.2019	14:00	17:00	3,00	Forstudie
30.01.2019	14:00	18:20	4,50	Forstudie
Hoveddel				
07.02.2019	14:00	16:00	2,00	Ix Developer

08.02.2019	12:00	15:00	3,00	Sette opp github
11.02.2019	12:00	14:00	2,00	Møte
13.02.2019	11:00	14:00	3,00	Ix Developer og SQLite
14.02.2019	12:00	14:00	2,00	Forberedelse til møte 4Subsea
15.02.2019	09:30	17:30	8,00	Møte med 4Subsea
16.02.2019	14:00	18:00	4,00	Oppdatert Raspberry
18.02.2019	20:00	22:00	2,00	Socket i Python og C#
19.02.2019	08:30	19:00	10,50	Forts. Tester PC som HMI
20.02.2019	16:00	21:00	5,00	Satt opp SQLite
21.02.2019	11:00	14:00	3,00	Lage HMI og database script
22.02.2019	11:00	20:00	9,00	Tester sending av data
26.02.2019	11:00	20:00	9,00	Satt opp grafisk interface
28.02.2019	14:00	18:00	4,00	Lest på Python
01.03.2019	10:30	17:00	6,50	Tester rockblock
02.03.2019	12:00	17:00	5,00	Utfordringer med rockblock modul
04.03.2019	11:00	14:00	3,00	Leser om PySerial
"	17:00	20:00	3,00	Gjør klart til testing
"	21:30	22:30	1,00	Laget nytt gantt skjema
05.03.2019	10:00	19:00	9,00	Tester metoder
06.03.2019	11:00	17:00	6,00	Tester kode - Feiler
08.03.2019	11:00	16:00	5,00	Tester kode - Feiler
10.03.2019	14:00	16:00	2,00	Feilsøker i kode
13.03.2019	10:00	18:00	8,00	Tester kode - Feiler
14.03.2019	10:00	14:00	4,00	Utkast midtveispresentasjon
15.03.2019	10:00	18:00	8,00	Lage CSQ script
16.03.2019	12:00	17:00	5,00	Oppdaterer socket kode
18.03.2019	07:30	14:00	6,50	Feilsøker i kode
"	17:00	19:00	2,00	Oppdaterer socket kode
19.03.2019	12:00	18:00	6,00	Leser om exceptions
20.03.2019	07:30	17:00	9,50	Oppdaterer rpi og rockblock
"	20:00	22:00	2,00	Oppdatert HMI kode
21.03.2019	08:00	14:00	6,00	Oppdatert rpi kode
22.03.2019	08:00	18:00	10,00	Tester sending av data
23.03.2019	10:00	20:00	10,00	Feilsøker i kode
24.03.2019	12:00	14:00	2,00	Ryddet i kode
25.03.2019	08:00	12:00	4,00	Feilsøker i kode
26.03.2019	14:00	17:00	3,00	Tester sending av data
27.03.2019	09:00	14:00	5,00	Oppdatert rpi kode
29.03.2019	11:00	19:00	8,00	Oppdatere timeliste/prosjektlogg
30.03.2019	12:00	16:00	4,00	Lager et forslag til HMI endring
"	19:00	22:00	3,00	"
31.03.2019	10:00	14:00	4,00	Fortsetter med Raspberry endringer
"	17:00	21:00	4,00	Fortsetter med Raspberry endringer
01.04.2019	17:00	20:00	3,00	Lager statusrapport til midveis
02.04.2019	11:00	13:00	2,00	Forberedelse til midveispresentasjon
03.04.2019	11:00	17:00	6,00	Forberedelse til midveispresentasjon
04.04.2019	08:00	10:00	2,00	Midtveispresentasjon
12.04.2019	09:30	16:00	6,50	HMI hos 4Subsea

15.04.2019	09:30	14:00	4,50	HMI hos 4Subsea
16.04.2019	09:00	14:00	5,00	HMI hos 4Subsea + Skriving
19.04.2019	09:00	11:00	2,00	Fører inn resten av forstudiet, med små endringer
28.04.2019	14:00	16:00	2,00	Arbeid/planlegging rapport
02.05.2019	10:00	15:00	5,00	HMI hos 4Subsea
03.05.2019	11:00	16:00	5,00	HMI hos 4Subsea
04.05.2019	10:00	13:00	3,00	Skrive på bachelor
06.05.2019	08:00	16:00	8,00	Skrive på bachelor
08.05.2019	08:00	16:00	8,00	Skrive på bachelor
10.05.2019	09:00	16:00	7,00	Seminar og arbeid
21.05.2019	13:00	17:00	4,00	Skrive på bachelor
22.05.2019	10:00	17:00	7,00	Skrive på Bachelor
23.05.2019	09:00	21:00	12,00	Feilsøket i HMI kode
24.05.2019	09:00	16:00	7,00	Laget start script på raspberry og testet dette.
25.05.2019	10:00	16:00	6,00	EXPO plakat og bachelor
"	18:00	21:00	3,00	Skrive på bachelor
26.05.2019	10:00	17:00	7,00	Skrive på bachelor
26.05.2019	18:00	20:00	2,00	Fjernet litt kommentarer og endret litt feil.
27.05.2019	09:00	22:00	13,00	HMI hos 4Subsea og feilsøking i raspberry kode iforhold til checksum
28.05.2019	09:00	21:00	12,00	HMI hos 4Subsea og feilsøking i Raspberry kode iforhold til checksum
29.05.2019	07:30	22:00	14,50	Feilsøking av checksum. Fant ikke ut hva som er feil...
30.05.2019	09:00	18:00	9,00	Rapport
31.05.2019	08:00	13:00	5,00	Rapport
			417,0	

S t e i n R u n e				
Dato	**Start**	**Slutt**	**Timer**	**Hva ble gjort**
			F o r s t u d i e	
06.01.2019	18:00	20:00	2,00	Gjennomgå tidligere oppgave.
08.01.2019	14:15	17:00	2,75	Undervisning
09.01.2019	14:00	16:30	2,50	Lese tidligere oppgave. spørsmål til 4sub
"	20:00	22:30	2,50	Forstudie
11.01.2019	11:30	14:00	2,50	Møte med 4Subsea
12.01.2019	11:00	15:00	4,00	Forstudie
13.01.2019	13:00	15:00	2,00	Forstudie
14.01.2019	10:00	14:00	4,00	Møte med Kjell-Eivind + Forstudie
"	20:00	23:00	3,00	Forstudie
15.01.2019	14:15	17:00	2,75	Undervisning
16.01.2019	15:45	18:00	2,25	Forstudie
18.01.2019	19:00	21:00	2,00	Raspberry
19.01.2019	20:00	24:00	4,00	Raspberry
20.01.2019	20:00	22:00	2,00	Forstudie
21.01.2019	12:00	16:00	4,00	Forstudie, skrive rapport
22.01.2019	14:15	16:00	1,75	Undervisning
"	20:00	22:00	2,00	Raspberry

29.01.2019	14:00	17:00	3,00	Forstudie, Spørsmål til 4Subsea
				H o v e d d e l
04.02.2019				SYK
05.02.2019				SYK
06.02.2019				SYK
07.02.2019				SYK
08.02.2019				SYK
09.02.2019	12:30	17:30	5,00	Innføring i 3D print
15.02.2019	10:00	13:00	3,00	Møte med 4Subsea++
19.02.2019	14:15	16:00	1,75	Undervisning
05.03.2019	14:00	18:00	4,00	Status møte
06.03.2019	11:00	17:00	6,00	Flytskjema. Tester RockBLOCK
08.03.2019	11:00	17:00	6,00	Tester Rockblock.
10.03.2019	18:00	21:00	3,00	Fusion360
11.03.2019	21:00	00:00	3,00	Fusion360
13.03.2019	11:00	18:00	7,00	Test Rockblock med Raspberry
15.03.2019	11:00	18:00	7,00	Teste stabilitet i dekning for RockBLOCK
17.03.2019	15:00	23:00	8,00	Fusion360
18.03.2019	16:00	19:00	3,00	Designe utkast til innkapsling
19.03.2019	16:00	01:00	9,00	Designe utkast til innkapsling
20.03.2019	11:00	17:00	6,00	Fusion360
22.03.2019	11:00	18:00	7,00	Testkjøring av Rockblock
"	22:00	00:00	2,00	Rapport skriving. Timer
25.03.2019	11:00	14:00	3,00	Testkjøring av kode.
26.03.2019	14:00	17:30	3,50	Test og feilsøking Rockblock
28.03.2019	18:00	23:00	5,00	Cura
29.03.2019	11:00	16:00	5,00	Oppdatere timeliste/prosjektlogg
30.03.2019	11:00	18:00	7,00	Montere og testkjøre 3D printer
31.03.2019	12:00	18:00	6,00	Kalibrering og feilsøking
01.04.2019	11:00	15:00	4,00	Rapport/ Midtveis
03.04.2019	11:00	17:00	6,00	Forberedelse til midveispresentasjon
04.04.2019	08:00	10:00	2,00	Midtveis
05.04.2019	12:00	18:00	6,00	Raspberry
07.04.2019	13:00	20:00	7,00	Raspberry, Fusion
12.04.2019	09:30	16:00	6,50	HMI hos 4Sub
13.04.2019	18:00	21:00	3,00	Raspberry
14.04.2019	18:00	22:00	4,00	On/Off btn, Fusion
15.04.2019	09:30	14:00	4,50	HMI hos 4Sub
16.04.2019	09:00	12:00	3,00	HMI hos 4Sub
"	16:00	20:00	4,00	Fusion
17.04.2019	08:00	15:00	7,00	Cura
19.04.2019	13:00	17:00	4,00	Raspberry
21.04.2019	16:00	19:00	3,00	Cura
22.04.2019	12:00	17:00	5,00	Kode for knapp + 3D-Print
23.04.2019	14:00	16:00	2,00	Lodding, HW etc
24.04.2019	12:00	16:00	4,00	Kranglet med 3d-Printer
27.04.2019	12:00	15:00	3,00	Print
28.04.2019	11:00	14:00	3,00	Testing og feilsøking av power btn

"	18:00	22:00	4,00	Montering
29.04.2019	11:00	17:00	6,00	Btn, feilsøking bilboteker Rpb
"	20:00	21:00	1,00	Rapport skriving
01.05.2019	15:00	17:30	2,50	Rapport skriving
"	20:00	22:00	2,00	Rapport skriving
02.05.2019	08:30	15:30	7,00	HMI hos 4Subsea
03.05.2019	09:30	16:00	6,50	Verksted 4Subsea
10.05.2019	11:00	14:00	3,00	Seminar
15.05.2019	23:00	02:00	3,00	Skrive på Bachelor
18.05.2019	18:00	21:00	3,00	Skrive på Bachelor
21.05.2019	13:00	18:00	5,00	Skrive på Bachelor
22.05.2019	10:00	17:00	7,00	Skrive på Bachelor
23.05.2019	10:30	14:30	4,00	iX Developer, Møte
25.05.2019	12:00	16:00	4,00	Div oppgave skriving. Plakat
"	21:00	22:00	1,00	Expo plakat
26.05.2019	11:00	17:00	6,00	Skrive på Bachelor
27.05.2019	09:00	19:30	10,50	HMI hos 4Subsea
"	22:00	23:30	1,50	Justeringer på 3D-modell
28.05.2019	09:00	21:00	12,00	HMI hos 4Subsea og feilsøking i raspberry kode iforhold til checksum
29.05.2019	10:00	20:00	10,00	Raportskriving, justering og printing av 3D-modell.
30.05.2019	10:00	18:00	8,00	Rapport
31.05.2019	08:00	13:00	5,00	Rapport
			352,3	

Eivind				
Dato	**Start**	**Slutt**	**Timer**	**Hva ble gjort**
			Forstudie	
08.01.2019	14:00	17:00	3,00	Undervisning
09.01.2019	14:00	16:00	2,00	Møte
10.01.2019	13:00	15:00	2,00	Forberedelse til 4subsea
11.01.2019	11:00	14:00	3,00	Møte med 4subsea
14.01.2019	10:00	13:00	3,00	Møte med veileder
15.01.2019	14:00	17:00	3,00	Undervisning
17.01.2019	13:00	15:00	2,00	møte
22.01.2019	14:00	17:00	3,00	Undervisning
24.01.2019	13:00	15:00	2,00	møte
26.01.2019	11:00	13:00	2,00	selvstendig
29.01.2019	11:00	15:00	4,00	Forstudie
31.01.2019	12:00	16:00	6,00	Forstudie
			Hoveddel	
13.02.2019	12:00	14:00	2,00	møte
14.02.2019	10:00	13:00	5,00	Raspberry pi+forberedelse til møte
15.02.2019	10:00	13:00	3,00	Møte med 4subsea
19.02.2019	14:00	17:00	3,00	Undervisning
05.03.2019	14:00	18:00	4,00	status møte
06.03.2019	14:00	18:00	4,00	Kurs i kode med Øystein, flowchart

08.03.2019	11:00	16:00	5,00	Testing av RockBLock
09.03.2019	11:00	17:00	6,00	Lese om Iridium satelittnettverk og RockBlock
10.03.2019	11:00	17:00	6,00	Python + repetisjon programmeringsfag
12.03.2019	10:00	18:00	8,00	Lære Python programmering
13.03.2019	11:00	17:00	6,00	Koding og testing: Raspberry til Rockblock kommunikasjon
14.03.2019	11:00	16:00	5,00	Programmering: Python, Raspberry
15.03.2019	11:00	16:00	5,00	Tester med RockBlock
16.03.2019	11:00	17:00	6,00	Repetere nettverksprogrammering
17.03.2019	11:00	17:00	6,00	Bli kjent med Raspberry
18.03.2019	09:00	17:00	8,00	Python
22.03.2019	10:00	18:00	8,00	Tester sending av data
23.03.2019	11:00	17:00	6,00	Research til alternative løsninger
24.03.2019	11:00	17:00	6,00	Fortsetter med Python selvstudie
25.03.2019	09:00	17:00	8,00	Python + gitHub
26.03.2019	14:00	18:00	4,00	Testing
27.03.2019	11:00	17:00	6,00	Legge til funksjon
28.03.2019	10:00	16:00	6,00	fikser på kode
"	17:00	20:00	3,00	
01.04.2019	10:00	16:00	7,00	Endring på metode i python Raspberry Pi + rapport/midtveis
"	21:00	22:00	1,00	Rapport / midtveis
02.04.2019	10:00	16:00	6,00	Rapport / midtveis
03.04.2019	14:00	17:00	3,00	Forberedelse til midveispresentasjon
04.04.2019	08:00	10:00	2,00	Midtveispresentasjon
12.04.2019	09:00	16:00	7,00	HMI hos 4Subsea
15.04.2019	09:00	14:00	5,00	HMI hos 4Subsea
16.04.2019	09:00	12:00	3,00	HMI hos 4Subsea
28.04.2019	09:00	16:00	7,00	Arbeid/planlegging rapport
02.05.2019	09:00	15:00	6,00	HMI hos 4Subsea
03.05.2019	09:00	16:00	7,00	HMI hos 4Subsea
04.05.2019	11:00	18:00	7,00	Lese kode, repetere C#
05.05.2019	11:00	18:00	7,00	Lese kode, repetere C#, forberedelse til presentasjon
09.05.2019	19:00	21:00	2,00	Forberedelse til presentasjon, seminar
10.05.2019	10:00	17:00	7,00	Seminar og arbeid
27.05.2019	13:00	19:00	6,00	Rapport
28.05.2019	09:00	21:00	12,00	HMI hos 4Subsea+ arbeid med rapport + EXPO
29.05.2019	10:00	22:00	12,00	Ferdiggjøring av Bacheloroppgave
30.05.2019	10:00	22:00	12,00	Ferdiggjøring av Bacheloroppgave
31.05.2019	08:00	13:00	5,00	Levere rapport
			288,0	

Appendiks H **Prosjektlogg**

Dato	Hva ble gjort
	Forstudie
06.01.2019	Så på bacheloroppgave som ble gjort vår 2018 av en annen bachelor gruppe
08.01.2019	Forelesning om bacheloroppgave
09.01.2019	Bachelor gruppen har et kort møte. Her blir det lagt en plan for hva vi skal spørre 4Subsea om på møtet vi skal ha med dem den 11.01.
11.01.2019	Første møte med 4Subsea. Blir enig i at vi setter oss inn i forrige gruppen sin bacheloroppgave, for så å gi de en tilbakemelding på hva vi ønsker/anbefaler. Se møtereferat for mer utdypende informasjon.
12.01.2019	Fortsetter arbeidet med å fordype oss i bacheloroppgaven fra i fjor
13.01.2019	Fortsetter arbeidet med å fordype oss i bacheloroppgaven fra i fjor
14.01.2019	Har første møte med intern veileder hvor vi forteller hva vi fant ut/ble enig med 4Subsea om.
15.01.2019	Fortsetter arbeidet med å fordype oss i bacheloroppgaven fra i fjor og starter arbeidet med å lage kravspesifikasjoner. Forelesning om bacheloroppgave.
16.01.2019	Starter skrivingen av forstudie og fortsetter med forypningen.
18.01.2019	Starter arbeidet med å lese om Raspbery Pi og Adafruit
19.01.2019	Fortsetter arbeidet med å lese om Raspbery Pi og Adafruit
20.01.2019	Fortsetter skriving på forstudie
21.01.2019	Fortsetter skriving på forstudie
22.01.2019	Forelesning om bacheloroppgave og jobbing med forstudie
29.01.2019	Fortsetter med skriving på forstudie. Forberede også noen spørsmål knyttet til forstudie som vi sender mail om til 4Subsea
30.01.2019	Legger inn en siste finish på forstudie og leverer.
	Bachelor
07.02.2019	Leser om Ix developer
08.02.2019	Setter opp Github for at vi skal ha en felles plass for koden
09.02.2019	Tar en innføring i 3D printing for å se på muligheten for å lage en prototype til EXPO.
11.02.2019	Har et møte med bachelor gruppen hvor vi prater om veien fram til nå og veien videre. Forbereder oss også til møte med 4Subsea den 15.02
13.02.2019	Fortsetter arbeidet med å lese om Ix developer. Starter også å lese om databasen SQLite som er på HMI.
14.02.2019	Siste forberedelser før møte med 4Subsea
15.02.2019	Møte med 4Subsea. Blir her enig om at vi går for løsning med rockblock og raspberry pi. Får utlevert utstyret (raspberry pi og rockblock).

16.02.2019	Oppdaterer raspberry pi og sjekker koden som ble laget av forrige gruppe.
18.02.2019	Leser om sockets i python3 og oppdatere socket koden på raspberry.
19.02.2019	Leser generellt om python3 kode. Ser også over socket koden på HMI. Ser om det er mulig å simulere PC(HMI) med SQLite database mot raspberry. Dette viser seg å være mulig etter litt testing. Forelesning om bacheloroppgave.
20.02.2019	Legger til en SQLite database og starter å kode den delen.
21.02.2019	Lager et enkelt script som simulerer HMI med en database.
22.02.2019	Fortsetter arbeidet med scriptet. Klarer å sende data fra HMI(PC) til raspberry.
26.02.2019	Fortsetter arbeidet med HMI, hvor det blir lagt til et grafisk interface for at simuleringen skal være så ekte så mulig.
28.02.2019	Fortsetter å lese om python3
01.03.2019	Forberedet HMI koden. Prøvd å finne en løsning for å sende data til rockblock
02.03.2019	Møter problemer med rockblock modul som vi har hentet på nettet. Tror det kan være en kombinasjon av python2 vs python3 og annen rockblock modul.
04.03.2019	Oppdager at det også er problemer i modulen med at seriell modulen som er brukt er utdatert. Retter opp i disse feilene. Lager også et nytt gantt skjema.
05.03.2019	Gruppen har en lite møte hvor vi tar opp status. Begynner å teste metodene i modulen.
06.03.2019	Lager et flyskjema for å sikre at alle er enig i hvordan ting er kodet. Går utendørs for å teste første gang om vi får sendt med Rockblock direkte koblet til PC. Vi fikk påvist at vi har dekning, men fikk ikke til å sende melding.
08.03.2019	Vi forsøker igjen å få sendt melding med rockblok. Vi kobler den direkte til PC og bruker Putty til å skrive kommandoer som sendes til Rockblock. Etter to forsøk får vi omsider sendt melding og ser at rockblock ikke er defekt. Vet ikke hvorfor det ikke virket første gang.
10.03.2019	Fortsatt ikke funnet ut hva feilen er. Men rydder opp i koden for at det skal bli lettere å feilsøke. Starter også å se på Fusion 360 for at vi skal ha mulighet for å 3D printe.
11.03.2019	Fortsetter å se på Fusion 360.
13.03.2019	Prøvde første gang å sende data fra HMI til rockblock. Det gikk ikke. Dette er nok på grunn av scriptet vi hentet fra mackerspace. Prøvde etterpå rett fra raspberry med egen lagd kode, noe vi fortsetter med de kommende dagene. Her fikk vi etter mange forsøk sendt, vet ikke hvorfor. Må teste/feilsøke hvorfor. Fikk også feilkode 32 (No network signal)
14.03.2019	Lager utkast til power point som vi skal bruke i midtveispresentasjonen

15.03.2019	Lager et test script som skal sjekke CSQ (signal styrke) over en periode på flere minutter. Gjør noen forsøk og får tilbake data som kan bli verdifull
16.03.2019	Oppdater socket kommuikasjonen til å bli lik begge flow chartene. Ikke testet om det fungere, men burde være ok.
18.03.2019	Oppdatert videre på socket kode. Vi har problemer med at man i worker thread ikke kan oppdatere GUI. Starter også å desgine utkastet til innkasplingen som skal inneholde raspberry pi og rockblock.
19.03.2019	Leste om python socket kommunikasjon for å prøve å finne ut hvordan skrive god exception kode. Lagt inn basis feilhåndtering i koden. Fortsetter også på designet til innkapslingen.
20.03.2019	Fortsetter med å vidreutvikle raspberry pi kode og rockblock kode. Ser videre på innkapslingen.
21.03.2019	Ryddet litt mer i kode. Legger til funksjoner i python kode som gjør at den vil loop igjennom og sende respons tilbake til HMI
22.03.2019	Endret mer på Python koden slik at den ble ryddigere. Fikk testet det komplette skriptet i dag ved å prøve å sende til satellitt. Første forsøk fikk vi sendt 3 ganger 340 bytes. Andre gang 2 ganger 246 bytes. Må feilsøke på hvorfor koden ikke fungerer opptimalt
23.03.2019	Arbeid - Fortsetter å feilsøke. Starter dagen med problemer hvor porten plutselig har forandret seg. Vi prøver å legge inn et script som auto finner port. Satser på at den finner rikig hver gang. Ellers må vi feilsøke på checksum koden fordi den plutselig ikke fungerer. Den funket igår...! Mulig også funnet ut hvor vi har sendt dobbelt.
24.03.2019	Endret litt små ting i kode. Lagt til 2 x issue med forklaringer på github. Også endret litt på mappe og fil navn slik at det ble litt mer ryddig.
25.03.2019	Lagt til enda ett issue på github. Diskuterer litt virkemåten og om dette er veien å gå fordi vi møter endel problemer på veien. Blir enig om å fortsette videre.
26.03.2019	Tester om overføring fungerer. Finner feil med timeout på serial object som mest sannsynelig har laget problemer med koden. Laget kommentarer rundt dette. Ellers fikser vi et script som skal håndtere hvis det er maintance på satellitten.
27.03.2019	Fjernet en del kode som ble lagt inn som feilkode i forbindelse med at timeouten ble økt fra 5 sekunder til 30 sekunder. Ikke lagt til noen feilhåndteringer, noe som må ses på senere. Kan være at det i praksis ikke er noe behov.
29.03.2019	Oppdaterer timelister og prosjektlogg slik at det er klart til midveispresentasjon.

30.03.2019	Ser på mulighet for å legge til rette for wifi og 4g på hmi. Lager et utkast som blir lastet opp på github. Starter også test printing av 3D modell.
31.03.2019	Fortsetter å implementere endringer som legger til rette for 4G og wifi.
01.04.2019	Forbereder statusrapport, timelister og presentasjon til midtveies.
02.04.2019	Forberedelser til midtveispresentasjon
03.04.2019	Forberedelser til midtveispresentasjon
12.04.2019	Vi reiser ut til 4Subsea for å starte på koding av HMI. Får låne PC av Henrik og gjør oss litt kjent med IX-Developer som brukes for å programmere HMI
15.04.2019	Hos 4Subsea. PC'en vi brukte for å programmere HMI går i blåskjerm etter en Windows oppdatering. Får ikke gjort så mye mer i dag da vi heller ikke har fått brukerkonto for å aksessere PC da Henrik hadde glemt dette og nå er på ferie.
16.04.2019	Hos 4Sub. Får lånt oss en ny PC. Dagen går med på å skaffe brukerkonto, men da vi ikke får tilgang til IX-developer og prosjektet vi alt hadde startet på så blir vi enig om at vi får komme tilbake igjen når folk er tilbake fra påske ferie.
22.04.2019	3D-printet utkast av i innkapsling. Skrevet ferdig og redigert kode for Av/på Knapp.
23.04.2019	Loddet kabler til Av/På knapp
24.04.2019	Skulle 3D-printe siste utkast av innkapsling. Fikk problemer med printer og brukte mye tid på å feilsøke denne.
27.04.2019	Fikk i gang 3D-printer og fikk printet innkapsling.
28.04.2019	Montering, testing og feilsøking av Av/på knapp.
29.04.2019	Fant at det var noen problemer med GPIO bibloteket som ble brukt i koden for Av/På knappen. Fikk rettet opp i dette og knappen virker nå som den skal.
02.05.2019	Ute hos 4Subsea. Får en pc med IX-Dev opp å gå og kommer oss i gang med å legge inn kode for HMI. Møter på et par problemer vedrørende threads i koden som ikke kjører som de skal.
03.05.2019	Er på verkstedet til 4Subsea hvor vi prøver å finne ut problemene fra i går. Det viser seg å være vanskelige, slik at vi ikke får en fungerende versjon.
04.05.2019	Skriver på bacheloroppgaven.
06.05.2019	Skriver på bacheloroppgaven.
08.05.2019	Skriver på bacheloroppgaven.
10.05.2019	Holder et kort foredrag om oppgaven på et seminar for elektroavdelingen og fortsetter med å skrive på bacheloroppgaven.
21.05.2019	Skriver på bacheloroppgaven.
22.05.2019	Skriver på bacheloroppgaven.

23.05.2019	Har et kort møte med veilder hvor vi får tilbakemeldinger på utkastet. Jobber også med å feilsøke på HMI koden. Har lastet ned en trial versjon av iX Developer for å teste dette. Etter mye testing får vi en gjort noe vellykede tester som må teste ut på HMI.
24.05.2019	Starter arbeidet med å lage et start script for kode på raspberry. Dette viser seg å være enklere enn først antatt og vi får en fungerende utgave ganske fort. Fortsetter videre med å skrive på bacheloroppgave.
25.05.2019	Starter på EXPO plakat og skriver på bacheloroppgaven.
26.05.2019	Fullfører EXPO plakat og skriver på bacheloroppgaven. Har per nå god kontroll på skriving, slik at det som gjenstår er en siste finpuss.
27.05.2019	Ute hos 4Subsea. Skriver på kode for HMI. Luket ut en god del feilmeldinger i.f.m. MO-status meldinger. Fått kommunikasjon mellom HMI og Rpi til å virke.
28.05.2019	Ute hos 4Subsea. Skriver på kode for HMI. Har en feil i koden som vi ikke finner ut av. Må fortsette å feilsøke.
29.05.2019	Fått problemer med validering av chk sum i kommunikasjon mellom raspberry og rockblock. Har i den forbindelse gjort en del feilsøking.
30.05.2019	Skriver på bacheloroppgaven.
31.05.2019	Retter opp enkle feil og mangler i oppgaven.