CO_2NSL (Datalogger)

Sune Andersen
S030762

Kongens Lyngby 2012
IMM-Mcs-2012-0084

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk IMM-Mcs-2012-0084

Summary (English)

The following report will describe the development of a computer system, and act as the final exams project for Sune Andersen prepared at Informatics Mathematical Modelling, the Technical University of Denmark acquiring the candidate degree in computer engineering. The project lasts 26 weeks, which must cover analyses, design, implementation and documentation of the project. Risø National Laboratory is getting more and more requests from The danish government on how to save energy. One of the main issue is saving money on power,special when it comes to streetlight. Before the end of the year 2012,1500 street lamps around Copenhagen will be changed for light sources with low power consumption. Technical and Environmental turn down the energy as a part of Copenhagen goal of reducing the citys CO2 emissions by 20 percent by the end of year 2015. But how much power will the new lamps comsume? And can a street lamp produce sufficient power even in Denmark?. Here will a low cost & lowpower Datalogger come handy. The data logger is an electronic device that records earthquakes(Sensor network), Wind ,daylight ,power used/produced on the street lamp over time. Data will then be uploaded via a wireless radio MESH network(868 Mhz) to a database server for later analyze. The Prototype is developed on two microcontrollers(AVR and ARM Cortex-A8) with the low power and with fault tolerant in mind, equipped with extra storage for offline catching(like a uSD(16/32Gb)).The ARM CortexA8-board is running a full version of Ubuntu(OMAP), with Apache-webserver,PHP and MySQL-database for local catching of data, in case of the network is offline. Data will then be sync with the database server then there is connectivity. Controlling the Datalogger device can be done from the control centers webinterface or on the device it self(via Web or SSH). The device can even be used for other purposes like a (MESH) WIFI net, something like freifunk in Berlin & WNDW. In a catastrophe area the lamp-network will still be running (because it is off-grid), even when the infrastructure is destroyed or very heavy loaded.

Forord

Denne rapport dokumenterer det arbejde,der er udført i forbindelse med mit afgangsprojekt som civilingeniør på IMM ved Danmarks Tekniske Universitet(DTU) fra Februar 2012 til August 2012 med ca. 3 ugers orlov. Igennem de sidste 3 års fuldtidstudie har jeg specialiseret mig inden for embededsystemer og IT sikkerhed på forskellige måde. Begge dele ser jeg som en disciplin som jeg vil bruge mere tid på efter endt uddannelse.

Projektet er udført på Risø hos Fotonik,som mest forsker inden for lys og de forskelllige aspekter det må afstedkomme. Jeg har tidligere har haft den store glæde at skrive mit diplomprojekt for AIT på Risø.Der finde en artikel i DTU Avisen nr.6-2007 omkring mit tidligere Risø-projekt. Dette projekt er udfør alene med assistance fra Fotoniks ansatte,Henning Engelbrecht Larsen har været min vejleder på Risø og Lektor Finn Gustafsson har været vejleder ved DTU. Projektet er opdelt i 5 dele,som er Etablering,Analyse,Design,Konstruktion og Testfasen(som er delt i 3 mindre dele). Se mere i afsnit **3.0.1 Tidsestimering**. Store dele af dette projekt vil (på sigt)være offentlig og under GNU General Public License (forkortes often GPL),i håbet om at mit projekt/ideer kan bruges af andre end Risø og da jeg er stor tilhænger af opensource ideen.
Denne rapport er skrevet i LaTeX med TexMaker og MikTek.
Projektet online : `co2nsl.deadmeat.dk` Mail : co2nsl@outlook.com

Lyngby, 03- august-2012

Sune Andersen
S030762
suna @ student.dtu.dk

Taksigelser

Det har været en lang og besværlig proces. Jeg var så super heldig at kende nogle ansatte fra Risø-fotonik som ved et tilfælde nævnte
at de stod og manglede studerende der kunne løse forskellige teknisk opgave de ikke selv havde resourcer til.

Den største tak skal lyde til Søren Stentoft Hansen fra Fotonik-DTU for at hjælpe mig med elektroniken og være en super støtte med forskellige tekniske emner som jeg ikke lige kunne finde løsninger på.

Ikke mindst en stor tak hele IT-afdelingen hos RISØ-Roskilde for at finde sig i mine forskellige test miljøer på deres netværk.

- Jeg skal aldrig glemme AIT ! ;-)

Tak til vejlederen fra Risø Henning Engelbrecht Larsen for de mange input omkring Labview.
Tak til vejlederen fra DTU Lektor Finn Gustafsson for input til min rapport.
Tak til Carsten Witt for at hjælpe med min Grøn Dyst Poster.
Tak til Stine Ellermann for at vil dele kontor med et rodehoved.
Tak til Peter Behrensdorff Poulsen for at tilbyde mig en ansættelse efter projekt afslutning.

Tak til min elskede Betina, fordi du udviser tålmodighed med en kæreste der skal nørd Linux servere og netværk nat og dag.

Denne rapport sætter et punktum for en lang og yderst spændende rejse. Jeg har under hele mit ophold betragtet det som en meget stor ære at kunne være en del af denne fantastiske fortælling, om en store forvandling af Risøs måde at lave dataopsamling på, og ikke mindst kan jeg i al fremtid sige:
Jeg har skrevet Eksamens opgave hos Risø ***2 gange!***
- Sune S. Andersen

Contents

Chapter 1

Identificering af interessenter

CO_2.-neutral byrumsarmatur er udendørsbelysning baseret på vindkraft , solkraft og LED-teknologi. Gate 21 har etableret et samarbejde mellem kommuner, designere, forskningsinstitutioner og producenter om at udvikle, afprøve og demonstrere ny energirigtig gadebelysning. Formålet er at skabe energi- og driftsbesparelser i kommunernes udendørsbelysning. LED er attraktive lyskilder til udendørsbelysning på grund af deres høje energieffektivitet, lange levetid og mindre miljøbelastning end de traditionelle lyskilder. Lysdioder giver derudover mulighed for at udvikle "stand-alone" lampeenheder, hvor energien til dioderne baseres på solceller implementeret i selve lampen og vindkraft fra små vindmøller.

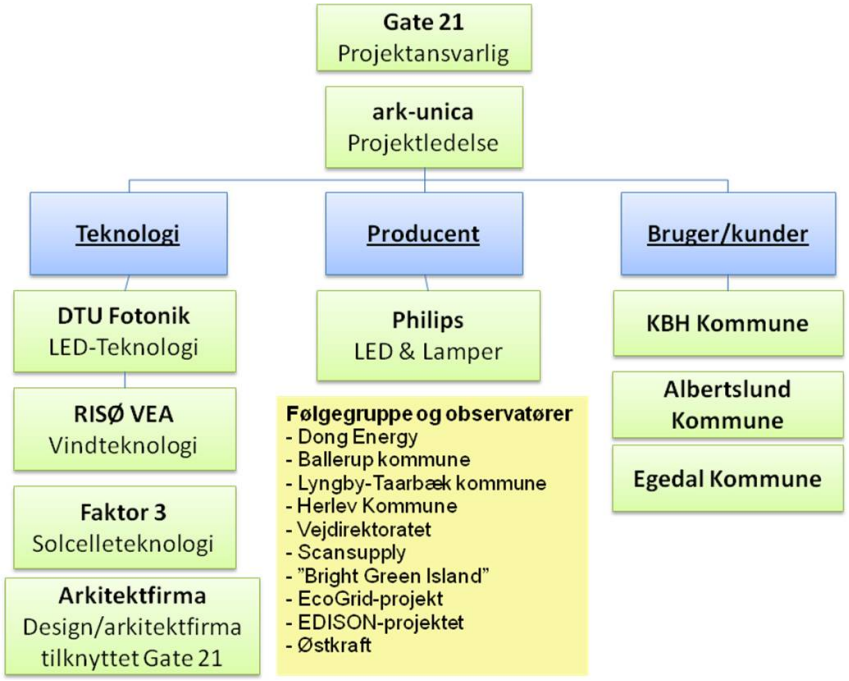


Figure 1.1: Projekt gruppen[1]

1.1 Gate21 gruppen

A-lampen – det digitale lys Albertslund Kommune har i partnerskab med blandt andet Philips, DTU Fotonik, Dong Energy og Odgård Design udviklet A-lampen og dermed banet vejen for et skift fra "glødepæren" til "det digitale lys" i udendørsbelysningen. Udviklingsprojekt Hovedelementerne i udviklingsprojektet af ny LED-belysning vil omfatte: Dataindsamling, research inden for den nyeste LED-, vind- og solcelleteknologi. Idéoplæg, analyser, design og udvikling af armaturer. Fremstilling, prøveopstillinger og test af prototyper i fuld skala. Formidling af resultater gennem udstillinger, publikationer, konferencer og seminarer samt markedsintroduktion. Finansiering: Samlet budget 5.2 mio. kr., hvoraf der er opnået støtte på 1.6 mio. kr. fra Elforsk.Partnere: ark-unica, DTU Fotonik, RISØ VEA, Faktor 3, Philips Lighting A/S, Gate 21, Københavns Kommune, Albertslund Kommune, og Egedal Kommune. Tidsramme: 2 år. [1]

1.2 Københavns Klimaplan 2025

Elforbruget omfatter ikke forbruget til eldrevet togdrift (80 GWh) og elopvarmning af private husholdninger (14 GWh).Elforbruget er årsagen til omkring halvdelen af CO_2-udledningen fra byen København. Så indsatsen for at nedbringe elforbruget er selvsagt af afgørende betydning for at nå Miljømetropol-målet om 20 procent reduktion af CO_2-udledningen i forhold til 2005. Det samlede elforbrug steg 3 procent fra 2005 til 2010, hvor forbruget var ca. 2.600 GWh alt inklusiv. ***Kilde :*** `http://www.kk.dk/byenslys`

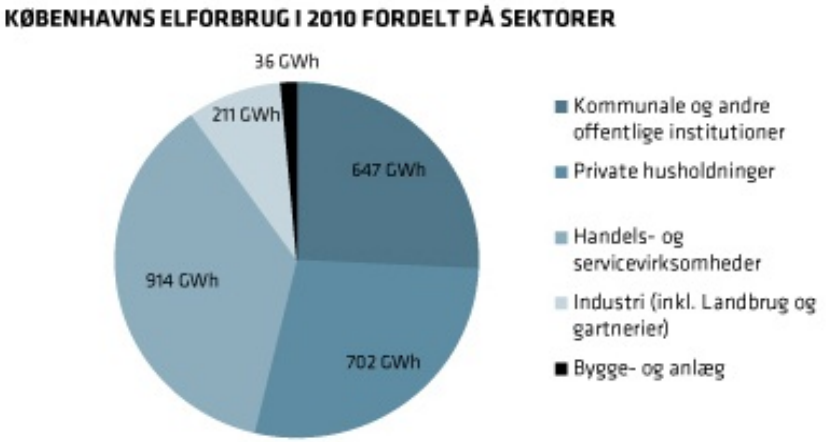


Figure 1.2: Projekt gruppen

Chapter 2

Opgaven fra Risø

På Risøs test område står 5 CO_2.-neutral byrumsarmatur som kommer med data fra forskellige analoge sensorer,så som produktion/forbrug af strøm og spænding,ialt 12 stk. målepunkter i hver lampe. Det ønskes at man kan justere måle hastigheden.Gerne i retning af noget embedsystemer,eventuelt med et kommando interface som styres på en webserver. Der ønskes en trådløst netværks løsning, som er så strøm besparende som overhoved muligt. Alle enheder skal kunne tåle det danske vejr(hele året!),strømudfald og netværks udfald. Måle data skal lægges ind i en database server som står på Risø-DTUs netværk, gerne Opensource baseret. Fra databasen skal der kunne laves data udtræk til Labview/Matlab, hvor der også skal laves et program til MatLab. Som vil blive brugt af andre studerende/forskere under andre dele af projektet. ***Billed og information er fra en "The Nheohybrid 400 manual"som er udviklede/solgt af nheolis[14]***.

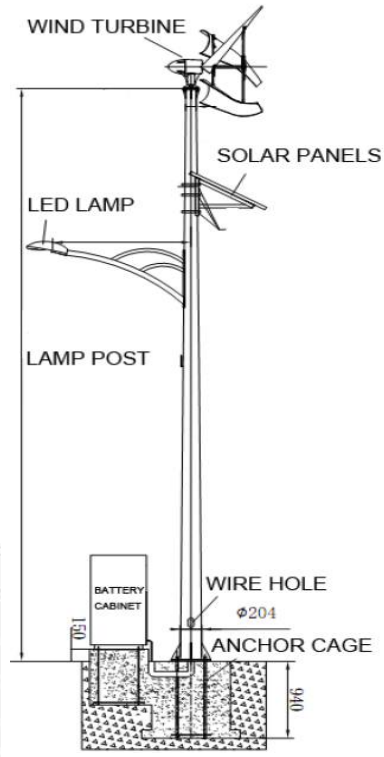


Figure 2.1: Eks.på GadeLampe[14]

Specification Of Wind Turbine			
Model	nheowind 3D 04	Rated Power	300w
Rated Voltage	25VAC	Rated Wind Speed	12m/s
Start-up Wind Speed	2m/s	Cut-in Wind Speed	2.5m/s
Maximum Power	400W	Rotor Diameter	1.5m
Qty Of Blades	3pcs	Survival Wind Speed	45m/s
Total Weight of Wind Turbine	32kg	Working Temperature	-25℃ to 60℃

Figure 2.2: Fra Producenten[14]

2.1 Analyse af opgave fra Risø

Opgaven deles op i 3 hoved dele. **Opsamling(Fase1)** ,**Transport (Fase2)** og **Lagering(Fase 3)**. Dette begrundes med brugerne kan trække data direkte fra Dataserveren til eksempelvis MySQL Workbench,Matlab eller Excel. Der er dermed ikke behov for nogle form for intraktion med dataopsamlingsudstyret eller det grafiske interface. Brugerne må så vidt muligt ikke bruge databasen der ligger på beaglebonen,da det kan påvirke de pågående målinger. Derfor skal data flyttes til en større server hvor tungbelastning og et eventuelt nedbrud vil ikke påvirke den videre dataopsamling. Det er oplystes at der er 12 sensorer + Tiden(UnixTime [5]) på hver lampe og alle 12 sensorer skal have en fejl-korrigering+ tiden.Det vil sige (12+2) * 2 = 28 felter med data. De 12 felter med måle data har en samplingstid på max. 1 pr. sekund for alle felter. Da flere af sensorne har forskellige parameter skal der korrigeres for dette. Dette gøres med Korrigeringsdata som sendes for alle måle punkterne 2 gange i døgnet. Det er et krav at udstyret skal være standart udstyr som kan købes hos Risøs levandøre af elektronik udstyr. Dette begrundes i at dataopsamlingsudstyret efterhånden vil gå til af at sidde ude for i alt slags vejr. Mere information om det Danske vejr og hvad man gør ved det i **Appendix J Dansk vejr**.

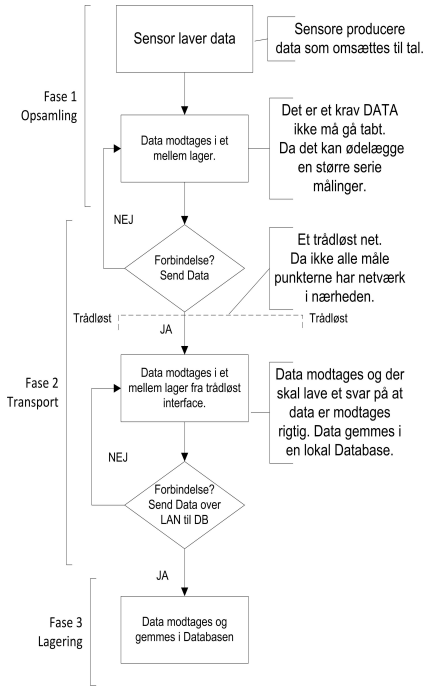


Figure 2.3: flowchart

2.1.0.1 Fase 4-Brugergrænsefladen

Den grafiske bruger interface (GUI) er mest for at gøre det mere brugervenligt og for at minimere teknikker timer ved ændring/fejlfinding at Dataopsamlingssystemet. Denne fase består af 2 dele. Nemlig på måleudstyret (hvis der er IP-trafik) og på serveren der skal modtage data og gør det tilgængeligt for dem måtte have et behov for disse data. På serveren vil der som udgangspunkt blive brugt standart styringsværktøjer,så som Webmin,phpMyAdmin og Munin. Dette begrundes i at det er AIT(Risø) der skal vedligeholde serveren. På Dataopsamlingsudstyret skal der udvikles en letvægts-hjemmeside med nogle grundlæggende muligheder for fejlfinding. Et godt eksemple er DD-WRT Web-interface[18],som man kan se på **Figure 2.5**.

Disse fejlfindingsværktøjer er:
Netværk Status (SQL forbindelse Risø-serveren)
Netlink? (IP/DNS /DHCP setting)
Hardware Status (Diskplads og belastning)
Modtager log. (modtages der data? Hvad modtages der helt som rå data?)
Database log. (hvad skrives der til Databasen)
Watchdog. (hvornår har watchdog'en fundet en fejl den måtte gøre noget ved?)

2.1.0.2 GFM-modelen

Denne model afspejler hvad der sker i problem området, alle hændelse der sker i problem området, vil forandringerne forplante sig ned igennem til modellen. **Grænseflade** er designet ud fra det skal være letvægts website,uden hensyn til rettigheder. **Funktionslaget** er selve motoren for hele projektet. Dette vil primært blive lavede i PHP. Her ligger også de forbindelser til databasen der skal udskifte hvis databasen skal udskiftes. Denne del er langt den største, da alle funktioner ligger i dette lag. I **modellen** hentes de data der skal behandles. I dette tilfælde lagers data i en database. Bemærk at den behøver ikke at fysisk ligge på samme maskine, da det er IP trafik der bruges.

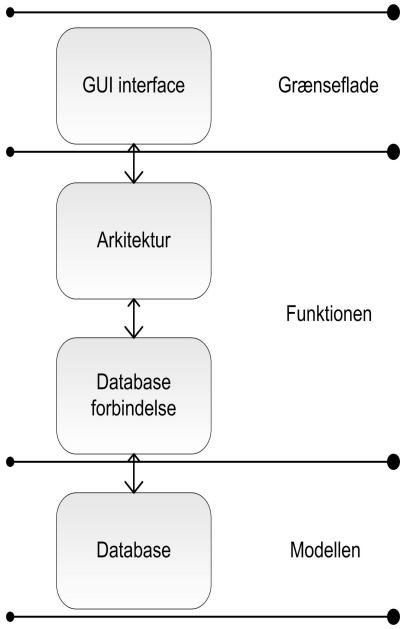


Figure 2.4: G-F-M model

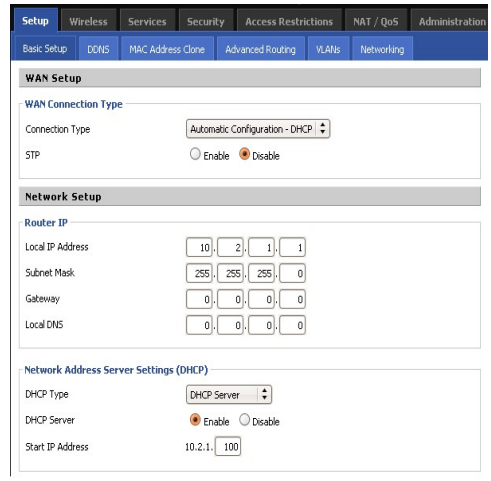


Figure 2.5: DD-wrt interface[18]

Chapter 3

Projektstrategi

I dette afsnit vil jeg i korte træk gennemgå alle punkter i min projektstrategi.
Fælder ved udvikling:

A. Den teknologiske udvikling løber fra projektet.
Inden for Embeded-hardware løber udviklingen hurtig. Denne udvikling er båret frem af marked for ARM-processore[2] som sidder i mobiltelefoner,computere og netværksudstyr. Moores lov hedder det, at stigningen i kompleksitet af mikroprocessorer og andre former for halvleder integrerede kredsløb, vil blive fordoblet hver 18. måned.En selvopfyldende profeti af Gordon Moore.[3]

B. Løsningen møder ikke brugeren på deres IT-niveau.
Udvikleren må første finde ud af hvem der skal bruge systemet til dagligt, er det en netværksansvarlig,en studerende eller er det forskere? Dette projekt har en meget bred brugerflade og der må derfor ikke stilles for store krav til brugerns viden om Linux systemer på embeded hardware.

C. Der går kludder i forskellig kode/dokument versioner
De fleste studerende kender det at man har kodet hele natten og er så træt man ikke lige lægger mærke til hvor man gemmer de seneste rettelser henne. Når projektet tages op senere, er det svært at finde ud af hvor er den senest version af koden og dokumenterne. Til at imødekomme det kunne anvendes CVS/Subversion eller forskellige online backuptjenster,mere om dette emne i **Afsnit 6 Risikostyring**

D. Løsningsmodellen ikke grundig nok.
I større projekterer hvor man er mange udviklere på forskelligt niveau er det vigtig de modeller man bruger er standart sprog alle kan forstå. I dette projekt skal beskrivelserne så vidt mulighed ramme UML standarten, så den/dem der evt. skal overtage projektet og videre udvikle det hurtig kan fortsætte udviklingen.

Metoder og processer.
Set ud fra de forskellige aspekter inden for projekt udvikling har jeg fundet på huske ordet **FRUGT**
Forundersøgelse
Rigtig Planlægning
Udviklingsstrategi
Kvalitetsstyring
Teststrategi

3.0.1 Tidsestimering

For at lave en tidsestimering, under **planlægning**, har jeg benyttet et GANTT-diagram, hvor slutdato for hver aktivitet definerer en endelige deadline for hver fase. Yderliggere bliver jeg nød til at indføre lukketider for at undgå for mange iterationer i samme fase. Det kan nemlig være fristende at fordybe sig i en bestemt problemstilling inden for en fase.**Udviklingsstrategi** for dette projekt ligger i at det har en kortere tidshorisont med en ufravigelig afleveringsfrist d.3 august , hvilket er grundlaget for overvejelserne omkring tidsestimeringer og planlægningsniveau. Da målet med projekt er veldefineret og udgangspunktet er at lave et Prof of Concept, har jeg benyttet den traditionelle 5-fase vandfaldsmodel med mulighed for fase iterationer, som er illustreret her under. Bemærk at der er 3 testfaser,dette begrundes i opgaven er delt op i 3 dele som alle hænger sammen. Hvilke betyder test fasen ændres hvis efterfølgende fase fejler.

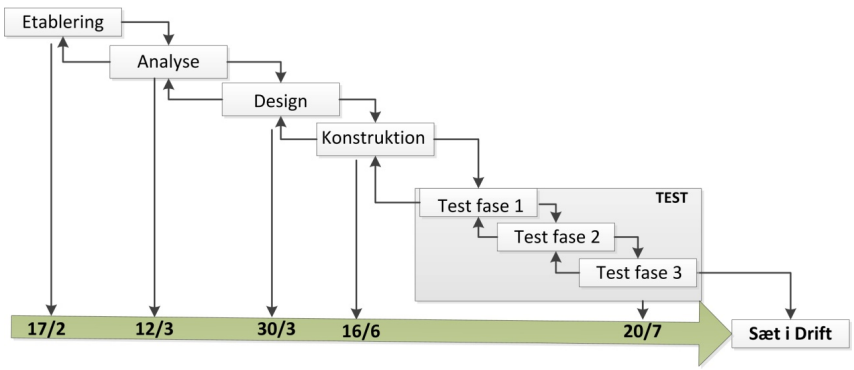


Figure 3.1: Tidsplan

Kvalitetsstyringen bygger udelukkende på mindre reviews med fotonik ansatte. Men da dette projekt er en prototype har jeg valgt at bruge FURPS+, som er omtalt i Applying UML and Patterns af Craig Larman [9]. Kravspecifikationen blev efterfølgende fremlagt og godkendt af de medarbejdere der er tilknyttede til projektet på et onsdagsmøde på Risø-Fotonik. **Teststrategien** kunne lægges op af brugen af IEEE829 standard for software test [7]. Den er dog fravalgt grundet erfaring i brugen og der er i stedet taget udgangspunkt i Poul Staal Vinje's Test af Software [10]

3.0.2 V-modellen

Idet der benyttes vandfaldsmodellen under planlægnings- og udviklingsstrategi, vil det være indlysende at bruge V-Modellen som teststrategi, i efterfølgende afsnit findes der en nærmere beskrivelse omkring brugen af denne model. Det er tidligere nævnt at projektet har en absolut deadline, derfor er det særligt vigtig at undgå fejl som forsager iterationer i V-modellen.Derfor vil jeg så vidt muligt prøve at holde mig til en 0-fejl-strategien,til flere af mine mindre tests.

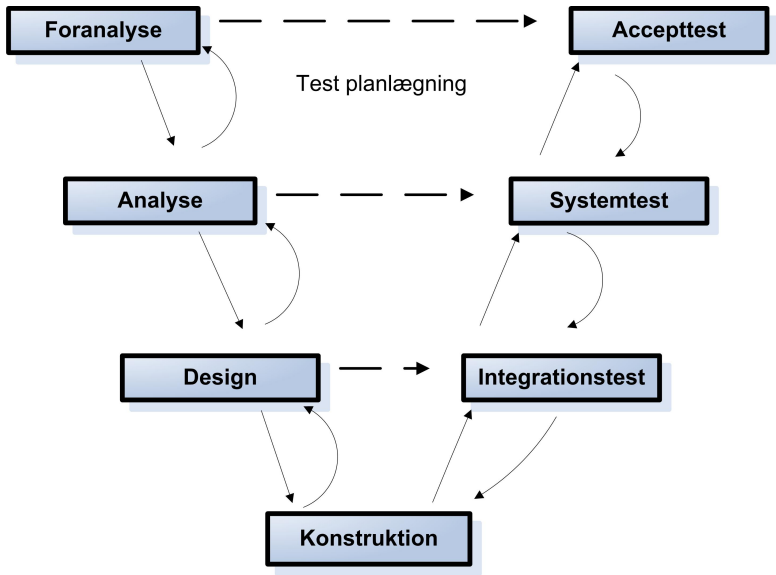


Figure 3.2: V-modellen [10]

Integrationstesten I denne test vil jeg vise at der kan sendes data over telit'en - hvor jeg sender data i det format der skal bruges direkte ind på End-point telit'ens comport. Bemærk at LabView har problemer med at hvis tiden nulstiller skriver den årstal 1904 og dermed kan UNIXtime[5] beregnes forkert.Mere I **Appendix E Labview test**,hvor man også kan se LabView programmet. **Bemærk!** I LabView tegner man sit program - man bruger ikke kildekode.

SystemTest Til denne test vil jeg bruge "modtage.pl" som gør det den åbner for porten og skriver det modtaget data til skærm. Bemærk! Man uden videre pipe output til en file hvis der ønskes at lave en testmåling over længere tid. Programmet og opsætning til denne test kan man læse om i **afsnit 9 Fra Sensor til Bruger**.

Accepttesten er hvad man kan kalde det ***big bang***,når alt kobles samme og man kan for første gang se hvor problemerne opstår. Ved accepttesten vil jeg vise at brugeren kan hente og bruge data direkte i MatLab uden man skal hente data ud til en komma file først,som har været procedure ved større projekter. Til denne test er udvilkede et lille MatLab script som henter data direkte fra Databasen på serveren og laver en graf ud af det. MatLab Accepttesten kan man læse i **afsnit 9 Fra Sensor til Bruger**.

Chapter 4

FURPS+

I dette afsnit vil jeg beskrive yderligere krav til systemet. Jeg har valgt at kategoriser efter modellen FURPS+ som er et element inden for Unified Process til at kategoriserer de funktionelle og ikke-funktionelle krav. FURPS+ Applying UML and Patterns af Craig Larman.[9]

4.0.3 Funktionalitet (Functionality)

Der er ikke yderligere funktionalitet end den, der er beskrevet i **Opgaven fra Risø** ,samt den funktionalitet der er beskrevet i **2.1.0.1 Fase 4-Brugergrænseflade**. Der vil være fokus på et letvægts webside med forskellige knapper og tilhørende funktioner.

4.0.4 Brugbarhed (Usability)

Systemets brugergrænseflade skal have et let anvendeligt og brugervenligt design,da ikke alle brugere har samme bruger niveau inden for Linux. Jeg har valgt at bruge hjemmeroutere som forbilled og en smule fra Rolf Molich's teori om brugervenlige EDB-systemer[17], så dialogen og design vil være let at anvende, lære og huske. For det meste er det første brugeren vil interesserer sig for ved et system,om det lever op til brugerens forventninger.

4.0.5 Pålidelighed (Reliability)

Pålideligheden i systemet er meget vigtig, da brugerne meget hurtig vil få følelsen af at dette system ikke virker 100 %. Dette vil i mange tilfælde betyde brugeren vælger ikke at bruge systemet og eventuelt går tilbage til det gamle system. Eller finder en måde uden om den tiltænkte måde at bruge systemet på. Adgangen til databasen vil være meget afhængig af pålideligheden af Risøs drift udstyr.

Da Serveren kommer til at være virtual på Risø's Vmware miljø, er det vigtig der forefindes en kopi som Risø's IT afdeling har ansvaret for.

4.0.6 Ydelse (Performance)

Da webinterfacet og data kun ligger på det intern netværk vil en tung belastning fra brugerens side ikke være noget problem. Men da serveren er en virtual maskine skal load og reload tiderne forkortes, dvs. at grafik som logoer og knapper ikke må fylde for meget, derfor burde de komprimeres. Brugerne må på intet tidspunkt trække data direkte fra dataopsamlingspunktet da det kan have katastrofale følger og kan medføre driftstop af målepunktet.

4.0.7 Vedligeholdelse (Supportability)

Den primære vedligeholdelse ligger på serversiden idet, at browseren forespørger på HTML-sider og de forskellige måledata værktøjer kun aflæser data i databasen. Derfor skal klienternes browser kun overholde W3C-specifikationerne og have mulighed for en MySQL forbindelse (ved brug af SQL og ODBC/JDBC).

4.0.8 Implementering (Implementation)

Web-delen skal kunne fungere på en almindelig arbejdsstation og smart phones,med de fleste standardinstallationer som har en Internet-forbindelse og er logget ind på DTU's domaine eller via en VPN uden for DTU.

4.0.9 Grænseflader (Interfaces)

Systemet skal kunne arbejde op imod en hvilke som helst MySQL database, samt nyere gængse Internet-browsere.Inden for grænseflade forventes det at det færdige produkt har haft disse 4 sikkerheds perspektiver for øje.

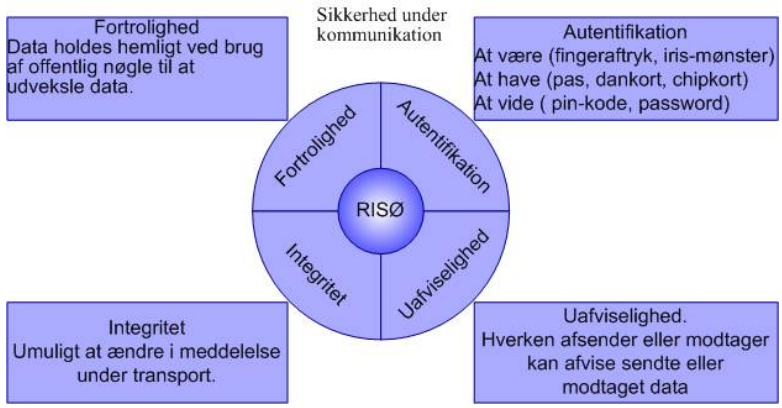


Figure 4.1: Sikkerhedsfirekløver

- Autentifikation klares ved brugernavn/adgangskode som AIT styre. ***NB! Dette brugernavn og adgangskode står i klar tekst i Matlab-scriptet og sendes i klartekst!(Se afsnit 9 Fra Sensor til Bruger.)***
- Fortrolighed. Som udgangspunkt er måledata ikke hemlige. Men der kan forkomme adgang til data der kan betragtes som fortrolige. Disse data kan krypteres efter behov.
- Integritet klares ved SSH til måleudstyret,SSL på websiden og SSH-tunnel ved Matlab/Labview scripts.
- Uafviselighed imødekommes ved at hver datapakke "stemples" med navn på måledata producent og en tidsstempling.Når pakken modtages tidsstemples den igen. Denne tidsstempling er UNIXTIME [5].

Chapter 5

Tekniske Rammer

Dette udviklings miljø er oprettet i henhold til DS484 10.1.4, hvor det klart er udtrykt at der skal adskilles mellem udvikling og drift miljøet som meget som muligt. Hvis muligt helst ikke på samme Net-segment. Derfor er det også vigtig en krydsforurening mellem de 2 miljøer under ingen omstændigheder må finde sted.

Ved det forstås at brugernavne og fortroligt data fra drift miljøet ikke må forefindes på udviklingsmiljøet. Begrundelsen ligger i at sikkerheden i et udviklingsmiljø ikke er på samme niveau som i drift miljøet.

Et godt eksempel er brugernavne og adgangskoder som i mange tilfælde er valgt ud fra de skal være lette, hurtigt at skrive og huske,da det skal skrives mange gange under test. Det kunne have katastrofale følger hvis sikkerheden indirekte kompromitteres i drift miljøet. Dette er også grunden til at bruger information, virtualle test servere og opsætning IKKE burde genbruges i drift miljøet.

Server & Database:Som udviklingsserveren er der valgt en Linux CentOS6.3 og OMAP-Ubuntu 11.x til beaglebone, begge med en Apache2.x og en MySQL 5.x som database. Begrundelsen for disse valg er at alt man skal bruge er gratis,opensource og projektet kan flyttes fra udviklingsmiljøet til en Linux drift server uden de store ændringer. Yderligere er der en god support fra et kæmpe community til både Linux,MySQL og Apache,da de alle 3 er opensource.

Test klient maskine:Test PC'en til de første tests er en Windows 7 med MySQL workbench og Matlab med jdbc (Database forbindelse til java). Dette er valgt ud fra standart arbejdsmaskine på Risø og ud fra at slutbrugerne skal bruge databasen til at hente måledata med MatLab/Labview,som skal bruges i forskellige projekter og artikler. På **Figure 3.2: V-modellen** kan man se hvorfor MatLab-scriptet fra **Fra Sensor til Bruger** passer ind i testen.

Flytbarhed:MySQL understøtter SQL standarten og derfor kan man eksportere tabeller til kommaseparerede filer eller som SQL sætninger, som derefter nemt kan importeres til den nye DBMS. For at flytte projektet til eksempelvis en Microsoft SQL server vil det kræve ODBC driverne installeres og kaldet ændres en smule i PERL-koden,som kan læse i fuld længde i **Appendix H ModtageData Koden(PERL)**.Som aktiv server script er valgt PHP og den bagvede liggende måledata genkendelse er skrevet i PERL [5].

Udviklingsplatform:Der bruges Ubuntu med Nano/Emacs og Windows7 med notepad++.Til at oploade koden til beaglebone og serveren bruges WinSCP. MySQL tilbyder værktøjer som eksempelvis MySQL Migration toolkit og MySQL Query Browser til at hjælpe med flytning og database opbygning og for at få databasens kald på plads bruges PhpMyAdmin.

Styreform:Der bruges milesten for at drive projektet frem, dog med mulighed for at fravige nogen dage som så senere må indhentes. Det kan nævnes der kommer påske,pinse ferie og en Roskilde Festival under selve projektets forløb.

Hardware til PoC:Det hardware der skal udvilkes til er et BeagleBone board. For at kunne starte BeagleBone boardet op skal der bruges et uSD-kort(samme slags som der sidder i eksemplevis telefoner).Der forfindes mere om installation i **Appendix D Beagle Bone Install**.
Til at sende data trådløst er indkøbt demokufferten Telit NE50 fra producenten af samme navn(**Figure 5.1: Telit NE50**).Yderligere er indkøbt et udviklingskit til AVR processore,det hedder et STK600. Da planen er at sætte en AVR til at styre dataflowet i forbindelse med hver telit.

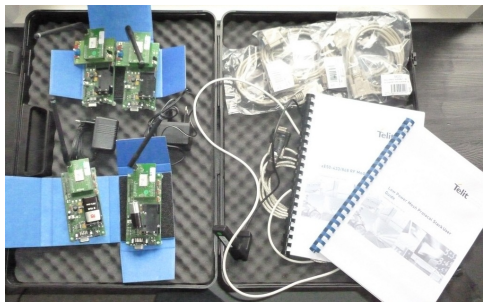


Figure 5.1: Telit NE50

Chapter 6

Risikostyring

I et større kandidat projekt er det en god ide at tage stilling til ting som kan hindre projektet i at nå sit mål, nemlig at blive færdigt. Man kan dele det op i 2 hovedgrupper, nemlig hindring i den fysiske verden. Eksempelvis sygdom, dødsfald, ulykker og trafikale hindringer for at møde op på Risø og lave projekt. Sygdom er der nok ikke så meget at gøre ved, ud over at betragte weekenden som arbejdsdag. For at i mødekomme hindringer i den fysiske verden har jeg valgt altid at have min 16Gb USB nøgle på mig, med hele mit projekt. Da hele projeket fylder omkring 2Gb som en Zip file, giver en hurtig hovedregning at jeg kan have det 8 gange. Hvilke giver mig mulighed for at jeg kan gå 8 arbejdsdage tilbage på min USBnøgle. Den anden del er de tekniske hindringer, så som nedbrud, hardware fejl,tyveri, virus/orm, hacker angreb eller software fejl. Der har jeg søgt vejledning i DS484 afsnit 10.5 (senere ISO27001) om sikkerhedskopiering hvor en af de ting der lægges vægt på er at sikkerhedskopierne opbevares adskilt og der skal foreligge en gendannelses procedure som man burde afprøve med jævne mellemrum. Yderligere kommer der løbende backup fra begge mine arbejdscomputere på UbuntuOne(i forbindelse med en Gmail konto)og på SkyDrive cloud drev(i forbindelse med en OutLook.com konto).

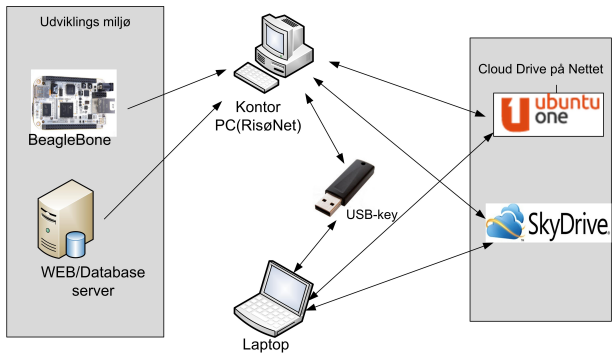


Figure 6.1: Backup setup

6.0.10 Test af Sikkerhedskopi

Sikkerhedskopien være testede for anvendelighed, når der foretages en fuld sikkerhedskopiering. Der er planlagt at der skal produceres en backup af opsætnings filerne fra beaglebone og den virtual server. Dette er for at kunne imødekomme en evt. re-installering af drift udstyret uden alt for meget nede tid. Backuppen fra Udviklingsmiljøet skal senere bruges til at lave en install file som skal bruges i AIT's vmware miljø.

6.0.11 Sikkerhed i Drift

Så længe dataopsamlingsudstyret og serveren står på Risø's område og sidder på Risø's netværk er de beskyttede mod trusler fra Internettet med den Cisco-Pix firewall AIT har og overvåget af AIT's Nagios og IDS-system. Hvis Database serveren og opsamlingsudstyret skal uden for Risø's netværk burde man overveje noget IP-filtering(firewall) og en opdateringsprocedure for alt software. De fleste operativsystemer kan sættes til at hente alle opdateringeringer fra Internettet automatisk. Denne fremgangsmåde kan dog give forskellige problemer,så som nyere versioner og genstart af systemet.

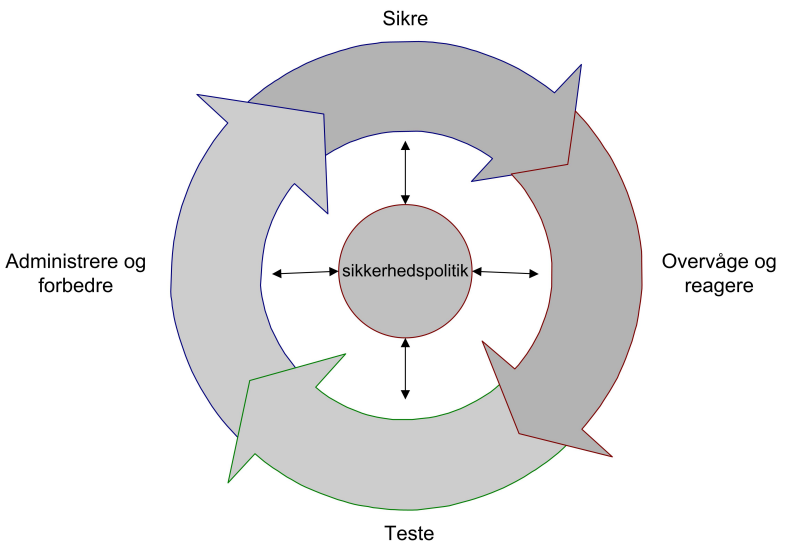


Figure 6.2: Sikkerhedscirkel [11]

Chapter 7

Forbilleder

Et af de forbilleder jeg har kiggede på er IceGoat [6],som er nogle autonome dataopsamlingsenheder et par steder nord for grønland. Grunden til denne er et godt forbilled er den kører Linux, den står et meget utilgængligt sted,selvforsynende fra solceller og påvirkede af vejret hele tiden. Da boey-icegoat står på i et meget tyndt befolket område skal alt over en satellittelefon. Herunder styring og data. Det betyder brugerne skal med simple kommandoer kunne genstarte,geninstallere og fejlfinde selv over en 9600 baud satellite linie. Dog skal udstyret så vidt mulig selv kunne finde en løsning på problemer der måtte opstå.

Figure 7.1: Boey-ICEgoat

En anden af de mest kendte dataopsamlings enheder (på et utilgængligt sted) der findes er Mars Rover[4].Men grunden til Mars Rover ikke er et godt forbilled er den har flere momenter,så som bevægelse og meget lange svartider. Men det kan man kan lære fra Mars Rover er at da den i de fleste tilfælde selv måtte finde en løsning på forskellige udfordringer den måtte rende ind i. Da den rendte ind i problemer den ikke selv kunne løse måtte Mars Rover sende en fejlmeddelse til NASA. Der findes en E-mail fra teknikkerne omkring problemet på Mars [4] og hvordan de løste problemet.

7.0.12 Hvad kan bruges?

Hvis man skal bruge noget fra disse forbilleder er det at vejret og det miljø udstyret skal stå i har stor indflydelse på elektronikken og de forskellige strøm producerende tiltag der er gjort. Herunder sne/frost på solcellen,vindmøllen og frost/vand i strøm-lageret. Det kan man løse på forskellige måder,en af måderne er varmetråde som man kender det fra bagruden i bilen.

Problemet ved denne metode er det bruger energi(strøm) som udstyret skulle bruge på at producere data og sende det hjem. Men holder systemet ikke solcellerne og vindmøllen is/frost fri vil udstyret langsomt gå i dvale til det bliver forår og tøvejr. Dette vil dog ikke være et problem så længe udstyret står på Risø's område da der vil være folk omkring udstyret næsten dagligt. Normalvis skal man have meget fokus på resourcer når man har med selvforsynende enheder at gøre. Specialt I dette tilfælde da en af målepunkterne er strøm produktion/-forbrug, derfor må måleudstyret ikke belaste eller bruge strøm fra det den måler på. Da det kan betyde målingerne bliver misvisende,med mindre man modregne den brugte energi eller gøre opmærksom på det i de målte data. Hvilke kan være meget svært da det kan svinge meget. Det er heldigvis sådan at der er strøm i en stikdåse i teststand 3. Det har vi valgt at benyttes os af.

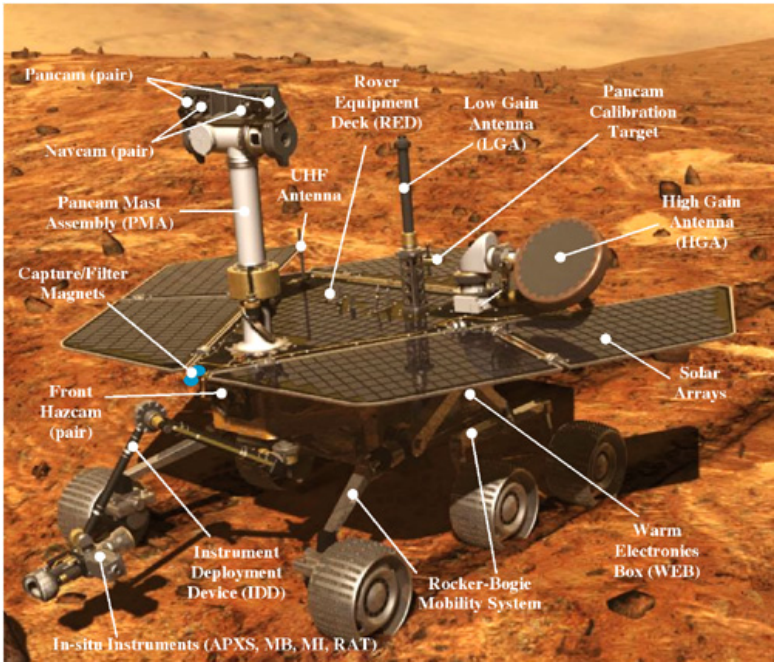


Figure 7.2: Mars Rover [4]

Chapter 8

Valg af Teknik

Figure 8.1: TestStand3 på Risø

8.0.13 Test område

Men hvilke hardware skal man indkøbe til dette projekt og hvilke overvejelser skal der gøres omkring indkøb og udskiftbart det forskellige hardware er.Den første opgave er at få måledata til Risø's netværk og videre til et sted hvor data kan bruges i forbindelse med forskellige forskningsresultater. Da VEA har erfarring med at deres kabler blive ødelagt at grasslåmaskine og aktivitet med tungt matriale i test område har de valgt at sætte alle deres kabler på 1 meter høje jordspyd. Derfor er der valgt at måledata fra hver måledata punkt skal sendes trådløst til opsamlingpunktet der videresender det til en database på Risøs driftsystem. For at benytte muligheden for trådløs datatransmission skal man være bekendt med ISM-bånd(jf.**bekendtgørelse BEK nr 458 af 23/05/2012**) VEA-Risø oplyser at de benytter område 433-434 Mhz til noget trådløst vejrstation der sidder på den ene vindmølle i test område. Da det er trådløst kræver det der ikke er for meget elektronisk støj,som der eksempelvis vil være i en storby. Da ingen af slutbrugerne har tilladelse til et lukkede

område,skal der bruges et åben frekevnsområde(ISM-båndet). Der er forslået 434Mhz,868 Mhz eller 2.4Ghz,som også bruges til eksempelvis vejrstation,babyalarmer, stegetermometer, termostater og meget mere hjemme elektronik. Mere om det i ***Appendix B Frekvens regler i Danmark***. En måling i testområdet med en HTC-X telefon indikere der ikke er noget trådløst netværk eller større bluetooth trafik. Hvilke betyder 2.4 Ghz vil være muligt som datatransport medie. Der kunne man vælge selv at sætte en WIFI-router op eller forespørg AIT(Risø's IT afdeling) om at få trådløst netværk i område. En anden mulighed kunne være at se på ZigBee/ZigBit som findes fra forskellige leverandøre. ZigBee teknologien tilbyder MESHnetværk og en rigtig god sikkerhed med fuld kryptering(AES) og med fuld TCP/IP ud til hver opsamlingspunkt.

8.0.14 Det økonimisk perspektiv

Et dataopsamlingsystem er et system der vil kører over længere tid og med en større mængde data strømmende igennem. I flere forsøgsopstillinger på Risø har man sat en standart PC eller en laptop til at samle data op.Når man spørg dem der sætter det op få man gerne svaret: "der er massere strøm i kontakten" og "en PC er nemmere at sætte op" **Bemærk!**:Der er valgt at se bort fra prisen på software,herunder en standart installation med Windows 7. Da det ellers vil give en betydlig højere pris på en standart PC.
Derfor er sprøgsmålet om der er noget at spare på strøm og indkøb?

8.0.14.1 Strømforbrug

En standart PC bruger omkring 300Watt , en Laptop bruger ca. 30 Watt og Et mini-board som beaglebone eller RaspberryPI bruger omkring 2 watt.
En beregning viser at:
Et nyere PC-system med 17' skærm bruger ca 300 W =0,3 kilowatt.
Strømforbrug per døgn = (0,3 kilowatt x 24 hours = 7,2 kWH
Strømforbrug per uge = 7 x 7,2 kwH = 50,4 kWH
Strømforbrug per år = 365 x 7,2 KWh = 2628 KwH
Strømregningen per år = 2628 kWh x 1,79 kr = 4704,12 kr
Et miniboard Ved et døgn : 002 watt x24 = 0048 0,048 Kwatt. Pr døgn
Strømforbrug per år = 365 x 0,048 KWh = 17,52 KwH
Strømregningen per år = 17,52 kWh x 1,79 kr = 31,36 kr

8.0.14.2 Hardware indkøb

Set i forhold til et økonimisk perspektiv,skalerbarhed og i forhold til hvad leverandøren kan skaffe til Risø (Jf. SKI aftale) er valgt 2 forskellige mini computere som begge kører Linux. Den lidt dyre Beaglebone til en vejl. Pris på 89 US$ og den lidt billigere Raspberry PI til en vejl. Pris på 35 US$
Prisen for de forskellige løsninger i US$:

Antal måle punkter	Smart Lamp(BB)	L.smart Lamps	Smart Rasp. PI	L.smart Rasp. PI	Ny AVR Rasp. PI
1	89	89,00	35,00	35,00	35,00
2	178	114,00	70,00	85,00	65,00
3	267	139,00	105,00	110,00	80,00
4	356	164,00	140,00	135,00	95,00
5	445	189,00	175,00	160,00	110,00
6	534	214,00	210,00	185,00	125,00
7	623	239,00	245,00	210,00	140,00
8	712	264,00	280,00	235,00	155,00
9	801	289,00	315,00	260,00	170,00
10	890	314,00	350,00	285,00	185,00

Figure 8.2: Priser vs. Antal

Bemærk: Disse tal kan svinge i forhold til dagspriserne på US-dolleren og om hvor store mængder og typen af data man ønsker at samle op.

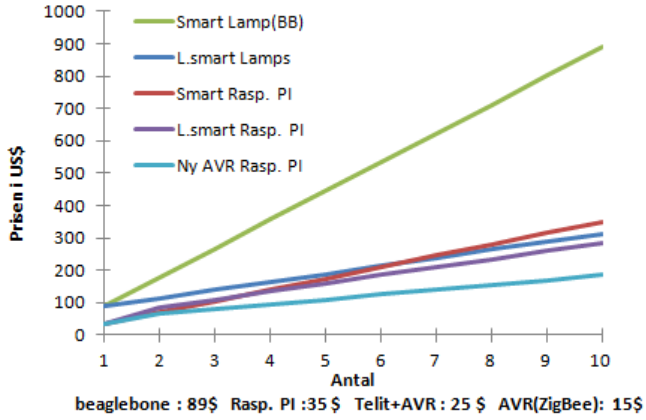


Figure 8.3: Priser graf

Smart Lamp(BB): Smart Lamps vil blive gennemgået i **afsnit "8.0.15 Løsnings modeller - model 2"**

L.Smart Lamps:Less Smart Lamps vil blive gennemgået i **afsnit "8.0.15 Løsnings modeller - model 1"**.
NB! Dette er den løsning Risø-fotonik har valgt at bruge.

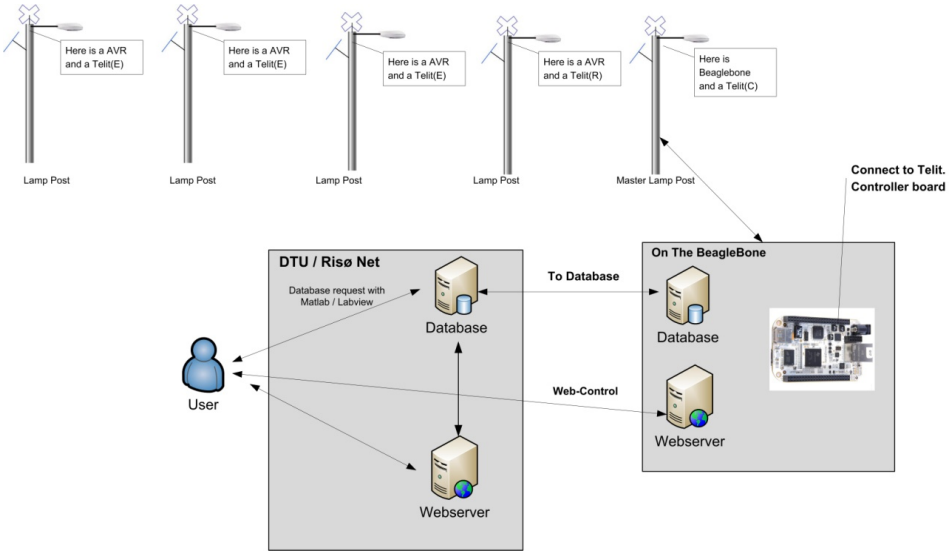
Smart Rasp.PI: Som Smart Lamps(BB) bare med Raspberry PI(Bilag 1) i stedet for en Beaglebone. Begrundelsen ligger i at Raspberry PI er meget billigere og har samme funktioner. Skal der en monitor på er det endnu en grund til at bruge Raspberry PI(**Bilag A Raspberry PI**) som har den noget bedre grafik CPU ARM1176 [13] og HDMI/composit udgang.

L.Smart Rasp. PI:Det er som forslag 1 –hvor Beaglebone er udskiftede med en Raspberry PI(**Bilag A Raspberry PI**).

Ny AVR Rasp. PI:Er beregnet ud fra ideen om at man dropper Telit'en og bruger en indbyggede ZigBee(IEEE 802.2.15) sender/modtager der er i eksempelvis en ATmega1280[16]I stedet for AVR640[15] som er den CPU der sidder på "L.Smart Lamps" (til at holde styr på sensores) til en vejl.pris på 15$.

8.0.15 Løsnings modeller-Model 1

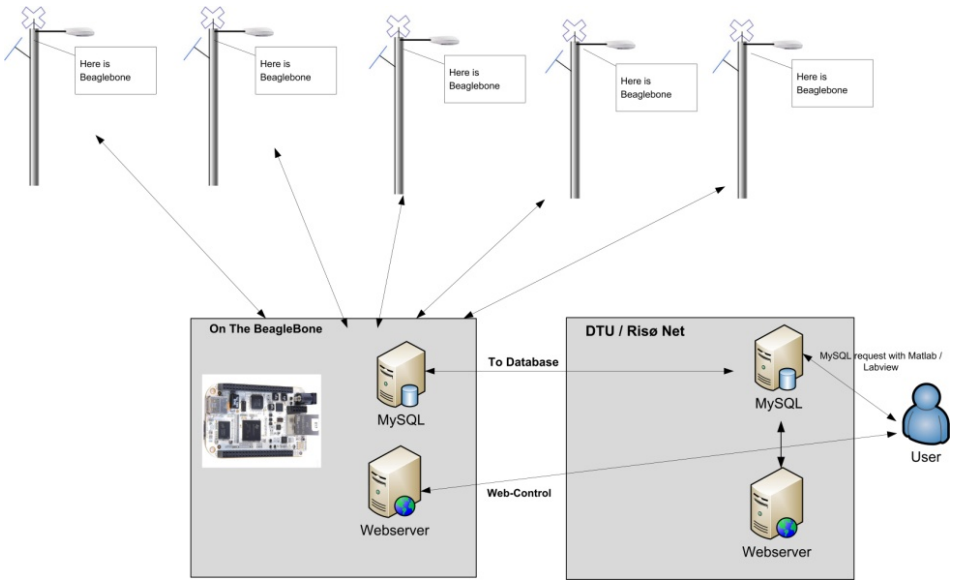
Mindre smarte Lamper(L.Smart Lamps) Grunden til den kaldes den "mindre smarte" er at CPU kræften i hver lampe er på kun ca. 20 Mhz og ikke noget direkte netværk(forstås som OSI'modellen). Hver lampe har en embeded AVR640 microprocessor for at styre dataopsamlingen og den bor sammen med en Telit Endpoint som er dens kommunikation med omverden.
Fordele er at: Da Telit nettet er et MESHnet(se afsnit om telit) er skalerbarheden meget høj,man kan uden videre sætte op til 12 sensore hver node uden at tag hensyn til IP-Adresser eller en fuld version af Linux der skal have TCP/IP for at kunne installere pakker og opdateringer.
Ulemperne er at: Da det er et radio netværk med en meget lav effekt er der kun en båndbredte på 19200 baud.(det betyder : 19200/8 = 2400 kbyte/s). De frekvens områder telit'en tilbyder (uden en licens) er begrænset til de åbne som alle andre også bruger.Kommandoerne til Telit'en er AT-hayes kommandoer,som man kender det fra modem'et. Det betyder at alt hvad man vil bruge telit'en til skal den have besked på som AT-kommandoer. Der er IKKE IP(Internet Protocol) hele vejen,men Telit'firmaets egen lukkede protocol som dog har et API man kan bruge.



8.0.16 Løsnings modeller-Model 2

Smarte Lamper(Smart Lamps) Fordele er at: Det er nemt at sætte op,da det ikke kræver hensyn og kommandoer til sende og modtage med modulerne. Hver lampe få en fuld computer,som kunne bruges til andre formål. Så som udbud af trådløst net og styring af forskellige ting,så som stepmotore så solcellerne kunne følge solen i sin bane. Man kan udnytte den høje båndbredte til eksemplevis live vejr advarsler/data eller måske et trafik kamera. Da der er mere processer kraft vil man kunne øge sikkerhedsniveauet betydligt,herunder brug af forskellige krypteringsformer. Da man har fuld IP-trafik hele vejen ud til lampen.

Ulemperne er at: Der skal en eller anden form for internet adgang fra hver lampe. Dette kunne dog løses med trådløsteknik så som 3G/4G eller med noget med en kortere rækkevide så som Bluetooth,Zigbee eller WIFI. Dette vil betyde den samlede mængde data fra hver lampe først samles på Database serveren. Prisen for hver enhed stiger betydligt. Mere om det **Det økonimisk perspektiv i afsnit 8.0.14**.



Chapter 9

Fra Sensor til Bruger

Den hardware og software der skal bruges til at få projektet til at fungere kan deles op i 4 dele,set ud fra de 3 faser som er nævnt tidligere i rapporten og den sidste del som er brugergrænsefladen. Det er værd at bemærke at Beaglebone også har et webinterface,men det bruges primært kun til debugging og overførelse af kommandoer.

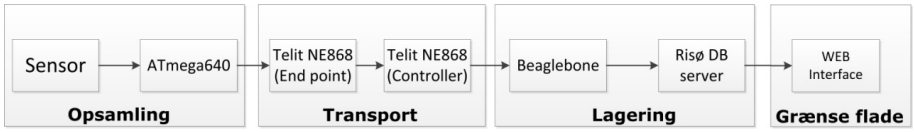
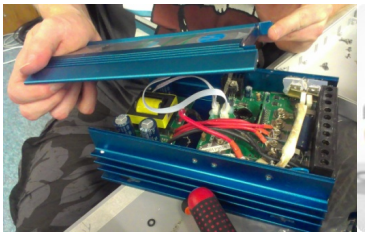


Figure 9.1: Datastream



(a) Styre Boks **(b)** Pyranometer **(c)** CO_2måler

Figure 9.2: Sensors**(Opsamling)**

Billed(a) Dette er styreboxen fra en CO_2Neutrale lampe(Nheohybrid 400). Bemærk de sorte klemmer der er for enden af kassen,der kommer alt information ud og ud af styrings boksen og det er derfor her der skal måles.
Billed(b) Er et pyranometer, som måler Watt pr.m^2.Data fra den model kan findes i **afsnit Matlab Accept Test**.
Billed(c) Som et fremtidig forslag til måling af luftkvalitet. I dette tilfælde er det CO_2 måleren fra fuehlersysteme.[12]
Figure 9.3: Telit-boardet med antenne. **Figure 9.4:** På selve sensorboardet sidder en ATmega640-16AU[15](med 12 forbindelser til sensore).

Figure 9.3: Sensor modulet

Figure 9.4: Sensor modulet(Underside)

9.0.17 Transport(Telit)

Her kan man se hvordan MESHnetværket er opbyggede , fra controlleren hvor data kommer ud af nettet til endepunkterne hvor data kommer ind fra sensors. Dette er et ultra low power mesh netværk som er designet som et batteridrevne sensor netværk. Ideen med MESHnet er det opbygges helt automatisk når man tænder for det. Det vil sige man skal storset bare(efter den er sat op til det) sætte Telit'en ud hvor den skal samle data op og den vil finde sin controller(på det netværk den hører til). Vi har valgt at have en controller og en router siddende hvor efter alle endpoints sidder. Endpoints er dem der sidder i hver lampe og det er her måledata kommer ind. Mere om Telit NE50 i **Bilag : Telit NE50 Datablad**

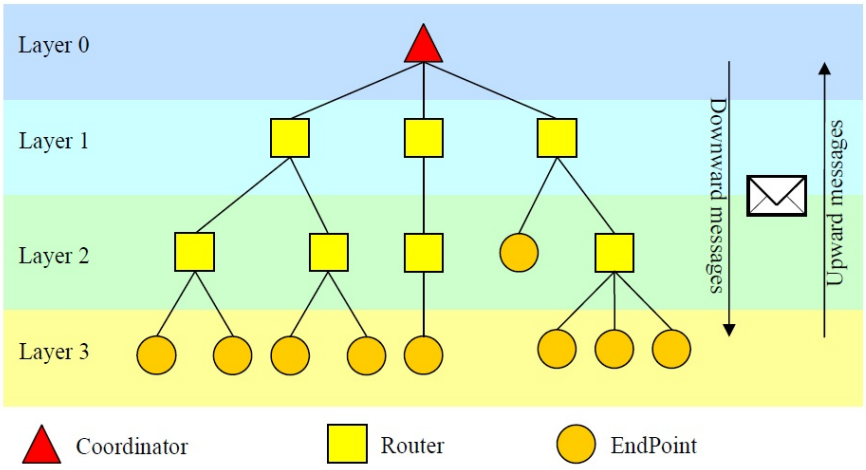


Figure 9.5: Opbygning ***Kilde :*** *http://www.telit.com*

9.0.18 Transport til Lagering(Hardwaren)

Ledningerne skal sættes sådan : *Bemærk at jeg bruger uart1 på pin9 – da man ellers skal lave speciale settings i default kernel under Ubuntu-Omap som jeg har valgt at bruge.*

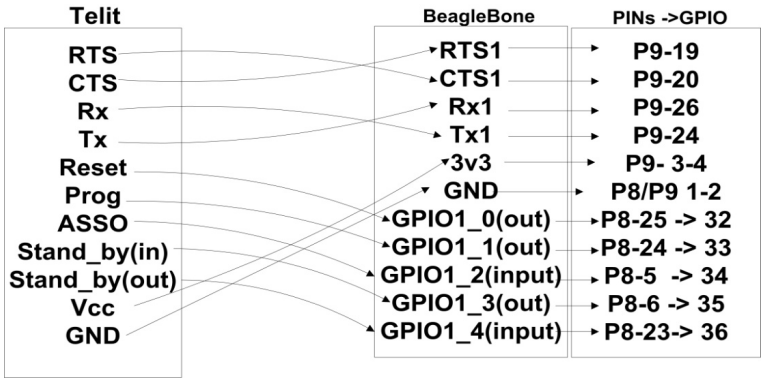


Figure 9.6: Telit til Beaglebone

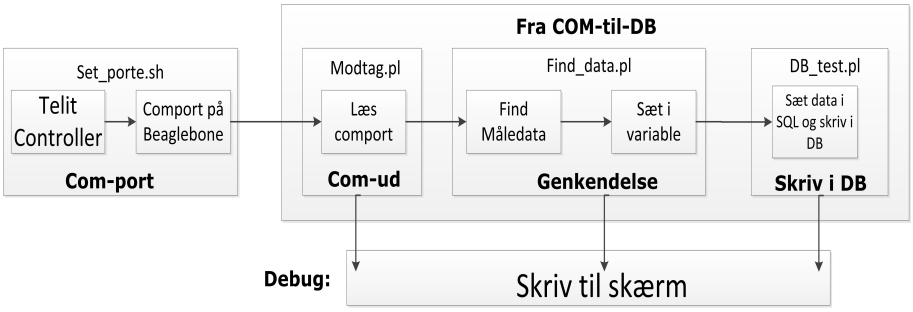


Figure 9.7: Debug og Design

Transport til Lagering(Script Design) Jeg har valgt at bygge det op i moduler og derefter sætte det sammen. Til dette er et script sprog som PERL [5] godt. Man kan dele det op i 3 dele,nemlig data skal ud af UART-porten,data skal genkendes og skrives i Databasen.

9.0.19 Beaglebone Scripts

Setup script skal sættes så beaglebone boardet starter scriptet ved genstart. Dette begrundes i at GPIO(se **bilag GPIO-Kernel**) mount points er virtuale og dermed bor i RAM'en og derfor skal sættes ved hver genstart.
Der skal man bruge Linux kommando'en update-rc.d.

For at ramme de rigtige GPIO'er skal man lave en lille beregning.
Eksempler: GPIO 0 _ 15 - Det beregnes sådan : 32*0+15 = 15
Jeg har valgt at bruge UART1. Derfor ser GPIO Beregning ser sådan ud:
32*1+0 =32. 32*1+1 =33. 32*1+2 =34. 32*1+3 =35. 32*1+4 =36.
Bash scriptet : set_porte.sh

```
1  #!/bin/sh  Port setup.
2  echo 7 > /sys/kernel/debug/omap_mux/gpmc_ad0
3  echo "32" > /sys/class/gpio/export
4  echo 7 > /sys/kernel/debug/omap_mux/gpmc_ad1
5  echo "33" > /sys/class/gpio/export
6  echo 7 > /sys/kernel/debug/omap_mux/gpmc_ad2
7  echo "34" > /sys/class/gpio/export
8  echo 7 > /sys/kernel/debug/omap_mux/gpmc_ad3
9  echo "35" > /sys/class/gpio/export
10 echo 7 > /sys/kernel/debug/omap_mux/gpmc_ad4
11 echo "36" > /sys/class/gpio/export
12 echo "high" > /sys/class/gpio/gpio32/direction
13 echo "low" > /sys/class/gpio/gpio32/direction
14 echo "low" > /sys/class/gpio/gpio33/direction
15 echo "low" > /sys/class/gpio/gpio34/direction
16 echo "in" > /sys/class/gpio/gpio34/direction
17 echo "low" > /sys/class/gpio/gpio35/direction
18 echo "low" > /sys/class/gpio/gpio36/direction
19 echo "in" > /sys/class/gpio/gpio36/direction
20 echo "high" > /sys/class/gpio/gpio32/direction
21 echo 20 > /sys/kernel/debug/omap_mux/uart0_rxd
22 echo 0 > /sys/kernel/debug/omap_mux/uart0_txd
23 echo 20 > /sys/kernel/debug/omap_mux/uart1_rxd
24 echo 0 > /sys/kernel/debug/omap_mux/uart1_txd
25 perl /home/ubuntu/serial_log.pl >> beaglebone.txt &
```

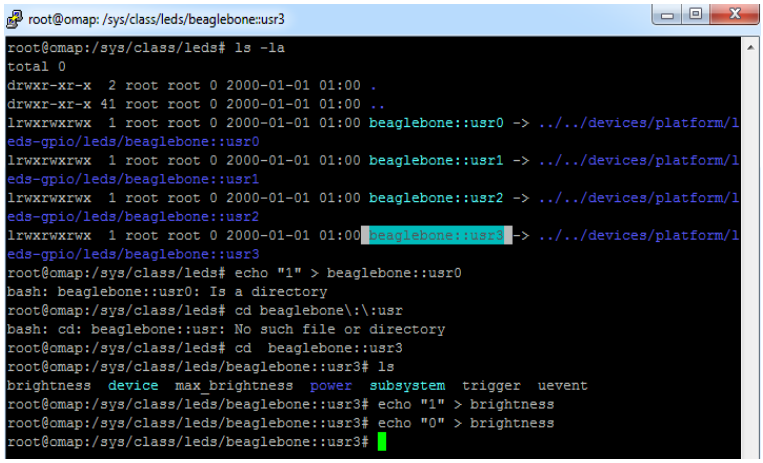


Figure 9.8: GPIO USB test

9.0.19.1 Volt meter Testen

Med et voltmeter kan man se der kommer 3.3volt på porten når man sætter porten høj. Dette bruges til at kunne styre de 3 primær switches der var på Telit demo-boardet. Ved at skrive til porten med echo og sætte den LOW eller HIGH kan man vippe med de forskellige switches(virtualt) der sidder på telit-controlleren via UART1 på Beaglebone.

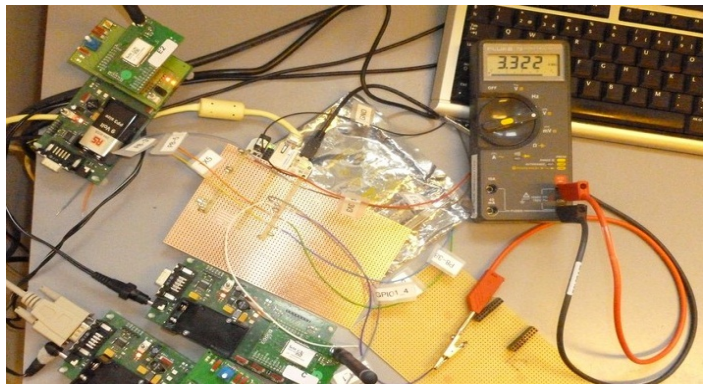


Figure 9.9: Voltmeter viser 3.3v

9.0.19.2 Test Data på comport.

Som skrevet i **3.0.2 V-modellen** vil en test vise om udstyret er sat rigtig op og kan modtage data fra Telit-controlleren. Ud fra det lille PERL[5] script **modtag.pl** kan jeg konstatere om der kommer data ind på UART1(comport1) på beaglebonen. Det fungere ved at UART1 porten (som I linux verden hedder /dev/ttyO1) Skrives ind i variablen $port_n og forskellige parameter sættes ud fra API'en til use Device::SerialPort. Modtag.pl scriptet er den første test kode for at se om der kommer noget ind på UART1. ***Perl Script : modtage.pl***

```
1  # search.cpan.org/~cook/Device-SerialPort/SerialPort.pm
2  use Device::SerialPort;
3  my $port_n ="/dev/ttyO1";
4  my $port_obj = new Device::SerialPort ($port_n) ||
5       die "Can't open $port_n $!";
6  #Serial port Parameters
7  my $buffer = 1024;
8  $baud_rate=19200;
9  $port_obj->databits(8);
10 $port_obj->handshake("none");
11 $port_obj->baudrate($baud_rate);
12 $port_obj->parity("none");
13 $port_obj->stopbits(1);
14 $port_obj->buffers($buffer,$buffer);
15 ################Vis Data#################
16 while(1){select(undef,undef,undef,0.80)
17 $data_read=$port_obj->read($buffer);
18 print "$data_read\n";}
```

9.0.20 Data Genkendelse

En Rå-datapakke fra UART1 kunne se sådan ud:

```
1  eg0 T=1113678328 A=41511 B=16516076 C=78655019
2  D=23 E=111 F=333 G=122 H=2222 I=222 J=222 K=331 L=344
```

Scriptsproget PERL[5] har rigtig gode muligheder for at kunne genkende data som kommer ind på forskellige måder og formattere det efter behov.
Database Formattet ser sådan ud :

Telitnr,Netnr,BBT,T,A,B,C,D,E,F,G,H,I,J,K
Telitnr = Node afsender (der kan være fra 0 til 99)
Netnr = Hvilke net noden tilhører (der kan være fra 0 til 99)
I dette tilfælde sættes den til 42 på beaglebone.
BBT = Tidsstempling for datamodtaget på beaglebone i UNIX TIME.[5]
T = Tidsstempling for data afsendt fra Telit-noden i UNIX TIME.[5]
*Da Teliten selv ikke har nogle logik til at aflæse tiden kommer denne tid fra AVR'en der sidder på Endepunkterne(se **Figure 9.3** og **Figure 9.4**)*
Dette er selve genkendelses delen. I dette script genkendes data
og skrives til skærmen med ***printf***. Dette er for at se om data kommer ind og hvad der i grunden kommer af data.

```
1  while(1){
2  select(undef,undef,undef,0.80);
3  $data_read=$port_obj->read($buffer);
4  ####################Telit styring#################
5  $Netnr=42;  $BBT = time;  $Telitnr=13;
6  ##########################################
7  if($data_read =~ m/e0f\s+T=(\d+)\s+A=(\d+)\
8  s+B=(\d+)\s+C=(\d+)\s+D=(\d+)\s+E=(\d+)\
9  s+F=(\d+)\s+G=(\d+)\s+H=(\d+)\s+I=(\d+)\
10 s+J=(\d+)\s+K=(\d+)\s+L=(\d+)/)
11 {printf("e0f:%s\n",join(',',
12 ($BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13)));}
```

9.0.21 Data Skriv

jeg har valgt ikke at gøre så meget ud af database designet, det er begrundet med at dette er et PoC hvor der kun gemmes data fra forskellige målepunkter,ingen bruger styring af interfacet eller opsætning af Beaglebone. SQL scriptet til opsætning kan læses i **Bilag DB server-CentOS**.

Bemærk: *Sættes beaglebone til at replikere til DB-serveren på Risøs netværk burde man overveje at kigge på komprimering med* ***MySQL-innoDB*** *til beaglebonen. I disse scripts er denne funktion er slået fra og data skrives direkte på DB-serveren,hvilke gør dataopsamlingssystemet er sårbart over for udfald på forbindelsen til DB-serveren.*

Der vil jeg gøre brug af DBI og DBD:mysql Perl's API. I dette tilfælde er værdierne ikke aflæst men "hardkodet" ind i scriptet for at se om beaglebone kan skrive i databasen, uden at der skal tages stilling til om data kommer rigtig ind fra UART-porten.
Perl Script : databasetest.pl

```
1  # API http://search.cpan.org/modlist/Database_Interfaces
2  use DBI;
3  use DBD::mysql;
4  $database = "beaglebone";
5  $host = "192.168.1.6";
6  $user = "fotonik";
7  $pw = "fotonik";
8  # DATA SOURCE NAME
9  $dsn = "dbi:mysql:$database:$host:3306";
10 $connect = DBI->connect($dsn, $user, $pw);
11 ####TEST DATA -START
12 $Telitnr=01; #String ?$Netnr=42;
13 $BBT = time; $T=3420093520; $A=408; $B=386;
14 $C=763; $D=453; $E=632; $F=207; $G=137;$H=967;
15 $I=106; $J=725; $K=230;
16 ####TEST DATA -END
17 $query = "INSERT INTO test
18 (Telitnr,Netnr,BBT,T,A,B,C,D,E,F,G,H,I,J,K)
19 VALUES (Telitnr,$Netnr,$BBT,$T,$A,$B,$C,
20 $D,$E,$F,$G,$H,$I,$J,$K)";
21 $query_handle = $connect->prepare($query);
22 $query_handle->execute();
```

9.0.22 Tiden på Beaglebone

Jeg har undersøgt hvor god Beaglebone er til at holde tiden. Denne undersøgelse kunne løses på flere måder. Man kunne lade Munin (RRD) gøre det og lave nogle fine grafer over det. Jeg har valgt at lave et lille script og putte det i crontab.(under daily). Det betyder hver dag kl.6.25 læser den scriptet og spørg Risøs NTP om hvad klokken er. Hvorefter den sætter klokken den fik fra NTP og skriver forskellen(Offset) i en logfile.
ntpdate 130.226.56.67
Resultat
:1,519109+1,543915+1,540759+1,507843+1,529694+1,540735+1,515301 / 7
1,5281 sek. Taber beaglebone pr. døgn. Målt over 7 dage.

9.0.23 Telit Send

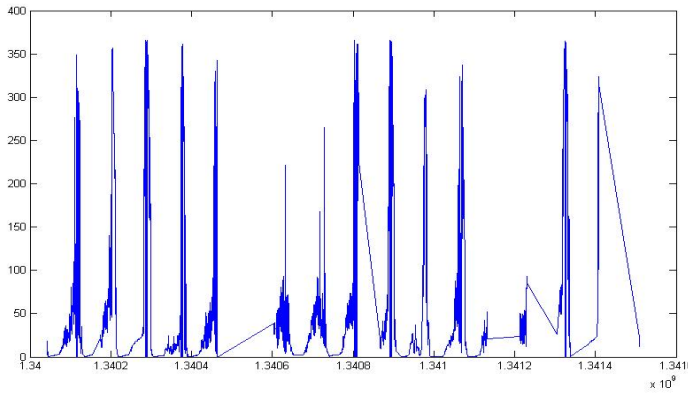
Under afsnit **Transport(Telit)** kan man se hvordan Telit netværket er byggede op omkring en MESH protokol.For at Telit netværket kender klokken sendes UNIXTIME ud. Der kunne man vælge at bruge en tidsenhed,så som DCF77 eller GPS. Der skal man være opmærksom på at for at kunne sende kommandoer skal tiden pakkes ind i start DATA-send og en slut DATA-send.
Hvis jeg vil bruge Hayes kommandoen: ATS300?
Den gamle måde serial frame:
|0x6D|0x45|0x31|0x41|0x54|0x53|0x33|0x30|0x30|0x3F|0x0D|
Den nye måde serial frame:
|0xAB|0x6D|0x45|0x31|0x41|0x54|0x53|0x33|0x30|0x30|0x3F|0x0D|0xCD

Perl Script : Telittime.pl

```
1 my $command1 = (pack( "H8","AB650C00"));
2 my $command2 = (pack( "H4","6FCD"));
3 my @Func = ($command1,time,$command2);
4 my $command = join "", @Func;
5 my $time = (time);
6 my $command = ($command1+$time+$command2);
7 $port_obj->write($command);print "Send = $command\n";
8 print "time = $time\n";
```

9.0.24 Matlab Accept Test

Resultat fra scriptet som bruges ved final test. For at scriptet kan kører skal man bruge mysql-connector-java.jar som skal ligge i samme mappe.



MatLab scriptet : testscripfranet.m

```
1  javaaddpath('mysql-connector-java-5.1.20-bin.jar');
2  %# connection parameteres
3  host = '127.0.0.1';user = 'fotonik';
4  password = 'fotonik'; dbName = 'beaglebone';
5  %# JDBC parameters
6  jdbcString = sprintf('jdbc:mysql://%s/%s', host, dbName);
7  jdbcDriver = 'com.mysql.jdbc.Driver';
8  conn = database
9  (dbName, user , password, jdbcDriver, jdbcString);
10 if isconnection(conn) % successfully connected
11     SQLquery = ['SELECT MIN( `BBT` ) AS ''BB_time''
12 ,AVG( `A` ) AS ''average_A'' FROM beaglebone.test where
13 BBT>UNIX_TIMESTAMP(''2012-06-18 20:00:00'') and
14 telitnr=16 GROUP BY round( `BBT` ) DIV 60'];
15      testdata = fetch(conn, SQLquery);
16 end close(conn);
17 for i = 1 : length(testdata)
18      data(i,1) = testdata{i,1};
19      if testdata{i,2} > 500000
20   data(i,2) = 0; else data(i,2) = testdata{i,2};
21   end end plot(data(:,1),data(:,2)*0.01/13.674)
```

Chapter 10

Styring via Web

For at gøre det bruger venligt er der udviklede et letsvægts web interface som primært bruges til at fejlsøge på beaglebone. Som nævnt i **2.1.0.1 Fase 4-Brugergrænseflade** har det været ønskligt fra Fotoniks ansattes side at der var et webinterface som man kunne lave de grundlæggende fejlsøgning med. Som forbilled er brugt **Figure 2.5: DD-wrt interface**. Webinterfacet giver mulighed for at : læse logs(gamle og de nye),teste forbindelser og genstarte interfaces eller hele modulet. (**Bemærk knapperne!**)

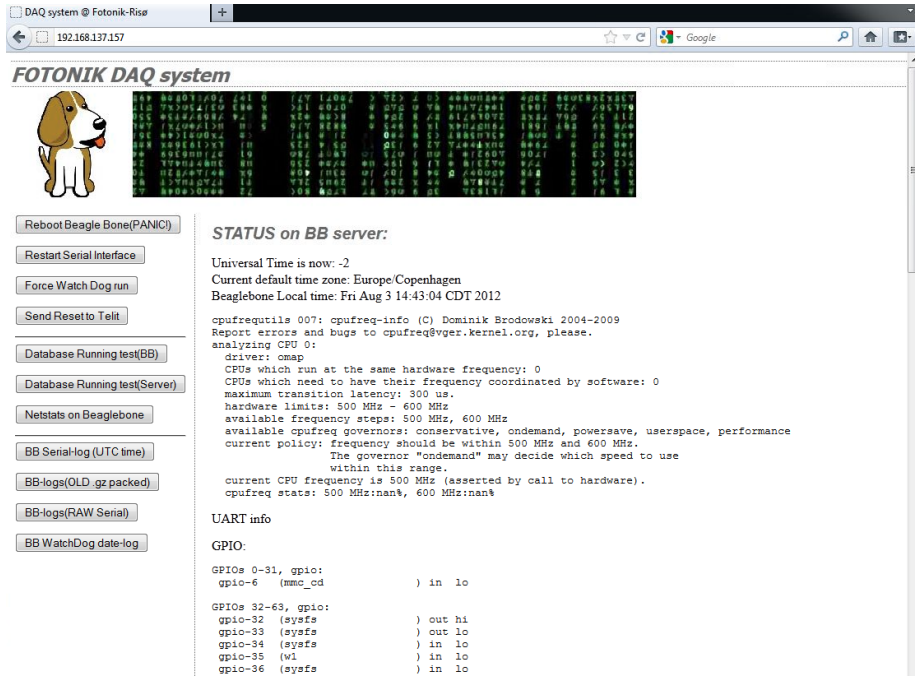


Figure 10.1: WebInterfacet

Dette er et udklip af index.php scriptet. Som giver brugeren forskellige status på beaglebonen så som tid/dato,CPU,Diskplads og hvad der kører lige nu.

index.php(CUT-version!)

```
1  <?php $output = shell_exec('sudo date');
2  echo "Beaglebone Local time: ";
3  echo "$output"; ?>
4  <?php  $output = shell_exec('sudo cpufreq-info');
5  echo "<pre>$output</pre>";
6  echo "UART info";
7  $output = shell_exec('sudo cat
8  /sys/kernel/debug/omap_mux/uart*');
9  echo "<pre>$output</pre>";
10 echo "GPIO:"; $output = shell_exec
11 ('sudo cat /sys/kernel/debug/gpio');
12 echo "<pre>$output</pre>";
13 echo "Space on Disk:"; $output =
14  shell_exec('sudo df');
15 echo "<pre>$output</pre>";
16 echo "What services is runnning?";
17 $output = shell_exec('sudo netstat -tna');
18 echo "<pre>$output</pre>";
19 echo "Whats running?(PS)";
20 $output = shell_exec('sudo ps -aux');
21 echo "<pre>$output</pre>";
22 ?>
```

De scripts som knapperne kalder:

resettelit.php

```
1  <?php
2  $output = shell_exec('echo "high" >
3  /sys/class/gpio/gpio32/direction ');
4  $output = shell_exec('echo "low" >
5  /sys/class/gpio/gpio32/direction');
6  $output = shell_exec('echo "low" >
7  /sys/class/gpio/gpio33/direction');
8  $output = shell_exec('echo "low" >
9  /sys/class/gpio/gpio34/direction');
10 $output = shell_exec('echo "in" >
11 /sys/class/gpio/gpio34/direction');
12 $output = shell_exec('echo "low" >
13 /sys/class/gpio/gpio35/direction ');
```

```
14 $output = shell_exec('echo "low" >
15 /sys/class/gpio/gpio36/direction');
16 $output = shell_exec('echo "in" >
17 /sys/class/gpio/gpio36/direction ');
18 $output = shell_exec('echo "high" >
19  /sys/class/gpio/gpio32/direction');
20 $output = shell_exec('echo 20 >
21 /sys/kernel/debug/omap_mux/uart0_rxd');
22 $output = shell_exec('echo 0 >
23 /sys/kernel/debug/omap_mux/uart0_txd');
24 $output = shell_exec('echo 20 >
25  /sys/kernel/debug/omap_mux/uart1_rxd');
26 $output = shell_exec('echo 0 >
27 /sys/kernel/debug/omap_mux/uart1_txd');
28 echo "Reseting␣Telit␣on␣Beaglebone(Controller)!"; ?>
```

Gennemtvinger kørelse af Watchdog scriptet. ***watchdog.php***

```
1 <?php
2 echo "Force␣Running␣WatchDog....
3 (Try␣Serial␣Restart␣if␣nothing␣happen)";
4 $output = shell_exec
5 ('sudo /etc/cron.hourly/watchdog.sh');
6 echo "<pre>$output</pre>";
7 ?>
```

Giver InterNet Status set fra Beaglebone. ***test3.php***

```
1 Beaglebone Net Stat info:<br>
2 <br>
3 <?php
4 $output = shell_exec('sudo netstat -s');
5 echo "<pre>$output</pre>";
6 ?>
```

Giver Database Status set fra Beaglebone. ***test2.php***

```
1 VMware Server MySQL <br>
2 <br>
3 <?php
4     mysql_connect("192.168.1.4", "fotonik", "fotonik");
5     $array = explode("  ", mysql_stat());
6     foreach ($array as $value){
7         echo $value . "<br />";
8     }
9 ?>
```

Giver Database Status på Beaglebone. ***test2.php***

```
1 BeagleBone MySQL info:<br>
2 <br>
3 <?php
4     mysql_connect("localhost", "fotonik", "fotonik");
5     $array = explode("  ", mysql_stat());
6     foreach ($array as $value){
7         echo $value . "<br />";
8     }
9 ?>
```

restartserial.php

```
1 <?php
2 echo "Restart Serial Interface.. Back in 2 min...";
3 $output = shell_exec('sudo /etc/init.d/serialrestart.sh');
4 echo "<pre>$output</pre>";
5 ?>
```

Genstarter Beaglebone. ***reboot.php***

```
1 <?php
2 $output = shell_exec('sudo reboot');
3 echo "<pre>$output</pre>";
4 echo "REBOOTING!!!! 3 min. and we are back!"; ?>
```

10.0.25 Fejl tolerance

Jeg har udviklede mit eget lille watchdog script.Ideen er den sprøg om dataopsamlingscriptet kører ellers skal den starte det.Hvis den skal starte noget henter den tiden fra NTP'serveren og skriver den i lastdate.txt, hvor efter den kalder dataopsamlingsscriptet og piper debug data til logfilen beaglebone.txt.Man kunne vælge at den også skal holde øje med database forbindelsen,det er dog fravalgt på nuværende tidspunkt.

watchdog.sh

```
1 #!/bin/sh
2 app='perl /home/ubuntu/serial_log.pl'
3 if [ ! "$(pidof $app)" ]
4 then
5 /usr/sbin/ntpdate 130.226.56.67 >> /var/www/lastdate.txt &
6 perl /home/ubuntu/serial_log.pl >> /var/www/beaglebone.txt &
7 fi
```

Det er være at bemærke at Linux Kernel har en watchdog. Hvis man ønsker at benytte den skal man skrive til: /dev/watchdog hvert minut(default!). Hvis der ikke skrives genstartes operativsystemet. Mere om setup og information på `http://manpages.ubuntu.com/manpages/hardy/man8/watchdog.8.html`

Chapter 11

Fremtiden for projektet

Starter man med at kigge på **"mindre smart lampen"** vil en af mulighederne være at bruge samme model til dataopsamling ved mindre projekteter hvor det ikke kræves der sidder en fuld computer.

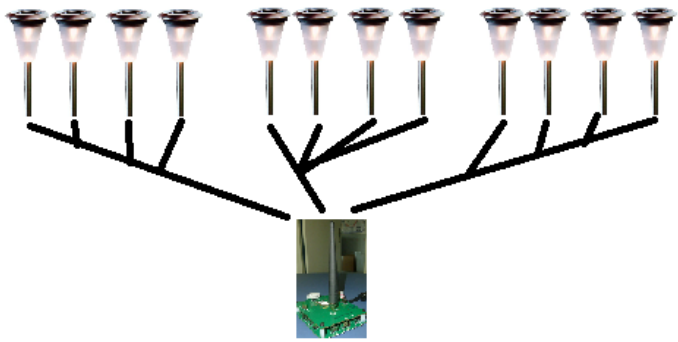


Figure 11.1: Måledata Grid

Hvis man skal uden for forsker verden kunne det være forskellige sportsgrene hvor live data er vigtige for beslutninger(motor-/sejl-sport) og tv seerne. På et større skib hvor forskellige tilbage meldinger om tingens tilstand kan gøre en forskel på beslutninger eller i en større virksomhed hvor man gerne vil have overvågning/alarmer omkring en produktions proces. Evt. i hjemmet til at overvåge huset med forskellige små sensore(Sensor grid), hvor man vil kunne styre og få svar på forskellige resource spørgsmål som eksemplevis varme/vand/strøm/Internet/TV(forbrug)/brand/gas eller tyveri alarmer på døre/vinduer. Da Telit controlleren sidder på et TCP/IP netværk vil det give mulighed for at styre det fra en computer/mobiltelefon på den anden side af jorden,hvis der er routening til/fra Internettet.

Ved **"Smart lampen"** vil en af mulighederne være at bruge samme model til steder hvor man har brug for fuld internet adgang hele vejen ud til lampen.Man kunne forestille sig i lande hvor gadelampen er udbredte,men de springer fiber kabling over og går direkte til den trådløse teknologi. Flere undersøgelser viser at man mange steder i Udviklingslande ikke har udbredt 3G/4G dækning.

Figure 11.2: MeshNet i Udviklingslande[19]

Det godt bud på hvordan smarte lamper kunne tænkes ind i gadebillede er fra "Unge vinder pris for halvgeniale internetlamper" artiklen Af Rasmus Palludan: ***Ved at putte små antenner ind i specialdesignede gadelamper, har de opfundet en potentiel løsning på et af telekommunikationssektorens nuværende problemer: Etablering af god radiodækning i tæt befolkede områder, hvor teleselskaberne i dag er i gang med en overgang til hurtige 4G-netværk, der kræver større båndbredde. Gadelamper er særligt smarte at bruge, fordi de hænger i en højde, der vil gøre dem optimale til at sende trådløse radiosignaler. [20]***

Figure 11.3: Internetlampe[20]

Chapter 12

Konklusion

Set ud fra Strømforbruget og dermed de resourcer der bruges på at samle data op,vil det være godt at skifte dataopsamlingsanlæg ud med et mere strømbesparende enheder. Set ud fra det **Det økonimisk perspektiv** vil "Ny AVR Rasp. PI"-løsningen være det bedste valg. Men den billigste løsning er det også den mest kompliceret løsningen der vil kræve mere tid at udvikle fra programmørens siden.

Ser man på det overordnede mål med projektet som var at undersøge om det var muligt at lave dataopsamling på en ny og smartere måde end den fotonik bruger på nuværende tidspunkt,nemlig en standart PC med et dataopsamlingkort til en frygtlig masse penge fra en leverandør.

Mit projekt vist hvilke fordele det har at bruge et mini board med en ARM processor på. Projektet har helt klart vist hvilke udfordinger der er i forbindelse med at få data over et trådløstnet og ind igennem et mini board og ud på Risøs netværk. Jeg har kunne konstaterer at de uSD kort der sidder på kortet får problemer med disk systemet når der læses/skrives større mængder data meget hurtig. Det er p.t uvist hvorfor Disk systemet ødelægges på uSD kortet der sidder på Beaglebone. Til dagsdato har vi ødelagt 2 Kingston 8Gb kort,dog kunne man reinstallere Ubuntu og så det til at kører igen. Men data der var på kortet var tabt! En løsning kunne være at mounte et NFS-drev fra DB-serveren,så data slet ikke læses/skrive på selve kortet. Det kræver Netforbindelsen virker altid. En anden løsning kunne være at lave en RAM disk, det vil dog betyde at genstartes opsamlingssystemet mister man de data der ligger på RAM disken.

Det er sådan at tiden er vigtig når det kommer til måleresultater. For at kunne sammenligne måleresultater med andres måleresultater skal man blive enige om hvad klokken i grunden er og hvor vi har den fra. Da beaglebone skal bruge internet til at sende sine data vil det være nærlæggende at bruge de services som det pågældende netværk tilbyder(NTP/PTP). Mere om det emne i bilag **Unixtime**.

Yderligere har det været en udfordring at disse små board kører Linux og er ikke lavede til folk der ikke har en eller anden grundlæggende kendskab til Linux. Jeg mener helt klart der er en udfordring i at lave et brugervenligt webinterface som man kunne styre hele dataopsamlings netværket på.

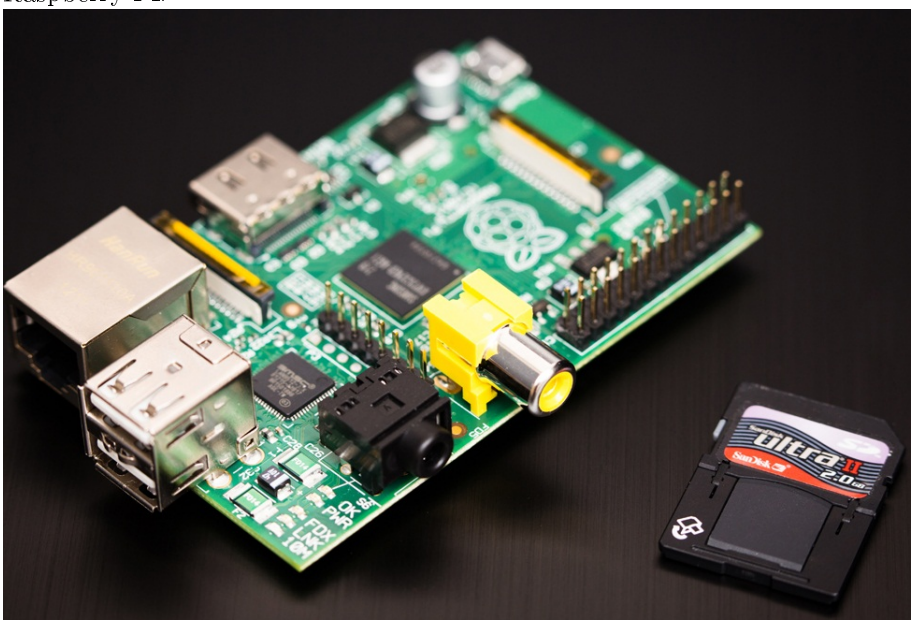
Jeg har kommet med et godt bud på et brugervenligt (debug)webinterface,som man kan se i afsnit **Styring via Web**.
Man kunne dog godt ønske sig at webinterfacet der ikke var på beaglebonen. Men i stedet var på DB-serveren på Risøs VMware miljø.
Skal man se på et forbilled her burde man nok over i sikkerhedsbranchen og se på et BOTnet,hvor kriminelle styre mange computere til at gøre noget bestemt ud fra et webinterface.

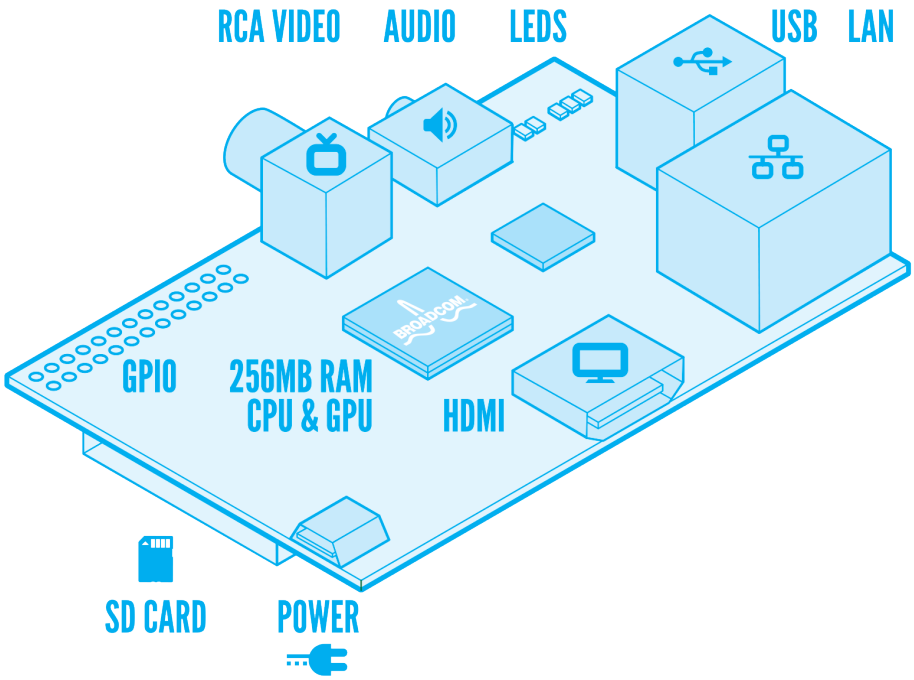
Projektet har et videre liv på Fotonik og skal rent faktisk bruges, det betyder at der ligger en ansættelse efterfølgende. For at følge dette projekt til dørs og måske få de mange ideer der er kommet frem under projektet ud til slutbrugerne.

Appendix A

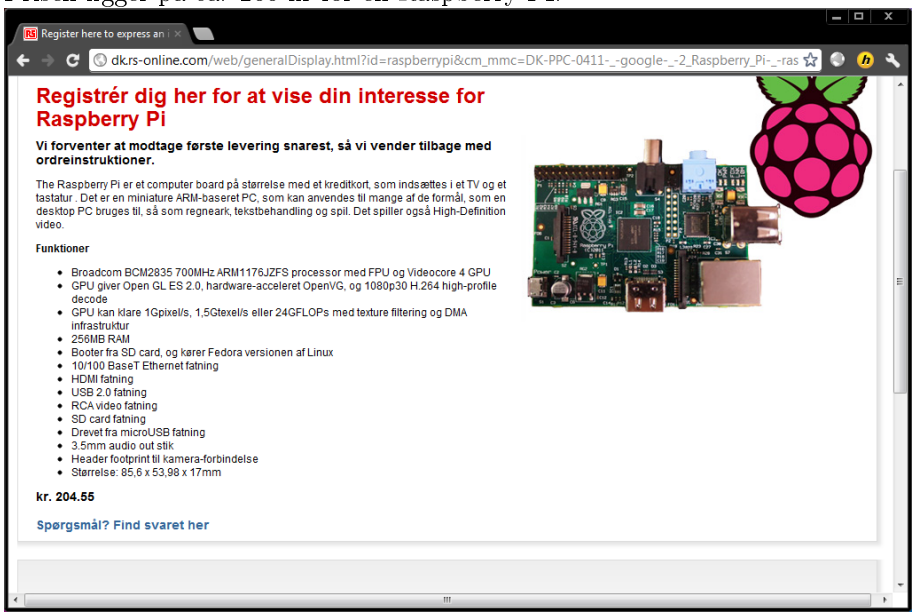
Raspberry PI

Indkøb,Mere hjælp og hvordan man installere:
`http://www.element14.com/community/groups/raspberry-pi?ICID=hp_raspberry`
`http://elinux.org/R-Pi_Hub`
`http://mitchtech.net/category/tutorials/raspberry-pi/`
`https://wiki.ubuntu.com/ARM`
Bemærk at Raspberry PI kun har 1 GPIO port og dermed ikke er særlig velegnet til data opsamling hvor der skal bruges flere porte.
Hvis man ændre opstart scriptet vil man med lidt held kunne bruge en Raspberry PI.





Prisen ligger på ca. 200 kr for en Raspberry PI.



Appendix B

Frekvens regler i Danmark

Mere information:
http://www.itu.int/ITU-R/terrestrial/faq/index.html
http:
//www.erhvervsstyrelsen.dk/frekvensplan-uddybende-information
Den fulde tekst kan læses på:
http://erst.lovportaler.dk/ShowDoc.aspx?schultzlink=bek20120459

Et udklip:(det område vi bruger!)
Frekvensbånd:
410-430 MHz (sender/modtager)
791-821 MHz (modtager)
832-862 MHz (sender)
880-915 MHz (fortrinsvis sender)
Begrænsninger i anvendelse: Brugerterminaler, som anvender
frekvenser i frekvensbåndet 791-862 MHz, kan anvendes uden tilladelse fra den 1. januar 2013.
Kilde : http://da.wikipedia.org/wiki/ISM-b%C3%A5nd
radiotags (Smart Tags) (13,5-MHz-båndet)
radiofjernstyringer (27 MHz, 40,6-MHz-bånd)
Babyphone (433-MHz-bånd)
Trådløs termometer (433-MHz-bånd)
Radiostyret omskifter, som f.eks. bilnøgle, radio-stikdåse (433-MHz-bånd)
Trådløs hovedtelefon eller radio-højttalere (868-MHz-bånd)
Trådløst videokamera (2400-MHz-bånd)
Trådløs tyverialarm (2400-MHz-bånd)
WLAN, Wi-Fi (IEEE 802.11b / IEEE 802.11g) (2400-MHz-bånd)
Bluetooth (2400-MHz-bånd)
Mikrobølgeovn (2400-Mhz-bånd)
ZigBee (2400-MHz-bånd)
HIPERLAN (5200-MHz-bånd)

Appendix C

UnixTime

We still perfecting Einstein's theory. We must apologize that our Atomic Watch loses 1 second every 20,000,000 years. Our scientists are working diligently to correct this problem. [Michael A Lombard http://tf.nist.gov/general/pdf/2429.pdfi]

Alle skal vide, hvad klokken er på et tidspunkt i dag, om at en bus om morgenen eller til at fejre nytår på det rigtige tidspunkt. Til denne form for tidtagning, er vores personlige ure og husholdningsartikler ure præcis nok. En typisk kvartsur for eksempel formår at holde tid inden for en anden i løbet af ti dage. Når det kommer til at sende data ned en telefonlinje eller navigere ved satellitter, men vi har brug for så meget præcision som muligt. Ansøgninger fra global kommunikation til satellitnavigation, landmåling og transportsystemer understøttes af præcise timing, og det samme stabil og nøjagtig tidshorisont skal være i brug overalt i sådanne systemer kan fungere korrekt.

http://www.nist.gov/pml/div688/grp40/index.cfm
http://www.meinberg.de/english/products/ntp-time-server.htm
http://www.lammertbies.nl/comm/info/GPS-time.html
Produkter: Man kunne bruge en Telit GPS - Inden for telit:
http://www.telit.com/en/products/satellite.php
http://www.meinberg.de/english/info/time-synchronization-telecom-networks.htm Brug af Radio-frekvens. NIST WWVB 60Khz som bruges i USA
http://www.nist.gov/pml/div688/grp40/wwvb.cfm
Brug af MSF – 60 khz som bruges i England.
http://www.npl.co.uk/science-technology/time-frequency/time/ Brug af GSM Ved at bruge en AVR323 Ref. http://www.atmel.com/Images/doc8016.pdf
`http://wiki.groundlab.cc/doku.php?id=gsm_library` Brug af TCP/IP (RJ-45/GPRS / 3G/4G) TTP - http://www.ttagroup.org/ttp/overview.htm NTP Network Time Protocol (RFC-1305) Daytime Protocol (RFC-867) Brug af UDP/IP (RJ-45/GPRS / 3G/4G) PTP IEEE 1588 Precision Time Protocol (PTP) http://www.nist.gov/el/isd/ieee/tutorials-1588.cfm
http://grouper.ieee.org/groups/1588/

C.0.26 Test af 2038 Bug

...Mere info : http://en.wikipedia.org/wiki/Year_2038_problem

```
#Fra Serveren:
#uname -a
#Linux xxxxx.risoe.dk 2.6.32-279.1.1.el6.x86_64 #1 SMP
#Wed Jun 20 11:41:22 EDT 2012 x86_64 x86_64 x86_64 GNU/Linux
#Script from : http://maul.deepsky.com/~merovech/2038.perl.txt
#Forventede resultat :
#Tue Jan 19 03:14:07 2038
#Fri Dec 13 20:45:52 1901
use Time::localtime;
use File::stat;
for ($clock = 2147483641; $clock < 4147499751; $clock++)
{
print ctime($clock);
print "\n";
}
```

Fra terminal: Tue Jan 19 03:14:07 2038 —-CUT—— Mon Jun 6 07:46:52 2101

`http://en.wikipedia.org/wiki/Unix_time`

Appendix D

Beagle Bone Install

Beaglebone boardet kan købes hos : `http://beagleboard.org/` hvor man også kan finde MEGET hjælp og setup guides.

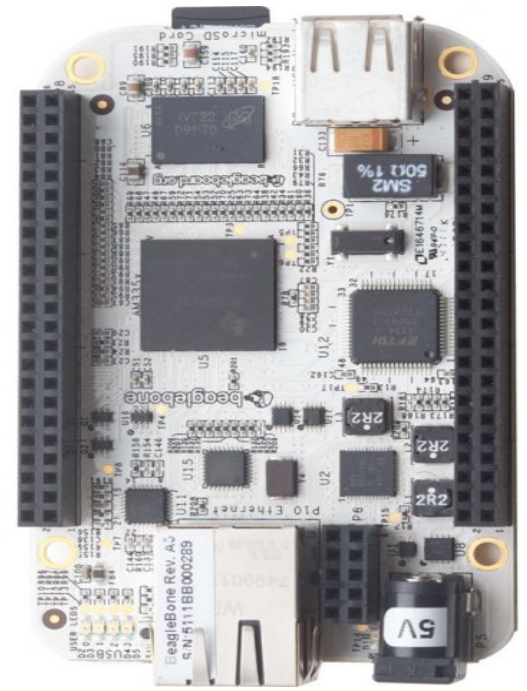


Figure D.1: Beaglebone

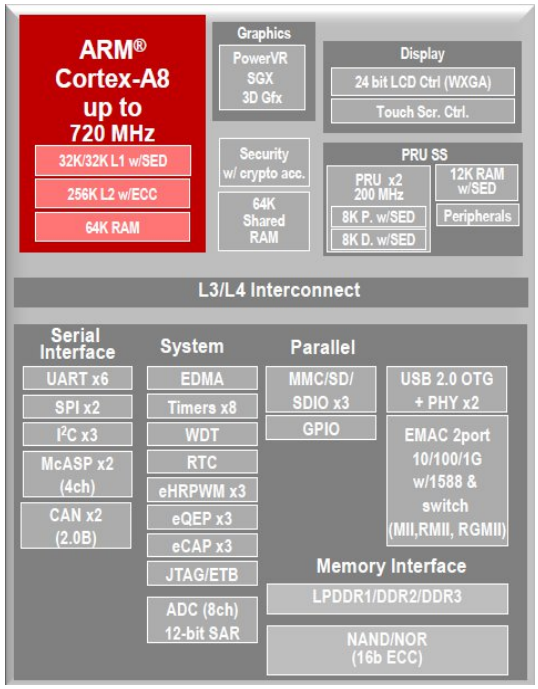


Figure D.2: CPU'en på Beaglebone

P9 pin put

	SIGNAL NAME	PIN	CONN		PIN	SIGNAL NAME	
GND		GND	1	2	GND		GND
Vcc		VDD_3V3EXP	3	4	VDD_3V3EXP		Vcc
		VDD_5V	5	6	VDD_5V		
		SYS_5V	7	8	SYS_5V		
	PWR_BUT*		9	10	A10	SYS_RESETn	
	UART4_RXD	T17	11	12	U18	GPIO1_28	
	UART4_TXD	U17	13	14	U14	EHRPWM1A	
	GPIO1_16	R13	15	16	T14	EHRPWM1B	
	I2C1_SCL	A16	17	18	B16	I2C1_SDA	
CTS	I2C2_SCL	D17	19	20	D18	I2C2_SDA	RTS1
	UART2_TXD	B17	21	22	A17	UART2_RXD	
	GPIO1_17	V14	23	24	D15	UART1_TXD	Rx
	GPIO3_21	A14	25	26	D16	UART1_RXD	Tx
	GPIO3_19	C13	27	28	C12	SPI1_CS0	
	SPI1_D0	B13	29	30	D12	SPI1_D1	
	SPI1_SCLK	A13	31	32	VDD_ADC		
	AIN4	C8	33	34	GNDA_ADC		
	AIN6	A5	35	36	A5	AIN5	
	AIN2	B7	37	38	A7	AIN3	
	AIN0	B6	39	40	C7	AIN1	
	CLKOUT2	D14	41	42	C18	GPIO0_7	
		GND	43	44	GND		
		GND	45	46	GND		

P8 pin put

	SIGNAL NAME	PROC	CONN		PROC	SIGNAL NAME	
GND		GND	1	2	GND		GND
	GPIO1_6	R9	3	4	T9	GPIO1_7	
ASSO	GPIO1_2	R8	5	6	T8	GPIO1_3	Stand_by(in)
	TIMER4	R7	7	8	T7	TIMER7	
	TIMER5	T6	9	10	U6	TIMER6	
	GPIO1_13	R12	11	12	T12	GPIO1_12	
	EHRPWM2B	T10	13	14	T11	GPIO0_26	
	GPIO1_15	U13	15	16	V13	GPIO1_14	
	GPIO0_27	U12	17	18	V12	GPIO2_1	
	EHRPWM2A	U10	19	20	V9	GPIO1_31	
	GPIO1_30	U9	21	22	V8	GPIO1_5	
Stand_by(Out)	GPIO1_4	U8	23	24	V7	GPIO1_1	Prog
Reset	GPIO1_0	U7	25	26	V6	GPIO1_29	
	GPIO2_22	U5	27	28	V5	GPIO2_24	
	GPIO2_23	R5	29	30	R6	GPIO2_25	
	UART5_CTSN	V4	31	32	T5	UART5_RTSN	
	UART4_RTSN	V3	33	34	U4	UART3_RTSN	
	UART4_CTSN	V2	35	36	U3	UART3_CTSN	
	UART5_TXD	U1	37	38	U2	UART5_RXD	
	GPIO2_12	T3	39	40	T4	GPIO2_13	
	GPIO2_10	T1	41	42	T2	GPIO2_11	
	GPIO2_8	R3	43	44	R4	GPIO2_9	
	GPIO2_6	R1	45	46	R2	GPIO2_7	

Forbindelse fra Beglebone til Telit(controlleren).
UART1 (/dev/ttyO1) P9 pins 24(TX) 26(RX)
UART2 (/dev/ttyO2) P9 pins 21(TX) 22(RX)
UART4 (/dev/ttyO4) P9 pins 13(TX) 11(RX)
UART5 (/dev/ttyO5) P8 pins 37(TX) 38(RX)

Bemærk!
Da det kan være lidt bøvlede at få disse script rigtig ud vil de være online fra Projekt-hjemmesiden som en pakkede file. Med vejledning i hvordan de installeres.
Projekt hjemmesiden er: `http://co2nsl.deadmeat.dk`

En kort beskrivelse af hvordan man installere Ubuntu og DAQ-softwaren på Beaglebonen. Den kan kører mange forskellige operativsystemer,så som: Android,Ubuntu,Debian,ArchLinux,Gentoo,Sabayon,Erlang,Fedora,QNX, FreeBSD Beaglebone kommer med et default operativsystem som hedder Ångström. Denne distro bygger på Gentoo. De fleste "ikke nørder" der har brugt Linux på et eller andet tidspunkt har mest rendt ind i Ubuntu eller RedHat.Derfor har jeg valgt at lægge en Ubuntu på uSDkortet.

```
1  #/bin/sh
2  echo "Default user: ubuntu pass: temppwd"
3  wget http://rcn-ee.net/deb/rootfs/precise
4  /ubuntu-12.04-r4-minimal-armhf-2012-07-16.tar.xz
5
6  md5sum ubuntu-12.04-r4-minimal-armhf-2012-07-16.tar.xz
7  echo "Is it:84dc6bb4c4d74ea2d45faf9aef80b819 ?"
8  tar xJf ubuntu-12.04-r4-minimal-armhf-2012-07-16.tar.xz
9  cd ubuntu-12.04-r4-minimal-armhf-2012-07-16
10 sudo ./setup_sdcard.sh --probe-mmc
11 echo "What is the /dev/ of your uSD card?"
12 # sudo ./setup_sdcard.sh --mmc /dev/sdb --uboot bone
```

For de Avanceret kan man bygge sin egen kernel. Det kan give mere hastighed da kernel kun indeholder de moduler der bruges og den passer til boardet.
Robert C Nelson Image Build :
`https://github.com/RobertCNelson/omap-image-builder`

```
git clone git://github.com/RobertCNelson/stable-kernel.git
cd stable-kernel
./build_kernel.sh

git clone git://github.com/RobertCNelson/linux-dev.git
cd linux-dev
git checkout origin/am33x-v3.2 -b am33x-v3.2
./build_kernel.sh
```

SSH til Beaglebone og skriv : ntpdate 130.226.56.67
Setup TimeZone: sudo dpkg-reconfigure tzdata Current default time zone: 'Europe/Copenhagen'

Installering af programmerne:
sudo apt-get install apache2 apache2-mpm-prefork apache2-utils apache2.2-bin apache2.2-common php5 libapache2-mod-php5 php5-mysql perl perl-base perl-modules php5-cli php5-common php5-mysql munin-common munin-node munin-plugins-extra

For at få kommandoerne til at virke fra Webserveren skal man
give webserveren mulighed for at afvikle shell kode som root.
Dette gøres med visudo : `https://help.ubuntu.com/community/Sudoers`
Der skal man tilføje : www-data ALL=NOPASSWD: ALL

Restart Serial Beaglebone:

```
<?php

echo "Restart Serial Interface.. Back in 2 min...";
$output = shell_exec('sudo /etc/init.d/serialrestart.sh');
echo "<pre>$output</pre>";
?>
```

Scriptet serialrestart.sh:

```
#!/bin/sh
/usr/bin/killall -HUP perl
/usr/sbin/ntpdate 130.226.56.67 >> /var/www/lastdate.txt &
perl /home/ubuntu/serial_log.pl >> /var/www/beaglebone.txt &
```

Reboot Beaglebone:

```
1 <?php
2 $output = shell_exec('sudo reboot');
3 echo "<pre>$output</pre>";
4 echo "REBOOTING!!!! 3 min. and we are back!"; ?>
```

Reset Telit:

```
1  <?php
2  $output = shell_exec('echo "high" >
3  /sys/class/gpio/gpio32/direction ');
4  $output = shell_exec('echo "low" >
5  /sys/class/gpio/gpio32/direction');
6  $output = shell_exec('echo "low" >
7  /sys/class/gpio/gpio33/direction');
8  $output = shell_exec('echo "low" >
9  /sys/class/gpio/gpio34/direction');
10 $output = shell_exec('echo "in" >
11 /sys/class/gpio/gpio34/direction');
12 $output = shell_exec('echo "low" >
13 /sys/class/gpio/gpio35/direction ');
14 $output = shell_exec('echo "low" >
15 /sys/class/gpio/gpio36/direction');
16 $output = shell_exec('echo "in" >
17 /sys/class/gpio/gpio36/direction ');
18 $output = shell_exec('echo "high" >
19 /sys/class/gpio/gpio32/direction');
20 $output = shell_exec('echo 20 >
21 /sys/kernel/debug/omap_mux/uart0_rxd');
22 $output = shell_exec('echo 0 >
23 /sys/kernel/debug/omap_mux/uart0_txd');
24 $output = shell_exec('echo 20 >
25 /sys/kernel/debug/omap_mux/uart1_rxd');
26 $output = shell_exec('echo 0 >
27 /sys/kernel/debug/omap_mux/uart1_txd');
28 echo "Reseting Telit
29 on Beaglebone(Controller)!"; ?>
```

Mere hjælp online på Ubuntu Help page: `https://help.ubuntu.com/`
LAMP: `http://www.howtoforge.com/ubuntu_lamp_for_newbies`

Script:index.php

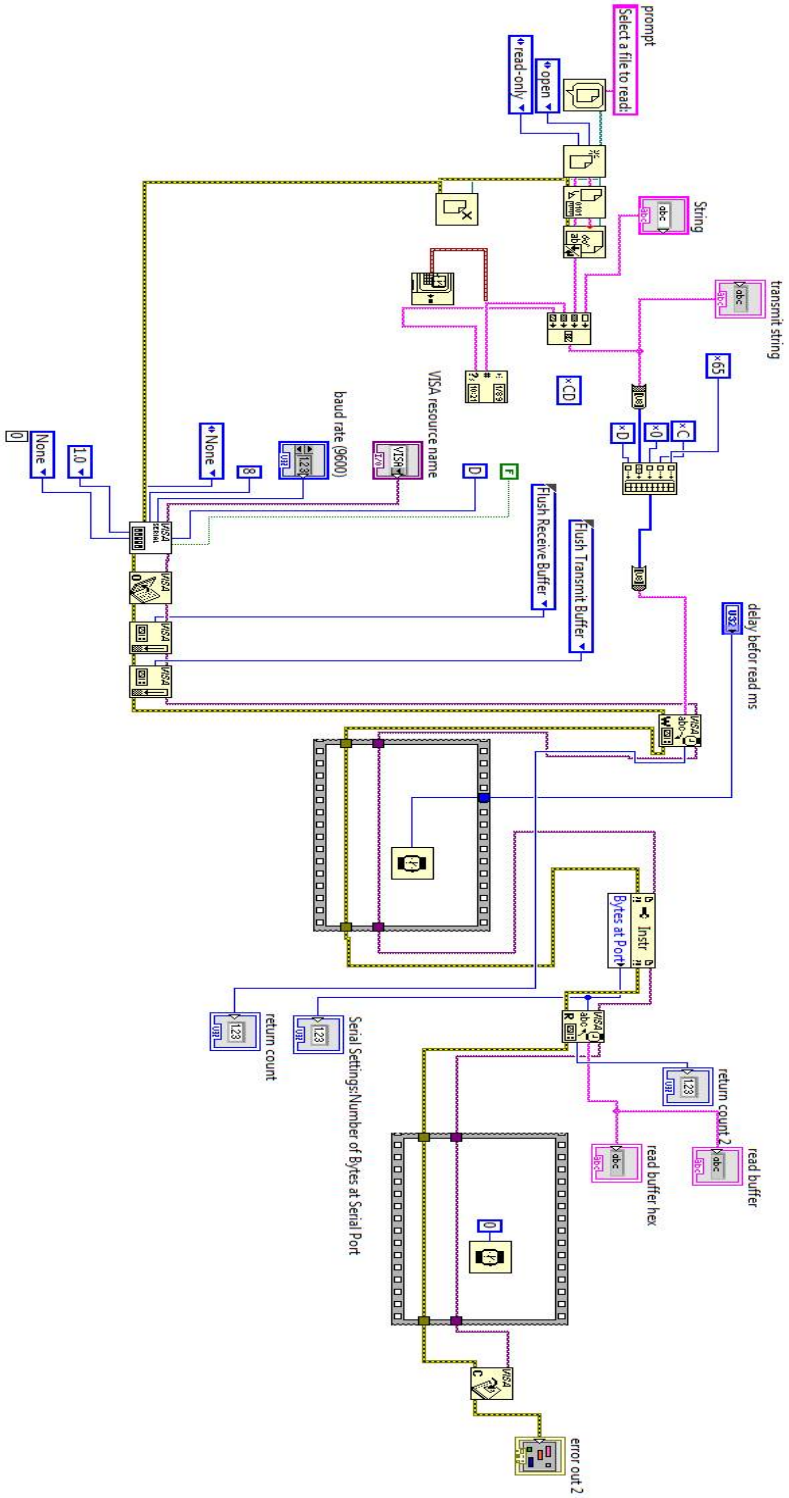
```
<html><head>
<title>DAQ system @ Fotonik-Risø</title>
<META http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<link rel="stylesheet" type="text/css" href="main.css">
</head><body> <table class="Global" cellpadding="0" cellspacing="0" border="0">
<tr><td height="30" colspan="2"><div class="Header_Text">FOTONIK DAQ system</div></td></tr>
<tr><td height="150" colspan="2"><table class="Header_Image" cellspacing="0" border="0"><tr>
<td width="100%" align="left"> </td></tr></table></td></tr><tr>
<td class="MenuColTD" align="center">
<div class="Menu"><FORM METHOD="LINK" ACTION="reboot.php">
 <INPUT TYPE="submit" VALUE="Reboot Beagle Bone(PANIC!)">
 </FORM><FORM METHOD="LINK" ACTION="restartserial.php">
 <INPUT TYPE="submit" VALUE="Restart Serial Interface">
 </FORM><FORM METHOD="LINK" ACTION="watchdog.php">
 <INPUT TYPE="submit" VALUE="Force Watch Dog run">
 </FORM><FORM METHOD="LINK" ACTION="resettelit.php">
 <INPUT TYPE="submit" VALUE="Send Reset to Telit">
 </FORM><hr><FORM METHOD="LINK" ACTION="test.php">
 <INPUT TYPE="submit" VALUE="Database Running test(BB)">
 </FORM><FORM METHOD="LINK" ACTION="test2.php">
 <INPUT TYPE="submit" VALUE="Database Running test(Server)">
 </FORM><FORM METHOD="LINK" ACTION="test3.php">
 <INPUT TYPE="submit" VALUE="Netstats on Beaglebone">
 </FORM><hr><FORM METHOD="LINK" ACTION="beaglebone.txt">
 <INPUT TYPE="submit" VALUE="BB Serial-log (UTC time)">
 </FORM><FORM METHOD="LINK" ACTION="/log/index.php">
 <INPUT TYPE="submit" VALUE="BB-logs(OLD .gz packed)">
 </FORM><FORM METHOD="LINK" ACTION="debug.txt">
 <INPUT TYPE="submit" VALUE="BB-logs(RAW Serial)">
 </FORM><FORM METHOD="LINK" ACTION="lastdate.txt">
<INPUT TYPE="submit" VALUE="BB WatchDog date-log">
</FORM></div> </td>
<td class="Content">
<h1 class="HeadingStyle">STATUS on BB server:</h1>
Universal Time is now: -2 <br>
Current default time zone: Europe/Copenhagen  <br>
<?php $output = shell_exec('sudo date');
echo "Beaglebone Local time: ";
echo "$output"; ?>
<?php  $output = shell_exec('sudo cpufreq-info');
echo "<pre>$output</pre>";
echo "UART info";
$output = shell_exec('sudo cat /sys/kernel/debug/omap_mux/uart*');
echo "<pre>$output</pre>";
echo "GPIO:"; $output = shell_exec('sudo cat /sys/kernel/debug/gpio');
echo "<pre>$output</pre>";
echo "Space on Disk:"; $output = shell_exec('sudo df');
echo "<pre>$output</pre>";
echo "What services is runnning?"; $output = shell_exec('sudo netstat -tna');
echo "<pre>$output</pre>";echo "Whats running?(PS)"; $output = shell_exec('sudo ps -aux')
echo "<pre>$output</pre>";
?> </td></tr><tr><td colspan="2" class="Footer"><div class="FooterWrap">
</div></td></tr></table></body></html>
```

Appendix E

Labview test

Labview test programmet. Der sendes AT kommandoen ind på Telit og derefter sendes der test data.Her rendte vi ind i en sjov lille tidsfejl. Det er sådan at Labview benytter filer fra MacOS og dermed har en wrap-round tid til januar 1 1904. Det er beskrevet af en bruger på Labviewsforum.
Here's the deal with January 1, 1904. Most (all?) operating systems store time as the number of seconds (or microseconds, etc.) since a certain time. This time is call the epoch. The number of seconds that the Mac could store covered about 130 years. The programmers wanted the system to be able to cover the birthdays of most users, so the epoch needed to start in the early 20th century. So why not start at January 1, 1900? The Macintosh designers needed to squish a lot of code into a very small amount of ROM/RAM. They were looking for any kind of tweak that could save them some room. The code to turn the number of seconds since the epoch into a date (like March 29, 1973) needs to know how many days there are in a year. Which means you need code to determine if a year is a leap year. A year is a leap year if it is a multiple of 4 unless it is a multiple of 100 except if it is also a multiple of 400. So, 1900 wasn't a leap year (multiple of 100), but 2000 was (multiple of 100, but also a multiple of 400). By avoiding the year 1900, the Mac designers never had to worry about the 100/400 part of the leap year rule (because the 130 year span would only cover 2000 which was a leap year). If the year was a multiple of 4, it was a leap year.So why not 1901 or 1902 or 1903? By starting with a leap year they didn't need an 'offset', just another way to save a little bit of space.And since LabVIEW started on the Mac, it inherited some of those quirks.
Source :`http://forums.ni.com/t5/LabVIEW/What-s-with-12AM-Friday-January-01-1904/td-p/409440`

The 1900 Date System vs. the 1904 Date System:
`http://support.microsoft.com/kb/q180162`


prompt
Select a file to read
read-only
open
String
transmit string
VISA resource name
baud rate (9600)
None
None
Flush Receive Buffer
Flush Transmit Buffer
delay befor read ms
Bytes at Port
return count
Serial Settings:Number of Bytes at Serial Port
return count 2
read buffer hex
read buffer
error out 2

Appendix F

GPIO-Kernel

Da hele gpio.txt filen fylder meget er valgt et lille klip fra filen. kald og setting i Sysfs til GPIO kan man læse mere om i den fulde tekst. Den fulde tekst findes på :
`http://www.kernel.org/doc/Documentation/gpio.txt`
Mere:
`http://processors.wiki.ti.com/index.php/GPIO_Driver_Guide`

A "General Purpose Input/Output" (GPIO) is a flexible software-controlled digital signal. They are provided from many kinds of chip, and are familiar to Linux developers working with embedded and custom hardware. Each GPIO represents a bit connected to a particular pin, or "ball" on Ball Grid Array (BGA) packages. Board schematics show which external hardware connects to which GPIOs. Drivers can be written generically, so that board setup code passes such pin configuration data to drivers.

System-on-Chip (SOC) processors heavily rely on GPIOs. In some cases, every non-dedicated pin can be configured as a GPIO; and most chips have at least several dozen of them. Programmable logic devices (like FPGAs) can easily provide GPIOs; multifunction chips like power managers, and audio codecs often have a few such pins to help with pin scarcity on SOCs; and there are also "GPIO Expander" chips that connect using the I2C or SPI serial busses. Most PC southbridges have a few dozen GPIO-capable pins (with only the BIOS firmware knowing how they're used).

The exact capabilities of GPIOs vary between systems. Common options:

- Output values are writable (high=1, low=0). Some chips also have options about how that value is driven, so that for example only one value might be driven ... supporting "wire-OR" and similar schemes for the other value (notably, "open drain" signaling).

- Input values are likewise readable (1, 0). Some chips support readback of pins configured as "output", which is very useful in such "wire-OR" cases (to

support bidirectional signaling). GPIO controllers may have input de-glitch/debounce logic, sometimes with software controls.

- Inputs can often be used as IRQ signals, often edge triggered but sometimes level triggered. Such IRQs may be configurable as system wakeup events, to wake the system from a low power state.

- Usually a GPIO will be configurable as either input or output, as needed by different product boards; single direction ones exist too.

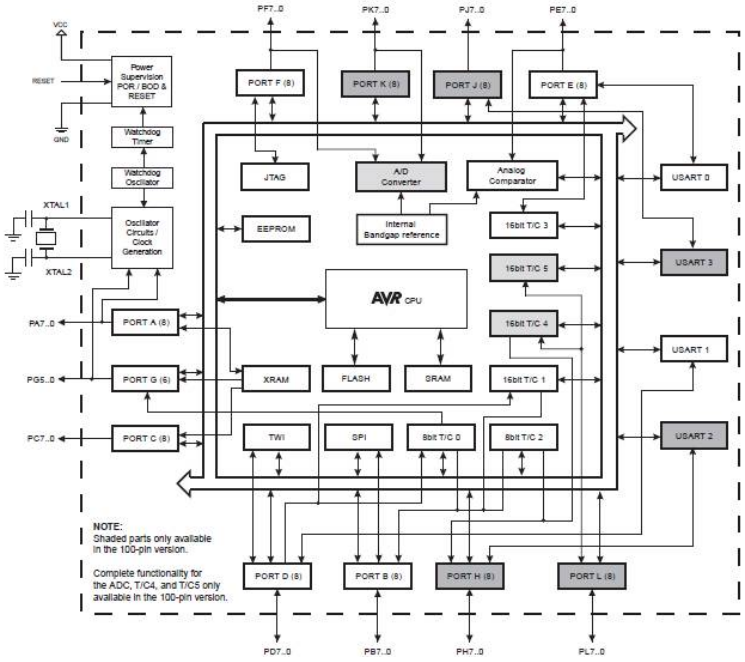
- Most GPIOs can be accessed while holding spinlocks, but those accessed through a serial bus normally can't. Some systems support both types.

On a given board each GPIO is used for one specific purpose like monitoring MMC/SD card insertion/removal, detecting card writeprotect status, driving a LED, configuring a transceiver, bitbanging a serial bus, poking a hardware watchdog, sensing a switch, and so on.

GPIO conventions Note that this is called a "convention" because you don't need to do it this way, and it's no crime if you don't. There **are** cases where portability is not the main issue; GPIOs are often used for the kind of board-specific glue logic that may even change between board revisions, and can't ever be used on a board that's wired differently. Only least-common-denominator functionality can be very portable. Other features are platform-specific, and that can be critical for glue logic.Plus, this doesn't require any implementation framework, just an interface.

Appendix G

AVR640



Appendix H

ModtageData Koden(PERL)

DATA modtages i formattet :
ef0 T=1340177375 A=68 B=476 C=112 D=926 E=578 F=738 G=969 H=199 I=93 J=885 K=9 L=467
ed0 T=1340177387 A=318 B=392 C=454 D=392 E=965 F=654 G=403 H=529 I=916 J=972 K=108 L=307

```
#!/usr/bin/perl -w
use Device::SerialPort;
use DBI;
use DBD::mysql;
my $port_n ="/dev/ttyO1";
my $port_obj = new Device::SerialPort ($port_n) || die " Can't open $port_n $!";
my $buffer = 8192;
#Serial port Parameters
$baud_rate=19200;
$port_obj->databits(8);
$port_obj->handshake("none");
$port_obj->baudrate($baud_rate);
$port_obj->parity("none");
$port_obj->stopbits(1);
$port_obj->buffers($buffer,$buffer);#(110,110); #(8192,8192);
################DB#######################
# CONFIG VARIABLES
$database = "beaglebone"; #Serveren
$host = "192.168.1.4"; # Server
$user = "fotonik";
$pw = "fotonik"; # DATA SOURCE NAME
$dsn = "dbi:mysql:$database:$host:3306";
################DB#######################
while(1){
$connect = DBI->connect($dsn, $user, $pw);
#$connect ->disconnect();
select(undef,undef,undef,0.80);
$data_read=$port_obj->read($buffer);
#print "$data_read\n";
###################RAW Log#############################
open (MYFILE, '>>/var/www/debug.txt');
print MYFILE "$data_read\n";
close (MYFILE);
#########################Telit styring###################
$Netnr=42;
$BBT = time;
#########################################################
$query = "INSERT INTO test (Telitnr,Netnr,BBT,T,A,B,C,D,E,F,G,H,I,J,K,L)
VALUES (?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)";
$query_handle = $connect->prepare($query);
#####################Telit 0D#######Begin##############
if($data_read =~ m/ed0\s+T=(\d+)\s+A=(\d+)\s+B=(\d+)\s+C=(\d+)\s+D=(\d+)
\s+E=(\d+)\s+F=(\d+)\s+G=(\d+)\s+H=(\d+)\s+I=(\d+)\s+J=(\d+)\s+K=(\d+)\s+L=(\d+)/)
{    $Telitnr=13; #String
printf("ed0:%s\n",join(',',($BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13)));
    $query_handle->execute($Telitnr,$Netnr,$BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13);
}
#####################Telit 0E#######Begin##############
if($data_read =~ m/ee0\s+T=(\d+)\s+A=(\d+)\s+B=(\d+)\s+C=(\d+)\s+D=(\d+)\s+E=(\d+)
\s+F=(\d+)\s+G=(\d+)\s+H=(\d+)\s+I=(\d+)\s+J=(\d+)\s+K=(\d+)\s+L=(\d+)/)
{    $Telitnr=14; #String ?
printf("ee0:%s\n",join(',',($BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13)));
$query_handle->execute($Telitnr,$Netnr,$BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13);
}
#####################Telit 0F#######Begin##############
if($data_read =~ m/e0f\s+T=(\d+)\s+A=(\d+)\s+B=(\d+)\s+C=(\d+)\s+D=(\d+)\s+E=(\d+)
\s+F=(\d+)\s+G=(\d+)\s+H=(\d+)\s+I=(\d+)\s+J=(\d+)\s+K=(\d+)\s+L=(\d+)/)
{    $Telitnr=15; #String ?
printf("e0f:%s\n",join(',',($BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13)));
    $query_handle->execute($Telitnr,$Netnr,$BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13);
}
if($data_read =~ m/ef0\s+T=(\d+)\s+A=(\d+)\s+B=(\d+)\s+C=(\d+)\s+D=(\d+)\s+E=(\d+)
\s+F=(\d+)\s+G=(\d+)\s+H=(\d+)\s+I=(\d+)\s+J=(\d+)\s+K=(\d+)\s+L=(\d+)/)
{    $Telitnr=15; #String ?
printf("ef0:%s\n",join(',',($BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13)));
    $query_handle->execute($Telitnr,$Netnr,$BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13);
}
#####################Telit 0G#######Begin##############
if($data_read =~ m/e0g\s+T=(\d+)\s+A=(\d+)\s+B=(\d+)\s+C=(\d+)\s+D=(\d+)\s+E=(\d+)
\s+F=(\d+)\s+G=(\d+)\s+H=(\d+)\s+I=(\d+)\s+J=(\d+)\s+K=(\d+)\s+L=(\d+)/)
{    $Telitnr=16; #String ?
printf("e0g:%s\n",join(',',($BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13)));
$query_handle->execute($Telitnr,$Netnr,$BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13);
}
if($data_read =~ m/eg0\s+T=(\d+)\s+A=(\d+)\s+B=(\d+)\s+C=(\d+)\s+D=(\d+)\s+E=(\d+)
\s+F=(\d+)\s+G=(\d+)\s+H=(\d+)\s+I=(\d+)\s+J=(\d+)\s+K=(\d+)\s+L=(\d+)/)
{    $Telitnr=16;
printf("eg0:%s\n",join(',',($BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13)));
$query_handle->execute($Telitnr,$Netnr,$BBT,$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13);
}
```

Appendix I

DB server-CentOS

Install en minimal version, uden nogen pakker. (heller ikke gnome/KDE) - der uploades via SCP (evt. winscp) – og IKKE via FTP. Login med root over ssh. (husk at enable sshd under setup -> services)

```
1
2  Installere en editor :
3  yum install nano  (ellers erstat nano med hvad du bruger)
4  rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY* ; yum update
5  yum install wget bzip2 unzip zip nmap
6  openssl lynx fileutils ncftp gcc gcc-c++
7
8  MySQL:
9  yum install mysql mysql-devel mysql-server;
10 chkconfig --levels 235 mysqld on /etc/init.d/mysqld start;
11 mysqladmin -u root password [dit mysql-root password]
12
13 install PHP:
14 yum install php php-devel php-gd php-imap
15 php-ldap php-mysql php-odbc php-pear php-xml
16 php-xmlrpc curl curl-devel perl-libwww-perl
17 ImageMagick libxml2 libxml2-devel;
18
19 [/etc/httpd/conf/httpd.conf ] DirectoryIndex index.html
20 index.htm index.shtml index.cgi index.php
21
22 install TLS: yum install cyrus-sasl cyrus-sasl-devel
23  cyrus-sasl-gssapi cyrus-sasl-md5 cyrus-sasl-plain dovecot
24
25 SSL Cerfikat:
26 yum install mod_ssl;
27 openssl genrsa -des3 -out server.key 1024 ;
28  openssl req -new -key server.key -out server.csr ;
29  openssl x509 -req -days 365 -in /root/server.csr
30  -signkey /root/server.key -out /root/server.crt
31
```

```
cp ~/server.crt /etc/httpd/ ;
cp ~/server.key /etc/httpd/
chmod 0400 /etc/httpd/server.crt;
chmod 0400 /etc/httpd/server.key
cp server.key server.key.orig;
openssl rsa -in server.key.orig -out server.key
Filen /etc/httpd/conf/httpd.conf

Tiden NTP:
yum install ntp;
chkconfig --levels 235 ntpd on;
ntpdate Risoe-NTP ;
/etc/init.d/ntpd start;

Automatisk Patches:
nano /etc/cron.weekly/time.sh

Skriv:
/usr/sbin/ntpdate [Risoe-NTP];
yum -y update ; yum -y upgrade
```

```
mysql -u root
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('yourpassword');

CREATE DATABASE beaglebone;
drop table `beaglebone`.`test`;
use beaglebone
CREATE TABLE test (Telitnr INT, Netnr INT,BBT INT,T INT,A INT,
B INT,C INT,D INT,E INT,F INT,G INT,H INT, I INT, J INT,K INT,L INT);
CREATE TABLE board (BoardSN INT,CPU INT,BBT INT,T INT,A INT,
B INT,C INT,D INT,E INT,F INT,G INT,H INT, I INT, J INT,K INT,L INT);
GRANT ALL PRIVILEGES ON *.* TO 'fotonik'@'localhost'
IDENTIFIED BY 'fotonik' WITH GRANT OPTION;
GRANT ALL ON test.* TO fotonik@'192.168.1.1' IDENTIFIED BY 'fotonik';
GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, DROP, INDEX,
 ALTER, CREATE TEMPORARY TABLES, LOCK TABLES ON test.* TO
 'fotonik'@'localhost' IDENTIFIED BY 'fotonik';
GRANT ALL PRIVILEGES ON *.* TO 'fotonik'@'192.168.1.6'
IDENTIFIED BY 'fotonik' WITH GRANT OPTION;
```

Mere hjælp og hvordan man installere:
`http://www.howtoforge.com/quick-n-easy-lamp-server-centos-rhel`

Appendix J

Dansk vejr

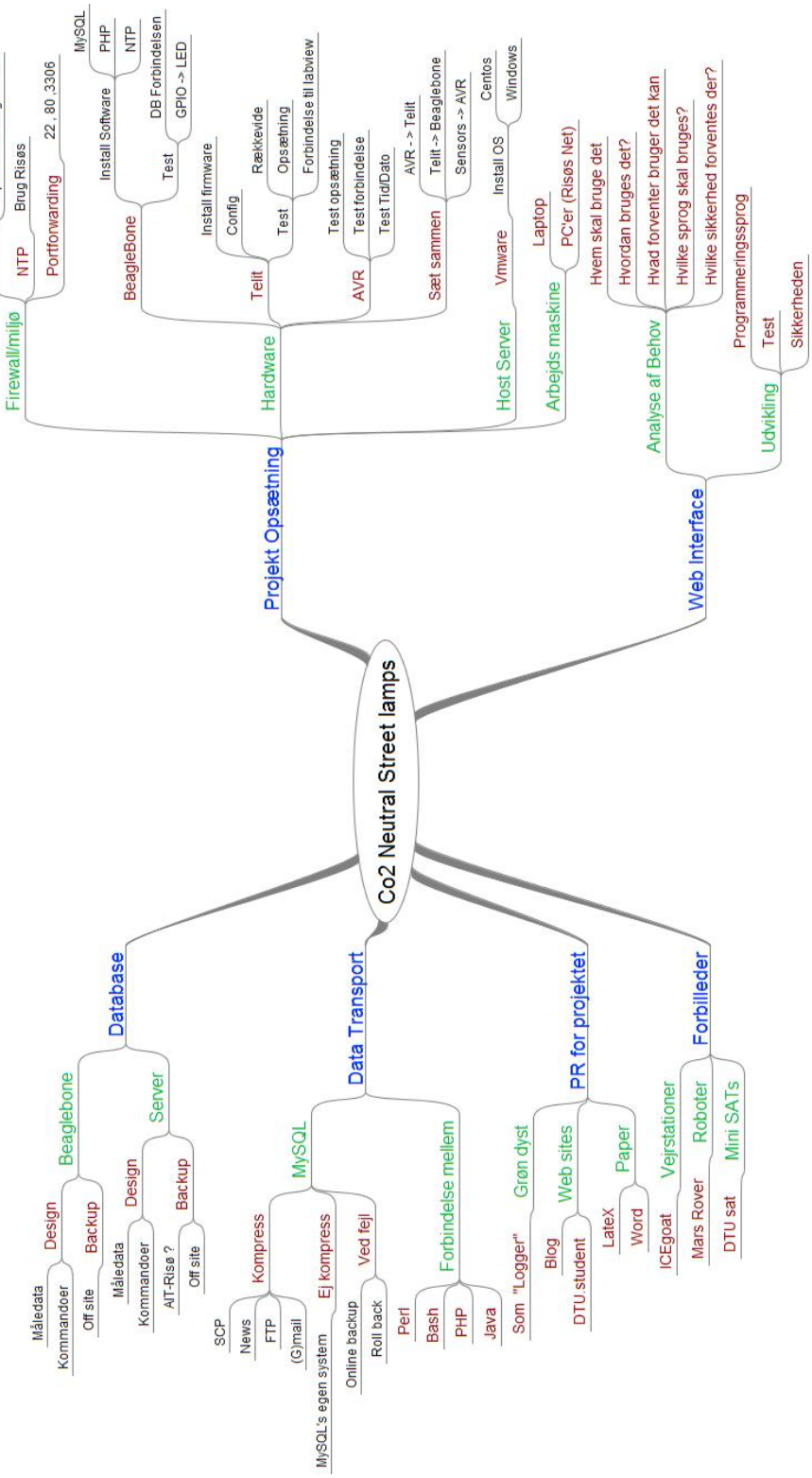
Måned	Gemmensnit °C	maks. °C	min. °C	nedbør mm	soltimer
Januar	0,3 (0,0)	9,8	-12,4	46 (57)	72 (43)
Februar	-0,1 (0,0)	9,6	-16,5	40 (38)	52 (69)
Marts	3,1 (2,1)	15,3	-7,6	31 (46)	143 (110)
April	9,9 (5,7)	22,5	-1,6	16 (41)	253 (162)
Maj	11,4 (10,8)	26,2	-3,2	54 (48)	239 (209)
Juni	15,1 (14,3)	28,2	2,6	75 (55)	252 (209)
Juli	16,4 (15,6)	27,1	6,2	113 (66)	171 (196)
August	16,1 (15,7)	27,6	4,9	132 (67)	150 (186)
September	14,1 (12,7)	25,9	3,8	92 (73)	135 (128)
Oktober	9,8 (9,1)	26,9	-2,6	61 (76)	130 (87)
November	6,7 (4,7)	14,6	-3,9	18 (79)	37 (54)
December	4,2 (1,6)	11,3	-5,1	99 (66)	50 (43)
Året	9.0 (7,7)	28,2	-16,5	779 (712)	1.683 (1.495)

Tal kommer fra DMI
`http://www.dmi.dk/dmi/vejret_i_danmark_-_aret_2011`
En måde at holde udstyret isfrit på er at bruge "heaters" eller opvarmere. Her skal det bemærkes at heaters bruger MEGET strøm og må kun bruges hvis det er vigtig!. Heaters fra minco:
`http://www.minco.com/products/heaters.aspx`

Is vindmølle Foto fra: `http://www.geosummit.org`
Tilladelse af : David Lipson.

Appendix K

MindMap


Co2 Neutral Street lamps
Projekt Opsætning
Firewall/miljø
DHCP
Spær IP til server
Spær IP til BeagleBone
NTP
Brug Risøs
Portforwarding
22 , 80 ,3306
Hardware
BeagleBone
Install Software
MySQL
PHP
NTP
Test
DB Forbindelsen
GPIO -> LED
Telit
Install firmware
Config
Test
Rækkevide
Opsætning
Forbindelse til labview
AVR
Test opsætning
Test forbindelse
Test Tid/Dato
Sæt sammen
AVR -> Telit
Telit -> Beaglebone
Sensors -> AVR
Host Server
Vmware
Install OS
Centos
Windows
Arbejds maskine
Laptop
PC'er (Risøs Net)
Web Interface
Analyse af Behov
Hvem skal bruge det
Hvordan bruges det?
Hvad forventer bruger det kan
Hvilke sprog skal bruges?
Hvilke sikkerhed forventes der?
Udvikling
Programmeringssprog
Test
Sikkerheden
Database
Beaglebone
Design
Måledata
Kommandoer
Backup
Off site
Server
Design
Måledata
Kommandoer
Backup
AIT-Risø ?
Off site
Data Transport
MySQL
Kompress
SCP
News
FTP
(G)mail
Ej kompress
MySQL's egen system
Online backup
Ved fejl
Roll back
Forbindelse mellem
Perl
Bash
PHP
Java
PR for projektet
Grøn dyst
Som "Logger"
Blog
Web sites
DTU student
Paper
LateX
Word
Forbilleder
Vejrstationer
ICEgoat
Roboter
Mars Rover
Mini SATs
DTU sat

Appendix L

GRØN DYST2012
Abstract+Poster

Co2 Neutral Streetlights

(low power & low cost data logger)

Sune Andersen For Grøn Dyst 2012

DTU – IMM/Fotonik, Technical University of Denmark
Supervisors Finn Gustafsson and Henning Engelbrecht Larsen

INTRODUCTION

Risø National Laboratory is getting more and more requests from The danish government on how to save energy. One of the main issue is saving money on power,special when it comes to streetlight. Before the end of the year 2012,1500 street lamps around Copenhagen will be changed for light sources with low power consumption. Technical and Environmental turn down the energy as a part of Copenhagen's goal of reducing the city's CO2 emissions by 20 percent by the end of year 2015. But how much power will the new lamps comsume? And can a street lamp produce sufficient power even in Denmark?. Here will a low cost & lowpower[1] Datalogger come handy.

DESIGN

The data logger is an electronic device that records earthquakes(Sensor network), Wind ,daylight ,power used/produced on the street lamp over time. Data will then be uploaded via a wireless radio MESH[5] network(868 Mhz) to a database server for later analyze. The Prototype is developed on two microcontrollers(AVR and ARM Cortex-A8) with the low power and with fault tolerant in mind, equipped with extra storage for offline catching(like a uSD(16/32Gb)).The ARM CortexA8-board is running a full version of Ubuntu(OMAP), with Apache-webserver,PHP and MySQL-database for local catching of data, in case of the network is offline. Data will then be sync with the database server then there is connectivity. Controlling the Datalogger device can be done from the control center's webinterface or on the device it self(via Web or SSH). The device can even be used for other purposes like a (MESH) WIFI net, something like freifunk in Berlin & WNDW[3,2]. In a catastrophe area the “lamp-network” will still be running (because it is off-grid), even when the infrastructure is destroyed or very heavy loaded.

HOW LOW IS LOW?

How low cost ? In a price range of around max. 100 $ pr. main-unit and around 30$ for each 868 mhz-node(max.253). A data logger with the same functions can be something like the DT82E[4] datalogger(no nodes!) from DataTaker that have a price of £752 pr. Unit.(like 1190 $) please note that’s in 2012 exchange rate. *How low power ?* The main board is running on max.4 Watt-5volt with a 500(720) Mhz Cortex-A8 and the 868 mhz-nodes is running on max.1 watt-3.3volt. The goal is a very low/non power-footprints on the measurements ,in this case a street lamp.

REFERENCES

1. Michael Holik. Diploma Thesis, *Ultra Low Power Datalogger*, University of West Bohemia, Pilsen, 2009.
2. Wireless Networking in the Developing World : http://wndw.net/
3. FreiFunk-Berlin : http://berlin.freifunk.net/.
4. DT82E : http://www.microdaq.com/datataker/6-analog-input-channels.php
5. Environmental Monitoring System with Wireless Mesh Network Based on Embedded System. Fifth IEEE International Symposium on Embedded Computing

Low power & Low cost data logger for CO_2 Neutral Streetlights

Sune Andersen – suna@student.dtu.dk

DTU Informatik

DTU Wind (Risø-campus)

DTU Fotonik (Risø-campus)

Introduction and Motivation

The Danish municipalities are going to replace a major part of the street lamps in the years to come. Interest in using renewable energy for street lamps is increasing and DTU Fotonik and DTU Wind are investigating solar and wind powered LED systems for street lights.
There is a need for simple, smart, cheap, low power data logger system for gathering information about different test setups.

The data logger system will provide new possibilities to:
- Control the lamps intelligently (through sensors)
- Network services
- Error messages

Old System

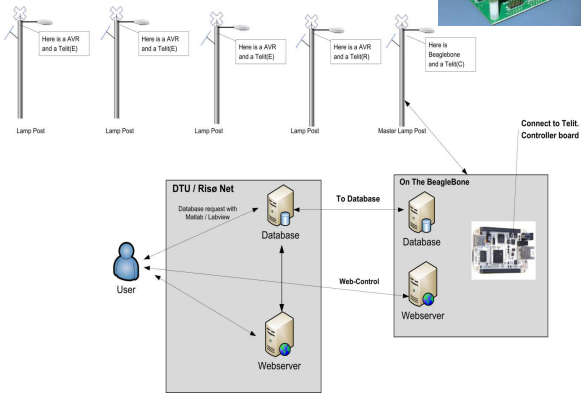
- A standard data acquisition : 300Watt
- Hardware: Dell Optiplex in a modcase.
- Running Microsoft Windows
- Price : ~400 US$

New System

- Low power data acquisition : 2 Watt
- Hardware: Beagle Bone[2]
- Running Ubuntu (Linux)
- Price : ~80 US$

The "less" Smart Lamp system: Idea 1

- Very low power Data acquisition system.
- With Telit[3] no frequency license needed because it's using ISM-band[1]
- Easy scaling without downtime to ~100x12 sensors or more….
- Can be used for collecting data from sensors like: air quality,CO_2 level and weather.



Example of an CO_2 Neutral Street Light:

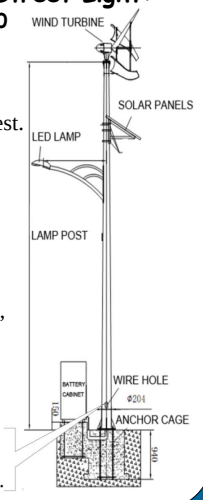
The Nheohybrid 400

Source: www.nheolis.com.cn

On Risø campus a CO2 neutral street light from China are being set up for test.

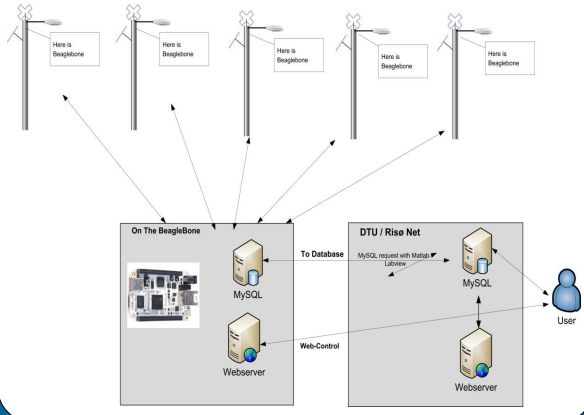
The data logger system will provide valuable data on:

the operation of the system components,wind turbine, solar cell energy production, actual local windsolar resources, battery state/load and LED consumption.



The Smart Lamp system: Idea 2

- Offer all protocols as a "normal" Linux server in all lamps.
- Can offer wireless highspeed Internet and phone services.
- Can be used as a alternatively relay network(MESH)
- Can be used for **live feed** and collecting data from any sensors like: air quality,CO_2 level,weather & traffic cameras.



Conclusion and Outlook

A cheap compact, reliable data logger system has been developed and Installed in testing of CO2 neutral streetlight on Risø campus. As a bonus it will offer the "smart lamp" functionalities.

References

1. **ISM(Industrial, Scientific, and Medical) : http://en.wikipedia.org/wiki/ISM_band**
2. **BeagleBone: http://beagleboard.org/**
3. **Blog about the project : http://blog.deadmeat.dk**
4. **Telit-868 : http://www.telit.com**

Appendix M

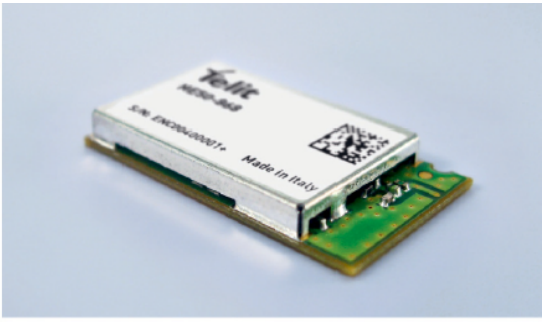
Telit NE50 Datablad

License-Free System
for Frequencies <1 GHz

Embedded

NE50-868
RF modules

38.4 Kbps - 25 mW



RoHS and WEEE Compliant

Available LGA Format

Embedded Low Power Mesh Stack

Ultra Low Power Consumption+Std-by Mode

Integrated Digital TTL I/Os and Analog Inputs

DOTA - Upgradeable and Configurable Over-the-Air

Telit NE50-868 RF modules are based on Mesh network concept in the license-free 868 MHz ISM band. With adjustable output power from 5 mW to 25 mW NE50-868 modules can reach up to 1500 m in LOS.

Advanced proprietary embedded low power mesh stack allows efficient power management on both end nodes and routers, network latency defined on the system requirements by setting different synchronous network time, data rate or message format, connecting up to 100 end nodes per router in a cluster tree architecture that enables scalability.

Low power mesh stack is designed for battery powered sensor networks that can be built automatically making it easily to integrated, thus reducing development time and cost for applications in building automation, metering (water, gas, electric), irrigation, tracking, lightning and access control. Telit NE50-868 is pin to pin compatible with ZE Family (Zigbee), ME Family (Wireless M-Bus) and LE Family (Telit Star Network).

Performance
- Range: Up to 1500 m (Ext antenna)
- Output Power: adjustable from 5 mW to 25 mW
- Serial Data Rate: Up to 115.2 Kbps
- Radio Data Rate: 38.4 Kbps
- Sensitivity (PER=1%): -103 dBm@38.4 Kbps

Power Requirements
- Power Supply: 2 to 3.6 V
- Board Consumption at 25 mW:
  Rx: < 26 mA
  Tx: < 45 mA
  Std-by:
  - With clock running (internal timer running) < 3µA

Physical Properties
- Board Format:
  Rectangular 26 x 15 mm, height 3 mm
- Extended temperature: -40°C to +85°C

Mesh features
- Ultra low power end point
- Up to 10 hops on the network
- Up to 10 000 device in the network
- Cluster tree
- Auto-association
- Auto-repair
- Configurable network period and synchronous part

Networking
- Frequency: 868 - 870 MHz
- Channels: 5
- Modulation: GFSK
- Serial Interface:
  RS232 TTL (Tx, Rx, Cts, Rts)
- Hayes Mode: Yes
- Download Over-the-Air: Yes
- Mesh Network: Yes
- I/O Copy: Yes
- Listen Before Talk: Yes

Order-No.

Please contact your Telit representative for order codes and further information.

Making machines talk.®

Bibliography

[1] Gate21 CO2-neutralbyrumsarmatur.Jacob Lundgaard.
http://www.gate21.dk/Projekter/CO2-neutralbyrumsarmatur/

[2] ARM Processors Andy Phillips
http://www.arm.com/products/processors/

[3] Gordon Moore - Moore's Law Inspires Intel Innovation
http://www.intel.com/content/www/us/en/silicon-innovations/moores-law-technology.html

[4] From:Mike Jones<mbj@microsoft.com> Sent: Sunday, December 07, 1997 6:47 PM. Subject: What really happened on Mars?
http://research.microsoft.com/en-us/um/people/mbj/mars_pathfinder/Authoritative_Account.html
http://marsrovers.jpl.nasa.gov/home/index.html

[5] Unix time,Defined as the number of seconds that have elapsed since midnight Coordinated Universal Time (UTC), January 1, 1970
http://en.wikipedia.org/wiki/Unix_time
http://www.unixtimestamp.com/index.php

[6] Live Data from ICEGoat http://obuoy.datatransport.org
Polar Field Service : http://www.polarfield.com
Arctic Research Support and Logistics Services : http://polar.sri.com
polar research community : http://www.polarpower.org

[5] Perl 5 is a highly capable, feature-rich programming language with over 24 years of development. Perl 5 runs on over 100 platforms from portables to mainframes and is suitable for both rapid prototyping and large scale development projects. http://www.perl.org/

[6] Software Engineering (6th Edition) (Hardcover) Ian Sommerville
Addison Wesley; 6 edition ISBN-13: 978-0201398151
http://www.comp.lancs.ac.uk/computing/resources/IanS/SE6/index.html

[7] 829-1998 IEEE standard for software test documentation E-ISBN: 0-7381-1444-8 ISBN: 0-7381-1443-X
INSPEC Accession Number: 6258325

[8] 830-1998 IEEE recommended practice for software requirements specifications E-ISBN: 0-7381-0448-5 ISBN: 0-7381-0332-2

INSPEC Accession Number: 6146520

[9] Applying UML and Patterns Craig Larman,Prentice Hall PTR,ISBN 0-13-092569-1
Website: http://www.craiglarman.com/

[10] Test Af Software,Poul Staal Vinje Teknisk Forlag,ISBN 87-571-1344-0

[11] The Security Wheel, Network Security Technologies and Solutions (CCIE Professional Development Series), p. 18, Figure 1-6.-Cisco Press; 1 edition (March 30, 2008) ISBN-10: 1587052466-ISBN-13: 978-1587052460

[12] CO2 Duct Air Quality Sensor for Carbon Dioxide Measurement. `http://www.fuehlersysteme.dk/luftkvalitet/co2-duct-sensor.html`

[13] The ARM1176 processor features `http://www.arm.com/products/processors/classic/arm11/arm1176.php`

[14] ZHANGZHOU NHEOLIS TECHNOLOGY CO.,LTD `http://www.nheolis.com.cn/`

[15] ATmega640, low-power Atmel 8-bit AVR RISC-based microcontroller combines 64KB ISP flash memory, 8KB SRAM, 4KB EEPROM, 86 general purpose I/O lines, 32 general purpose working registers `http://www.atmel.com/devices/atmega640.aspx`

[16] MCU Wireless chipset for IEEE 802.15.4 and ZigBee® applications. It is a bundle of the Atmel AVR ATmega1280 and Atmel AT86RF212 radio for the regional 700/800/900MHz/2.4Ghz frequency bands. `http://www.atmel.com/products/microcontrollers/Wireless/bundles.aspx`

[17] Bruger venlige edb-systemer, 2. udgave 1999 Rolf Molich,Teknisk Forlag,ISBN 87-571-1647

[18] DD-WRT is a Linux based alternative OpenSource firmware suitable for a great variety of WLAN routers and embedded systems. `http://www.dd-wrt.com` Setup and Help: `http://www.dotkam.com/2008/10/02/configure-multiple-ssids-with-one-router/`

[19] Wireless Networking in the Developing World. `http://wndw.net/`

[20] Unge vinder pris for halvgeniale internetlamper `http://www.b.dk/tech/unge-vinder-pris-for-halvgeniale-internetlamper`