

AARHUS UNIVERSITET

4. SEMESTERPROJEKT GRUPPE 1
PROJEKTDOKUMENTATION

Spændingsregulator

201509249 - Caroline Møller Sørensen
201611140 - Sophia Amalie Mortensen
201505115 - Laurids Givskov Jørgensen
201505195 - Dennis Slot Larsen
201508333 - Søren Jensen
13114 - Jeppe Hansen

Vejleder
Emir Pasic

29. maj 2017

Indhold

Indhold	i
1 Introduktion	1
1.1 Problemformulering	1
1.2 Projektbeskrivelse	1
2 Termliste	3
3 Kravspecifikation	4
3.1 Systembeskrivelse	4
3.2 MoSCoW	4
3.3 Funktionelle krav	5
3.3.1 Beskrivelse af automatisk mode	5
3.3.2 Usecase Diagram	6
3.3.3 Aktør Beskrivelse	6
3.3.4 Usecase 1 - Start manuel styring	6
3.3.5 Usecase 2 - Stop manuel styring	7
3.3.6 Usecase 3a - Skift trin	7
3.3.7 Usecase 3b - Skift trin	8
3.4 Ikke funktionelle krav	8
3.4.1 Trintransformer	8
3.4.2 Belastning	8
3.4.3 Måleenhed	8
3.4.4 Kommunikation	9
4 Accepttestspezifikation	10
4.1 Funktionelle Krav	10
4.2 Ikke funktionelle krav	11
5 Arkitektur	13
5.1 Blok definitionsdiagram	13
5.2 Intern blok diagram	14
5.3 Allokéringsdiagram	17
5.4 Brugergrænseflade	18
5.4.1 Automatisk mode	18
5.4.2 Manuel mode	19
6 Foranalyse	20
6.1 Valg af transformer	20
6.2 Valg af styringsenhed	20
6.3 Overvejelser omkring måleenhederne	20

6.4	Simulering af distributionslinje	21
7	Kommunikationsprotokoller	22
7.1	TCP protokol	22
7.2	UART protokol	22
8	Design af distributionslinje, belastning og trinskifter	24
8.1	Distributionslinje	24
8.2	Belastning	25
8.2.1	Beregning for Belastning	25
8.2.2	Implementering af Belastning	26
8.3	Power factor	26
8.4	Trinskifter	28
8.5	Modultest	29
8.5.1	Spænding over belastning	29
8.5.2	Power factor	30
9	Design af Måleenhed	31
9.1	Hardware	31
9.1.1	Diagram	31
9.2	Software	33
9.2.1	Overordnet beskrivelse	33
9.2.2	Analog til digital konvertering	34
9.2.3	Fourier algoritme	37
9.2.4	Beregning af rms og power faktor	38
9.2.5	Total Harmonic Distortion	39
9.2.6	UART forbindelse til styringsenhed	40
9.2.7	Kalibrering af måleenhed.	41
9.3	Modultest	43
10	Design af Styringsenhed	45
10.1	Kontrolmodul	45
10.1.1	TCP kommunikation	45
10.1.2	Styring af trinskifteren	49
10.2	Brugergrænseflade	51
10.2.1	Automatisk mode	52
10.2.2	Manuel mode	53
10.3	Kommunikationsmodul	54
10.4	Modultest	57
10.4.1	Kontrolmodul	57
10.4.2	Brugergrænseflade	61
10.4.3	Kommunikationsmodul	62
11	Integrationstest	64
11.1	Integrationstest mellem Måleenhed og Styringsenhed	64
11.2	Samlet integrationstest	66
12	Accepttest	69
12.1	Funktionelle Krav	69
12.2	Ikke funktionelle krav	70
13	Metode	72

Kapitel 1

Introduktion

1.1 Problemformulering

Når belastningerne i et distributionssystem ændres, vil spændingsniveauet variere. Det er vigtigt, at spændingsniveauet holdes stabilt. Hvordan sikres dette?

1.2 Projektbeskrivelse

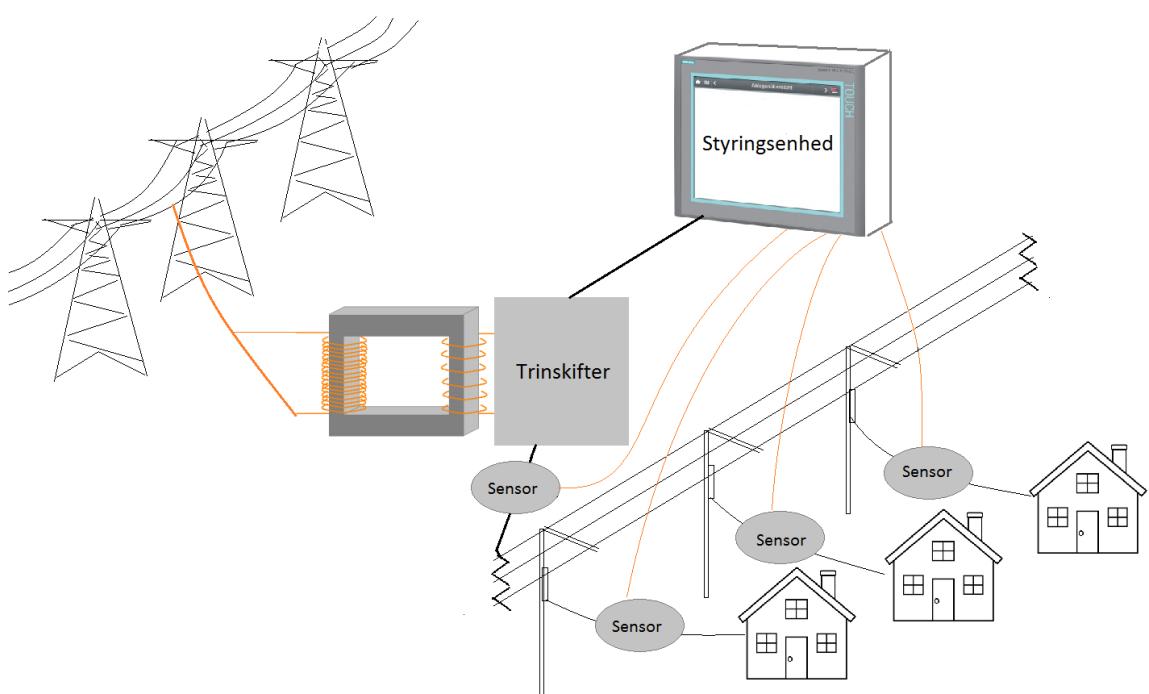
Formålet med dette projekt, er at opbygge et system, der simulerer det danske distributions-system. For energileverandører i Danmark er det et lovmæssigt krav, at spændingsforsyningen hos forbrugerne altid ligger på 230 volt $\pm 10\%$, og det ønskes at undersøge mulighederne for at opfylde dette.

I dette projekt vil fokus være på stykket fra distributionstransformer og ud til forbrugere. Systemet skal bestå af en trinskifter, en distributionslinje og to eller flere varierende belastninger. Det ønskes at måle strøm, spænding og power factor således, at spændingsregulatoren hele tiden kan holde spændingen på $\pm 10\%$, selvom belastningen ændres. Normalt måles disse værdier ved distributionstransformeren, men i dette projekt ønskes det at måle hos hver enkelt belastning. På den måde fås en bedre overvågning af systemet og bedre mulighed for at observere hvilken betydning, f.eks. belastningens afstand til distributionstransformeren har for spændingsniveauet.

Systemet skal have to indstillinger – en til manuelt valg af spændingsniveau og en til automatisk valg af passende spændingsniveau.

Det ønskes desuden at kunne måle frekvensindholdet i systemet for at kunne observere et eventuelt indhold af harmoniske. De harmoniske i systemet er højfrekvente og vil afsætte varme i transformerne og dermed forkorte deres levetid. Det er derfor relevant at kende til indholdet af disse.

Det er et krav, at målte værdier i systemet vises på en skærm.



Figur 1.1: Visuel fremvisning af system

Kapitel 2

Termliste

Term	Beskrivelse
Spændingsregulator	Samlet system
Trintransformer	Transformer med variabelt omsætningsforhold
Trinskifter	Omfatter trintransformer og relækredsløb
Styringsenhed	PLC, HMI og Arduino
Centralt	Ved trinskifter
Decentralt	Ved forbrugeren
Måleenhed	PSoC og tilhørende hardware

Tabel 2.1: Termbeskrivelse

Kapitel 3

Kravspecifikation

3.1 Systembeskrivelse

Systemet, der udvikles, har til opgave at regulere spændingsniveauet på en distributionslinje, afhængigt af målinger fra sensorer ved hver forbruger. I dette projekt er det ikke muligt at realisere, derfor udvikles et produkt, der kan simulere scenariet. Det simuleres ved at skalere spændingsniveauet fra standardniveauet på 230V ned til 4V på distributionssiden. Dette gør det muligt at arbejde med en 8-trins transformer med specifikationen 24V/8-0V.

I prototypen udvikles en impedans til simulering af en distributionslinje med en længde svarende til en typisk distributionslinje.

På distributionslinjen tilsluttes et antal belastninger, der skal illustrere husstande. Disse er designes således, at skalering passer med resten af systemet. Dette er rammen systemet skal arbejde indenfor.

En enkelt type måleenhed fremstilles, som placeres centralt ved hver belastning. Disse måleenheder kan måle spænding, strøm, power faktor og harmoniske. Systemets frekvens er 50Hz ligesom frekvensen på det danske elnet.

Data fra måleenhederne samles i en styringsenhed, der har til opgave at regulere spændingsniveauet, så det altid ligger på $4V \pm 10\%$ på distributionssiden ved at skifte trin på transformeren. Dette står trinskifteren for.

På styringsenheden er det muligt at observere de målte værdier på en touchskærm. Ved manuel styring er det på samme touchskærm, hvor der kan skiftes trin på transformeren.

Systemet er en simulering af et produkt, der kan løse problemet. Den overordnede struktur er tænkt sådan, at den kan skaleres op.

3.2 MoSCoW

- Systemet **skal** bestå af en trintransformer 24/0-8V
- Systemet **skal** måle spænding, strøm, power factor og harmoniske centralt og decentralt
- Systemet **skal** vise data på en skærm
- Systemet **skal** simulere en distributionslinje og flere forbrugere
- Systemet **skal** kunne reguleres manuelt
- Systemet **burde** kunne reguleres automatisk
- Systemet **kunne** have en log

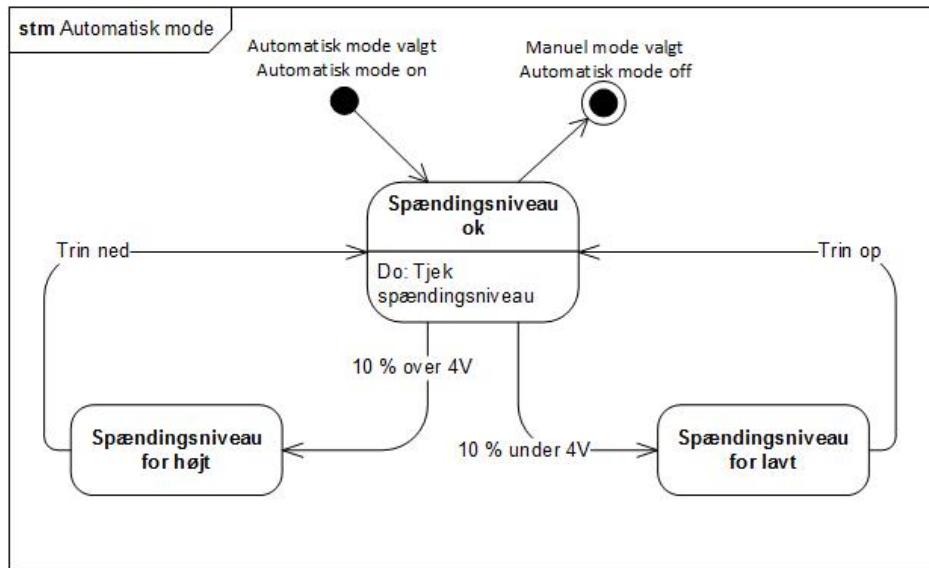
- Distributionslinjen **kunne** indeholde en decentral producent
- Systemet **vil ikke** fjerne harmoniske

3.3 Funktionelle krav

I dette afsnit beskrives de funktionelle krav for systemet. De dele, hvor en bruger interagerer med systemet er beskrevet med usecase diagrammer. Den automatiske del er beskrevet og vist vha. et STM diagram.

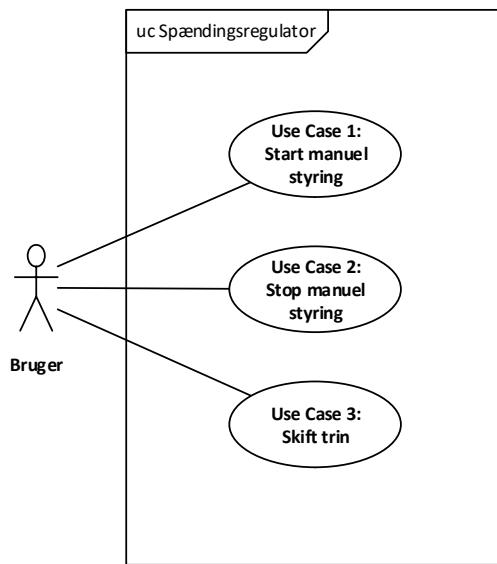
3.3.1 Beskrivelse af automatisk mode

Når spændingsregulatoren er i automatisk mode, kontrolleres spænding ved forbrugerne. Hvis den spænding er for høj eller lav iht. de 4V skiftes der et trin op eller et trin ned.



Figur 3.1: Beskrivelse af automatisk mode

3.3.2 Usecase Diagram



Figur 3.2: Usecase Diagram

Systemet indholder tre usecases, der alle er initieret af brugeren. Den automatiske del af systemet er beskrevet i afsnit 3.3.1.

3.3.3 Aktør Beskrivelse

Brugeren er den primær aktør. En sikkerhedsgodkendt operatør der kan betjene systemet.

3.3.4 Usecase 1 - Start manuel styring

Navn:	UC1 - Start manuel styring
Mål:	At sætte systemet i manuel mode
Initiering:	Initieres af brugeren.
Aktører:	Brugeren (Primær)
Samtidige forekomster:	1
Forudsætninger:	At systemet er funktionelt og i automatisk mode
Resultat:	Systemet er i manuel mode
Hovedscenariet:	
1	Brugeren trykker Manuel styring på skærmen.
2	Systemet skifter til Manuel mode.
3	Systemet aktivere manuel skærm.

Tabel 3.1: Fully dressed use case for UC1 - Start manuel styring

3.3.5 Usecase 2 - Stop manuel styring

Navn:	UC2 - Stop manuel styring
Mål:	At sætte systemet i automatisk mode
Initiering:	Initieres af brugeren.
Aktører:	Brugeren (Primær)
Samtidige forekomster:	1
Forudsætninger:	At systemet er funktionelt og i manuel mode
Resultat:	Systemet er i automatisk mode
Hovedscenariet:	
1	Brugeren trykker Automatisk styring på skærmen.
2	Systemet skifter til Automatisk mode.
3	Systemet aktivere automatisk skærm.

Tabel 3.2: Fully dressed use case for UC2 - Stop manuel styring

3.3.6 Usecase 3a - Skift trin

Navn:	UC3a - Skift trin op
Mål:	At skifte et trin op på transformeren
Initiering:	Initieres af brugeren.
Aktører:	Brugeren (Primær)
Samtidige forekomster:	1
Forudsætninger:	At systemet er funktionelt og i manuel mode
Resultat:	Transformerens trin er skiftet et trin op
Hovedscenariet:	
1	Brugeren vælger Trin Op på skærmen.
2	Systemet skifter et trin op på transformeren.
3	Aktuelt trin vises på skærmen.
4	Måleværdier opdateres på skærmen.

Tabel 3.3: Fully dressed use case for UC3 - Skift trin

3.3.7 Usecase 3b - Skift trin

Navn:	UC3b - Skift trin ned
Mål:	At skifte et trin ned på transformeren
Initiering:	Initieres af brugeren.
Aktører:	Brugeren (Primær)
Samtidige forekomster:	1
Forudsætninger:	At systemet er funktionelt og i manuel mode
Resultat:	Transformerens trin er skiftet et trin ned
Hovedscenariet:	
1	Brugeren vælger Trin Ned på skærmen.
2	Systemet skifter et trin ned på transformeren.
3	Aktuelt trin vises på skærmen.
4	Måleværdier opdateres på skærmen.

Tabel 3.4: Fully dressed use case for UC3 - Skift trin

3.4 Ikke funktionelle krav

3.4.1 Trintransformer

1. Maks belastning af spændingsregulatoren er 20VA
2. Nominel spænding på primær siden er 24VAC
3. Nominel spænding på sekundær siden er 4 , 5 eller 6 afhængigt af trin VAC
4. Skal minimum kunne leve 500mA

3.4.2 Belastning

1. Modstandsværdi på 54Ω giver spændingsfald på 10%, når spændingen fra regulatoren er 4V.

3.4.3 Måleenhed

Kravene til målingerne i Måleenheden, er opgivet i en procent, da det ikke er relevant for vores "proof of concept" at gå dybere ned i målenøjagtighed og præcision.

1. Måle spændingen ved trinskifteren og forbrugerne mellem 0 og 8 Vrms
2. Måle spændingen med en præcision på $\pm 5\%$
3. Måle strømmen ved trinskifteren og forbrugerne mellem 0 og 500mA
4. Måle strømmen med en præcision på $\pm 5\%$
5. Måle og beregne power factor med en præcision på $\pm 5\%$
6. Beregne THD med en præcision på $\pm 5\%$

3.4.4 Kommunikation

Kravene til kommunikationen gælder for hele kommunikationsvejen fra sensorer til brugergrænseflade. Disse krav er sat jf. "proof of concept" hvor fejlhåndtering ikke har prioritet.

1. Forsinkelsen på brugergrænsefladen ifht. ændringer i målte værdier må ikke overstige 2,5 sekund.
2. 95% af alle sendte data skal være korrekte, og uden forstyrrelser.

Kapitel 4

Accepttestspezifikation

4.1 Funktionelle Krav

UC1	Handling	Forventet resultat	Resultat	OK
Start manuel styring	Bruger vælger Manuel Mode	På skærmen vises Manuel Mode med mulighed for valg af trin		

UC2	Handling	Forventet resultat	Resultat	OK
Stop manuel styring	Bruger vælger Automatisk Mode	Skærmen viser Automatisk Mode, hvor trinknapperne er deaktiverede		

UC3a	Handling	Forventet resultat	Resultat	OK
Skift trin op	Systemet er i Manuel Mode. Bruger vælger Trin Op på skærmen	Systemet skifter trin op på transformeren og skærmen opdateres med nye værdier.		

UC3b	Handling	Forventet resultat	Resultat	OK
Skift trin ned	Systemet er i Manuel Mode. Bruger vælger Trin Ned på skærmen	Systemet skifter trin ned på transformeren og skærmen opdateres med nye værdier.		

4.2 Ikke funktionelle krav

Trintransformer	Handling	Forventet resultat	Resultat	OK
Nominel spænding på primærsiden er 24VAC	Spændingen på primærsiden måles.	Den målte værdi er 24VAC.		
Nominel spænding på sekundær side er 4, 5 eller 6 VAC afhængig af trin	Spændingen på sekundær sidesiden måles for hhv. trin 4, 5 og 6.	De målte værdier er 4, 5 og 6V		
Skal minimum kunne leve 500mA .	Strømmen på sekundær sidesiden måles.	Transformeren kan leve over 500mA		

Belastning	Handling	Forventet resultat	Resultat	OK
Modstandsværdi på 54Ω giver spændingsfald på 10%, når spændingen fra regulatoren er 4V.	Den givne modstand indsættes som belastning, og spændingen herover måles.	Spændingen over belastningen måles til 3,6V.		

Måleenhed	Handling	Forventet resultat	Resultat	OK
Måle spændingen ved trinskifteren og forbrugerne mellem 0 og 8 Vrms	Måleenheden testes med spændinger fra 0 til 8Vrms, i intervaller af 500mV.	Korrekt spændingsmåling i hele intervallet.		
Måle spændingen med en præcision på $\pm 5\%$	Måleenheden påtrykkes en spænding på 3,5Vrms. Der laves herefter ti målinger	Gennemsnits afvielsen forventes at være under $\pm 5\%$.		
Måle strømmen ved trinskifteren og forbrugerne mellem 0 og 500mA	Måleenheden testes med strømme fra 0 til 500mA i intervaller af 50mA	Korrekt strømmåling i hele intervallet.		
Måle strømmen med en præcision på $\pm 5\%$	Måleenheden påtrykkes en spænding på 300mVrms (Svarrende til 300mA), der laves herefter ti målinger	Gennemsnits afvielsen forventes at være under $\pm 5\%$		

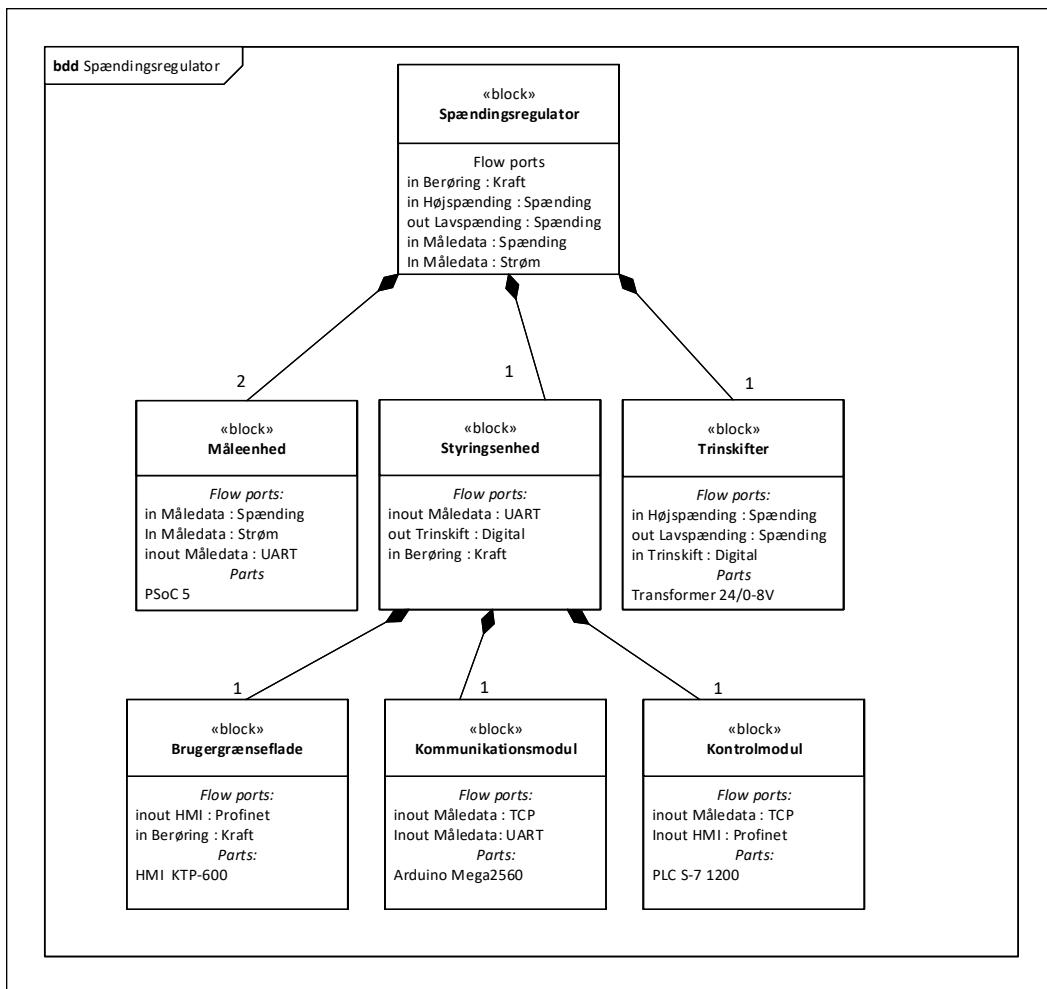
Måle og beregne power factor med en præcision på $\pm 5\%$	Måleenheten mäter power factor over en belastning på distributionslinjen, der sammenlignes med beregnet power factor	Afvigelsen forventes at være under $\pm 5\%$		
Beregne THD med en præcision på $\pm 5\%$	Måleenheten påtrykkes en firkantsignal med 1V amplituder og 1V offset. Der sammenlignes med beregnet THD for firkantsignal	Afvigelsen forventes at være under $\pm 5\%$		

Kapitel 5

Arkitektur

5.1 Blok definitionsdiagram

Et BDD for spændingsregulator ses på figur 5.1. På diagrammet ses de overordnede blokke, spændingsregulator består af. En beskrivelse af hver blok kan læses under figur 5.1.



Figur 5.1: BDD Spændingsregulator

Måleenhed står for at måle spænding, strøm og faseforskydningen herimellem. Ligeledes skal denne kunne måle indholdet af harmoniske frekvenser. Den skal bestå af hardware til

måling af de nævnte parametre og en PSoC. På enheden skal behandlingen af rådataet også ligge, så dette kan formidles til Styringsenheden.

Styringsenhed har til opgave at styre trinskifteren ud fra de data den får fra målenehederne. Den består af en PLC, der skal kommunikere med brugergrænsefladen, så en bruger kan følge med i data fra Måleenhederne.

Brugergrænsefladen står for at formidle måledata til brugeren gennem en skærm, men det er også her at brugeren skal kunne interagere med systemet i manuel tilstand.

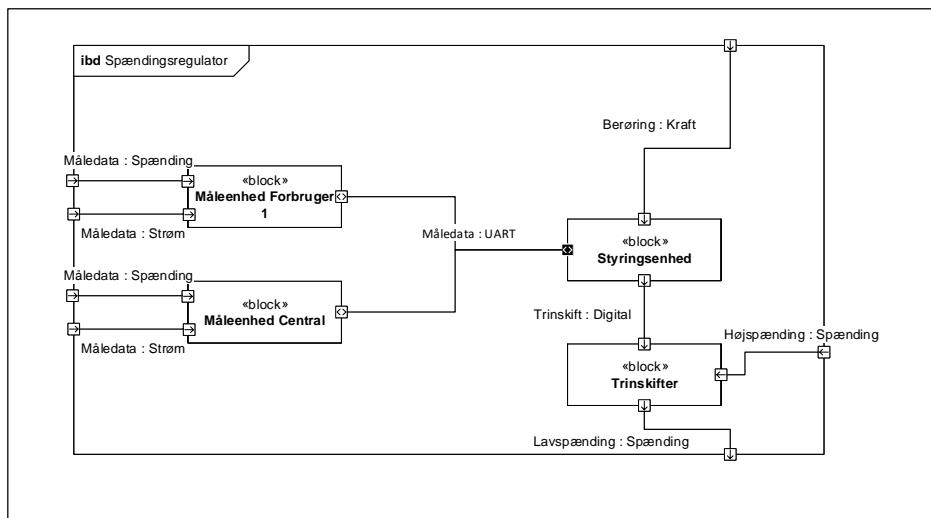
Kommunikationsmodul skaber en kommunikation fra Måleenhedens PSOC til Styringsenhedens PLC.

Kontrolmodul laves på en PLC, til at styrer trinene på trinskifteren.

Trinskifter er en enhed der kan skifte trin på en transformer ud fra et signal fra styringsenheden. Den skal altså bestå af et relæ for hvert trin, der kan kontrolleres af styringsenheden.

5.2 Intern blok diagram

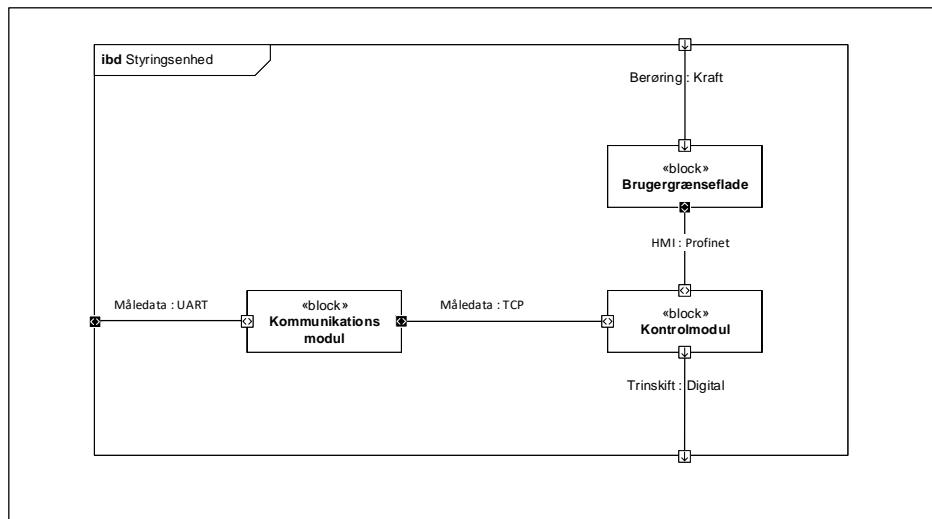
På figur 5.2 og figur 5.3 ses IBD for henholdsvis Spændingsregulator og Styringsenhed. På diagrammerne ses de interne forbindelser i systemet. Under figurerne er tilhørende signalbeskrivelser, se tabel 5.1 og tabel 5.2, der uddyber diagrammerne nærmere.



Figur 5.2: IBD for Spændingsregulator

Blok	Navn	Type	Signal	Beskrivelse
Måleenhed	Måledata	Spænding	In	Måledata er spændingsniveauet på distributionslinjen.
	Måledata	Strøm	In	Måledata er strømniveauet på distributionslinjen.
	Måledata	UART	InOut	UART forbindelse til Styringsenhed
Styringsenhed	Måledata	UART	Inout	UART forbindelse til Måleenhed
	Berøring	Kraft	In	Tryk på Brugergrænseflade
	Trinskift	Digital	Out	Trinskift er en digital kommando til Trinskifter
Trinskifter	Trinskift	Digital	In	Trinskift er en digital kommando fra Styringsenhed.
	Højspænding	Spænding	In	Er spændingen på højspændingssiden af transformeren.
	Lavspænding	Spænding	Out	Er spændingen på lavspændingssiden af transformeren.

Tabel 5.1: Signalbeskrivelse for Spændingsregulator



Figur 5.3: IBD for Styringsenhed

Blok	Navn	Type	Signal	Beskrivelse
Brugergrænseflade	Berøring	Kraft	In	Tryk på Brugergrænseflade
	HMI	Profinet	InOut	Forbindelse til Kontrolmodul
Kontrolmodul	HMI	Profinet	InOut	Forbindelse til Brugergrænseflade
	Trinskift	Digital	Out	Trinskift er en digital kommando til Trinskifter
	Måledata	TCP	InOut	TCP forbindelse til Kommunikationsmodul
Kommunikationsmodul	Måledata	TCP	InOut	TCP forbindelse til Kontrolmodul
	Måledata	UART	InOut	UART forbindelse til Måleenhed

Tabel 5.2: Signalbeskrivelse for Styringsenhed

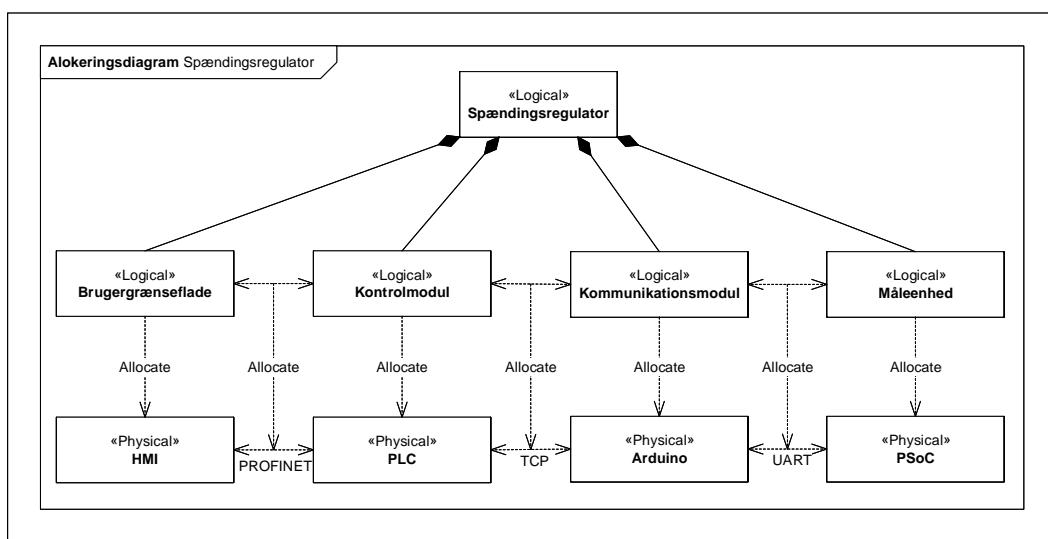
5.3 Allokéringsdiagram

På figur 5.4 ses allokéringsdiagram for spændningsregulatoren. Diagrammet er lavet for at danne overblik over softwaren, derfor er analog moduler undladt. Diagrammet viser hvilke platforme de logiske blokke skal laves på, og hvordan kommunikationen er mellem blokkene.

1. Brugergrænsefladen allokeres på en HMI skærm.
2. Kontrolmodulet allokeres på en PLC.
3. Kommunikationsmodulet allokeres på en Arduino.
4. Måleenhederne allokeres på PSoCs.

Kommunikationen mellem blokkene er allokertet på tre forskellige protokoller, se Kapitel 7 for uddybning af disse.

1. Mellem HMI og PLC anvendes Siemens standard PROFIBUS
2. Mellem PLC og Arduino anvendes en TCP-protokol
3. Mellem Arduino og PSOC anvendes en UART-protokol

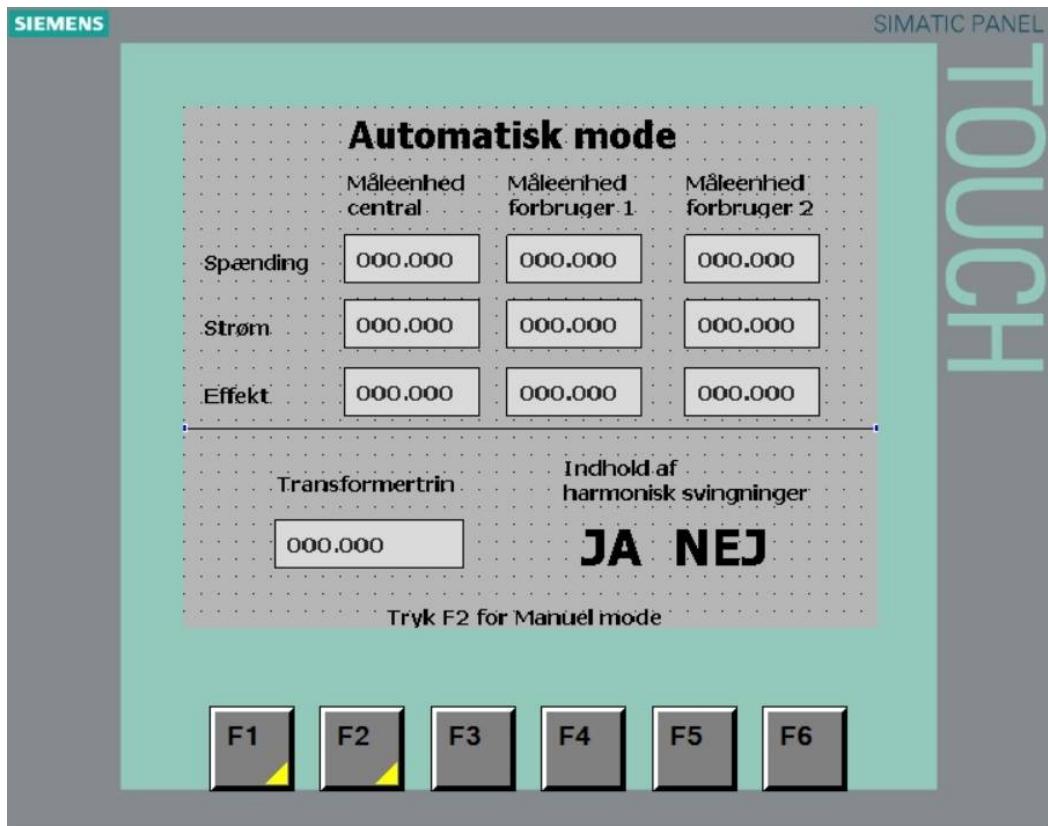


Figur 5.4: Alokéringsdiagram for spændingsregulator

5.4 Brugergrænseflade

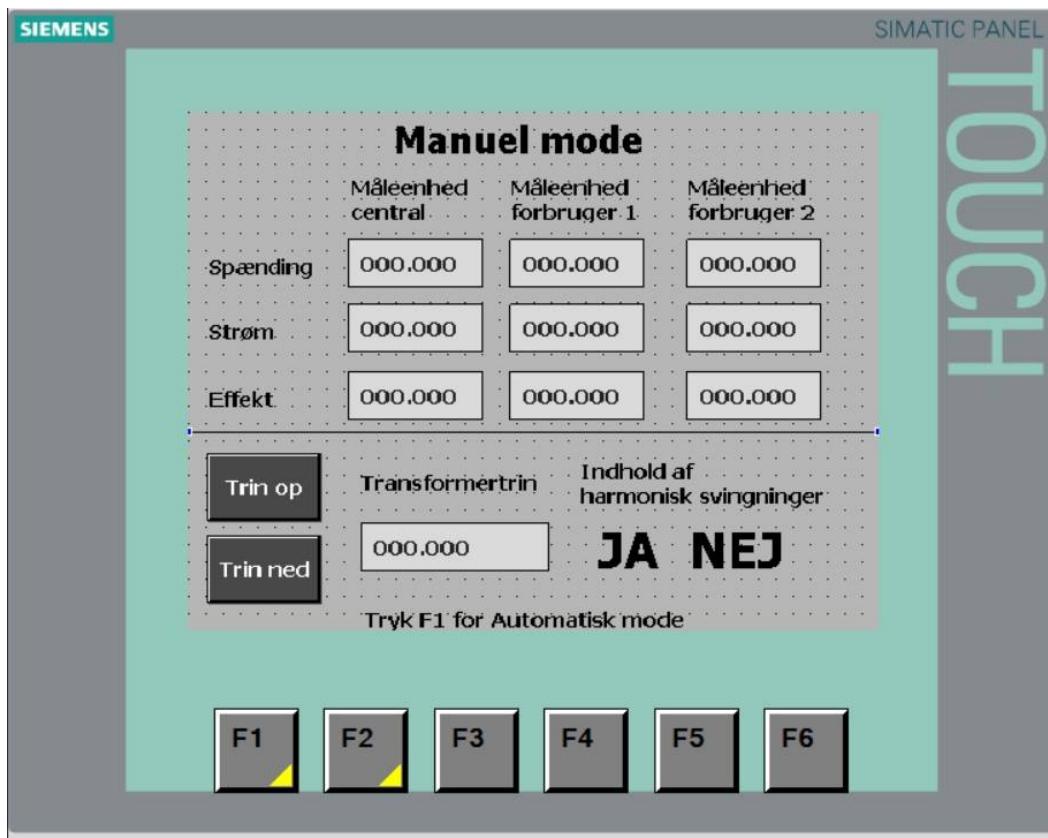
Brugergrænsefladerne der ses på figur 5.5 og 5.6 er udkast designet tidligt i processen før noget funktionalitet blev udviklet. Det er altså dem der er blevet brugt som udgangspunkt for det videre design. Det endelige design af brugergrænsefladerne kan ses i afsnit 10.2 Brugergrænseflade under Design.

5.4.1 Automatisk mode



Figur 5.5: HMI Automatisk mode

5.4.2 Manuel mode



Figur 5.6: HMI Manuel mode

Kapitel 6

Foranalyse

6.1 Valg af transformer

Projektgruppens vejleder udleverede to transformere. Den ene er der ingen mærkeplade og ledningerne er alle samme farve. Den anden har mærkeplade og forskellige farvet ledninger. De to transformere blev begge undersøgt i laboratoriet, med en 18VAC på primærsiden, derefter måltes spændingen på de forskellige trin. Transformeren uden mærkeplade hoppede små skridt fra ca 3 - 4.5V, hvor transformeren med mærkeplade havde skridt på 1 V fra 0 til 8V. Gruppen blev enige om at det var ligeegyldigt om det var små eller store skridt transformeren hoppede, bare prototypen blev lavet så den passede til transformeren. Det blev derfor besluttet at anvende transformeren med mærkeplade og forskellige farvet ledninger.

6.2 Valg af styringsenhed

Det er besluttet at lave Styringsenhedens kontroldelen på en Programmable Logic Controller (PLC), dette gøres for at realisere, hvordan det ville laves i virkeligheden. Alternativt kunne det laves på en PSOC eller Arduino, men da der på dette semester er undervisning i PLC styring virker det derfor oplagt. I samme fag undervises også i Human Machine Interface (HMI), så det var oplagt at anvende til brugergrænsefladen. Selve kommunikationsdelen har krævet flere overvejelser, hvor både en PSoC og en Arduino har været ind i billedet. Dette skulle at PLC'en kun har et frit RJ-45 stik, som der var brug for en form for switch for at kunne tilslutte flere måleenheder. Valget faldt i første omgang på en PSoC med tilhørende Ethernet shield, da PSoC blev brugt til måleenheden. Undervæjs blev skiftet over til en Arduino, fordi den er nemmere at arbejde med indenfor Ethernet kommunikation.

6.3 Overvejelser omkring måleenhederne

Det var først tænkt at der kunne laves eller købes nogle sensorer som skulle tilsluttes PLC'en. PLC'en kan dog ikke måle hurtigt nok på AC forbindelser til at få et reelt billede af signalet. Så det kan undersøges for harmoniske svingninger. Ved at sample et signal på en PSOC, med en hurtigere samplefrekvens, kan der udregnes fourier. Fourier kan anvendes til at udregne størrelsen på et signal ved bestemte frekvenser, og derved kan systemet undersøges for harmoniske. Der findes en måde til at angive hvor stort et indhold af harmoniske, der er i et forsyningssystem, som kaldes Total Harmonic Distortion (THD). Dette er et tal i procent, som angiver hvor stort indholdet af harmoniske er. Da beregningen af dette er forholdsvis simpelt, vil denne metode blive brugt til at analyserer systemets indhold af harmoniske.

6.4 Simulering af distributionslinje

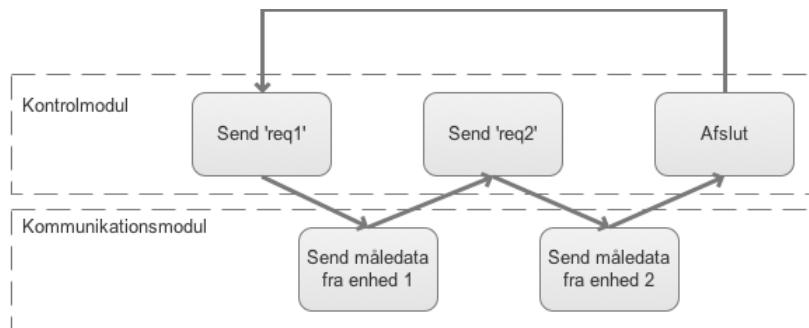
For at opnå en virkelighedstro simulering af en distributionslinje blev der i starten af projektet taget kontakt til energiselskabet Eniig. Herfra modtog projektgruppen data på et udsnit af et distributionsnet. Forinden var der i samarbejde med vejleder fundet kabedata på nkt cables hjemmeside. Afstandene mellem distributionstransformer og belastninger i Eniigs data er forholdsvis korte (100-500 meter). Med de fundne kabedata vil der ikke være modstand- og spolevirkning, der vil påvirke spændingen i simuleringen. Velvidende at det ikke stemmer overens med virkeligheden, valgtes det derfor at simulere en distributionslinje på 60 kilometer.

Kapitel 7

Kommunikationsprotokoller

7.1 TCP protokol

Mellem kommunikationsmodulet og kontrolmodulet er anvendt en TCP protokol. TCP er valgt, da systemet ikke kræver hurtig, men derimod en meget pålidelig datatransmission. PLC'en anvender stikstandarden RJ-45 og har mulighed for at kommunikere med hastighederne 10/100 Mb/s. Komponenterne i systemet tillader 100 Mb/s. PLC'en er opsat som client, der forespørger data fra Arduinoen, som svarer med det forespurgte data. Kommunikation over TCP forbindelsen anvender protokollen vist på figur 7.1.



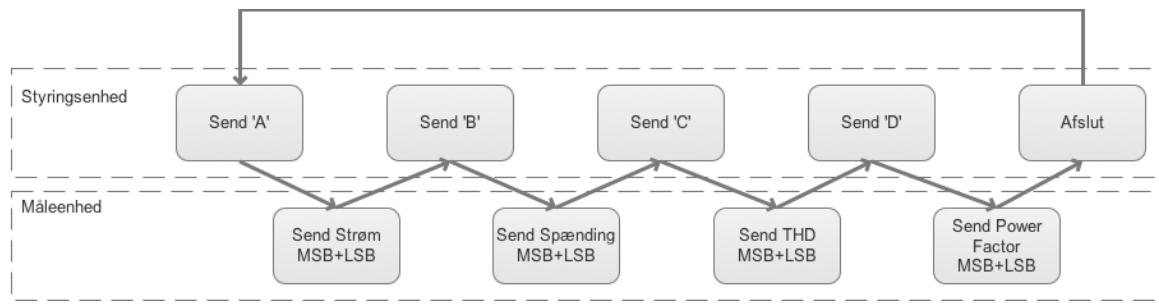
Figur 7.1: Protokol for kommunikation mellem kommunikationsmodul og kontrolmodul

Måledata består af otte bytes i sæt af to bytes. De fire sæt dækker over måleværdier for henholdsvis, strøm, spænding, THD og powerfactor. Yderligere information om TCP kommunikationen kan findes i afsnit 10.1.1 TCP kommunikation og 10.3 Kommunikationsmodul.

7.2 UART protokol

UART protokollen anvendes mellem Måleenheden og Styringsenheden. Opsætning af UART forbindelse overholder indstillingerne i Tabel 7.1. Data som transmitteres over UART forbindelsen sendes i pakker af 8-bit. Da værdier for strøm, spænding, power factor og THD er uint16 værdier, skal der for hver værdi sendes to pakker, henholdsvis MSB og LSB.

Kommunikation over UART forbindelsen skal overholde følgende protokol, se Figur 7.2.



Figur 7.2: Protokol for kommunikation mellem Måleenhed og Styringsenhed

Tabel 7.1: UART protokol

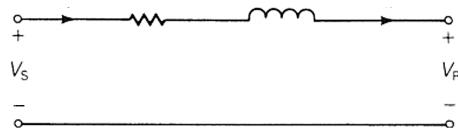
	Indstilling	Bemærkning
Mode	Full UART (Rx+Tx)	
BaudRate	9600	
DataBits	8	
ParityType	None	
Stopbits	1	
Flowcontrol	None	

Kapitel 8

Design af distributionslinje, belastning og trinskifter

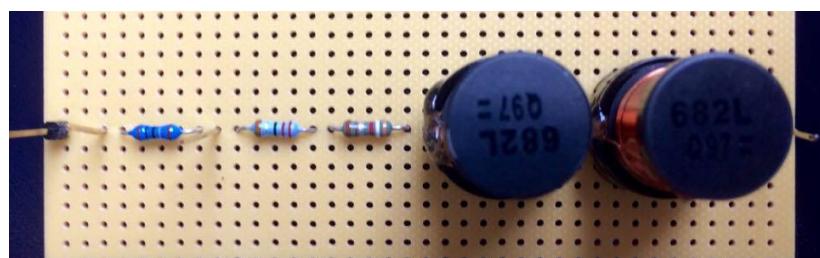
8.1 Distributionslinje

På baggrund af foranalysen 6.4 er længden af distributionslinjen, der ønskes simuleret, valgt til 60 km. Ud fra databladet for den valgte kabeltype ses at der vil være $0,1 \Omega /km$ og $0,219 \text{ mH}/\text{km}$, se bilag B1. For at kunne simulere disse værdier er opbygget et kredsløb med $6,2 \Omega$ modstand i serie med en $13,6 \text{ mH}$ spole. Dette stemmer overens med teorien for korte transmissionslinjer, hvor kun modstand og spolevirkning indgår i modellen. Modellen ses på figur 8.1 og gælder for linjer op til 80 km.



Figur 8.1: Model for kort transmissionslinje

Distributionslinjen er implementeret på et printkort, hvorpå to spoler og to modstande er monteret i serie for at opnå de ønskede værdier for 'kablet'. Med en kabellængde på 60 km burde modstanden være 6Ω og spolen $13,14 \text{ mH}$. På baggrund af tilgængelighed på skolens lager blev den realiserede modstand $6,2 \Omega$ og den realiserede spole $13,6 \text{ mH}$. Der er desuden monteret en 1Ω modstand i serie. Denne giver mulighed for at placere en Måleenhed til at overvåge denne del af systemet. Det færdige print til simulering af distributionslinje ses på figur 8.2

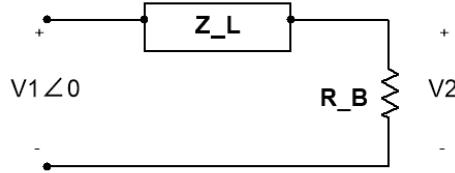


Figur 8.2: Simulering af distributionslinje

8.2 Belastning

8.2.1 Beregning for Belastning

Med denne distributionslinje og med trinskifteren, der står på 4V trinnet, kan det beregnes hvilken belastning, der vil medføre, at spændingen hos forbrugeren falder til under 10 % af 4V. Kredsløbet og beregningen ses nedenfor.



Figur 8.3: Kredsløb til beregning af belastning

Først bruges spændingsdelerformlen

$$V2 = V1 \cdot \frac{R_B}{Z_L + R_B} \quad (8.1)$$

$$0,9 \cdot |V1| = |V1 \cdot \frac{R_B}{Z_L + R_B}| = V1 \cdot \frac{R_B}{R_L + jX_L + R_B} \quad (8.2)$$

$$0,9 = \left| \frac{R_B}{R_L + jX_L + R_B} \right| \quad (8.3)$$

$$0,9 \cdot |R_L + jX_L + R_B| = |R_B| \quad (8.4)$$

$$0,9 \cdot \sqrt{(R_L + R_B)^2 + X_L^2} = R_B \quad (8.5)$$

$$(R_L + R_B)^2 + X_L^2 = \frac{R_B^2}{0,81} \quad (8.6)$$

$$R_L^2 + R_B^2 + 2 \cdot R_L \cdot R_B + X_L^2 = \frac{R_B^2}{0,81} \quad (8.7)$$

$$R_B^2 \cdot \left(1 - \frac{1}{0,81}\right) + 2 \cdot R_L \cdot R_B + X_L^2 = 0 \quad (8.8)$$

Værdier indsættes og herved fås:

$$R_B^2 \cdot (-0,24) + 12,4\Omega \cdot R_B + 18,23\Omega = 0 \quad (8.9)$$

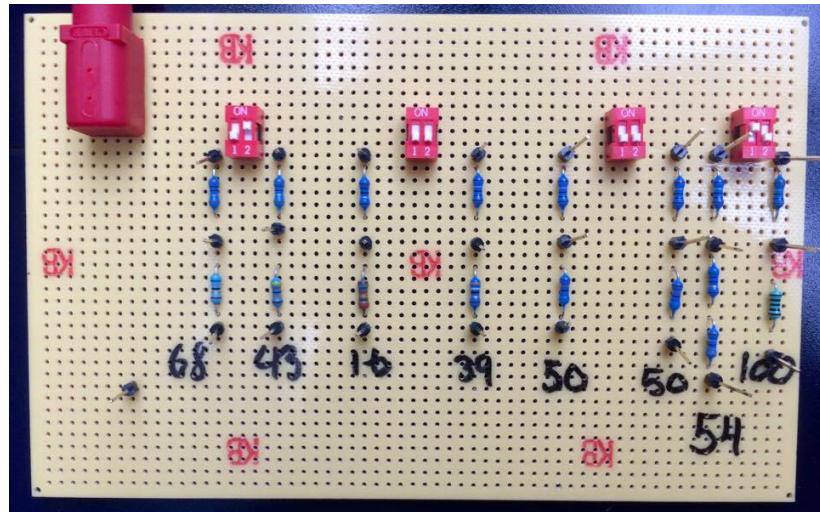
Andengrads ligning løses og derved findes den modstandsværdi, der vil give et spændingsfald på 10%.

$$R_B = \frac{-12,4 \pm \sqrt{153,76 - 4 \cdot (-0,24) \cdot 18,23}}{-0,24 \cdot 2} = \frac{-12,4 \pm 13,1}{-0,47} = 54,3\Omega \quad (8.10)$$

Denne modstandsværdi i det viste kredsløb og med den valgte distributionslinje vil resultere i at spændingen falder under det ønskede niveau som er 4V. Der tages derfor udgangspunkt i denne værdi, men efterfølgende vil belastninger bestemmes ud fra simuleringer i værktøjet Multisim. Simulering der understøtter beregningerne ses på figur 8.6

8.2.2 Implementering af Belastning

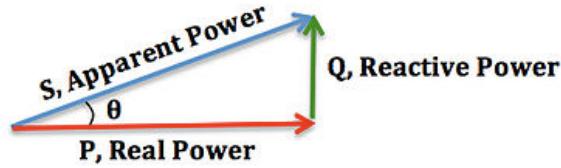
Til implementering af belastninger/forbrugere er der på baggrund af foregående beregninger valgt modstande i intervallet 16Ω til 100Ω . Belastninger er placeret i parallel og til hver belastning hører en kontakt, således der let kan skiftes mellem forskellige værdier. Der er desuden monteret pin til forbindelse til distributionslinje og bananstik til forbindelse til 0V på transformeren. Yderligere er der for hver belastning monteret en 1Ω modstand således Måleenheden kan overvåge tilstanden hos hver forbruger. Det færdige print med belastninger ses på figur 8.4



Figur 8.4: Færdigt print med belastninger/forbrugere

8.3 Power factor

For yderligere monitorering ønskes det også at undersøge power factor i systemet. Når effekt transportereres gennem en distributionslinje består det af både aktiv og reaktiv effekt, og power factor fortæller ratioen mellem aktiv og tilsyneladende effekt. Denne parameter fortæller hvor meget effekt, der kan tages ud af systemet og udnyttes hos en belastning/forbruger. Da belastningerne i dette system er rent ohmske, og der kun er spolevirkning fra 'kablen, forventes der en power factor tæt på 1, men som vil være lagging netop pga, spolen. Forholdet mellem aktiv, reaktiv og tilsyneladende effekt kan ses på figur 8.5



Figur 8.5: Effekttrekant

På baggrund af dette kan den aktuelle power factor med de valgte kabelparametre og en belastning på 54Ω nu beregnes. Beregninger ses nedenfor.

$$Z_R = 54\Omega \quad (8.11)$$

$$R_L = 6,2\Omega \quad (8.12)$$

$$X_L = 2 * \pi * 50Hz * 13,6mH = 4,27\Omega \quad (8.13)$$

$$Z_D = (6,2 + 4,27j)\Omega \quad (8.14)$$

$$Z_L = Z_R + Z_D = (60,2 + 4,27j)\Omega \quad (8.15)$$

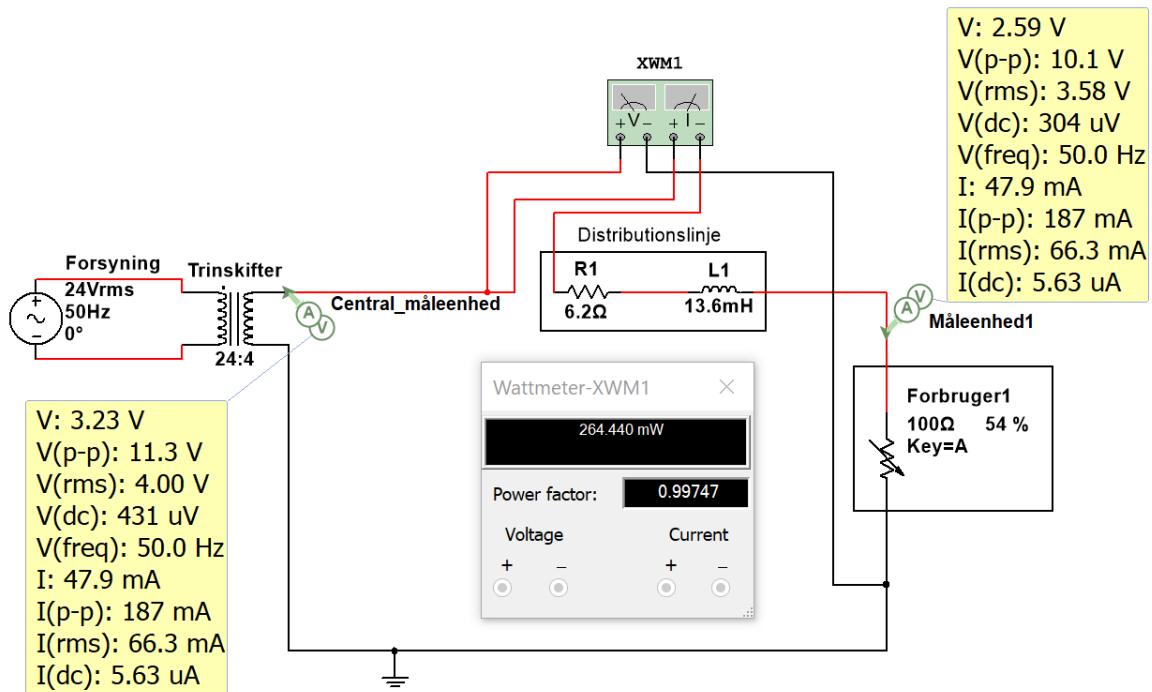
$$I = \frac{4V}{|Z_L|} = 0,07A \quad (8.16)$$

$$S_L = I^2 \cdot Z_L = (0,26 + 0,02j)VA \quad (8.17)$$

$$P = 0,26W \quad (8.18)$$

$$pf = \frac{P}{|S_L|} = 0,996 \quad (8.19)$$

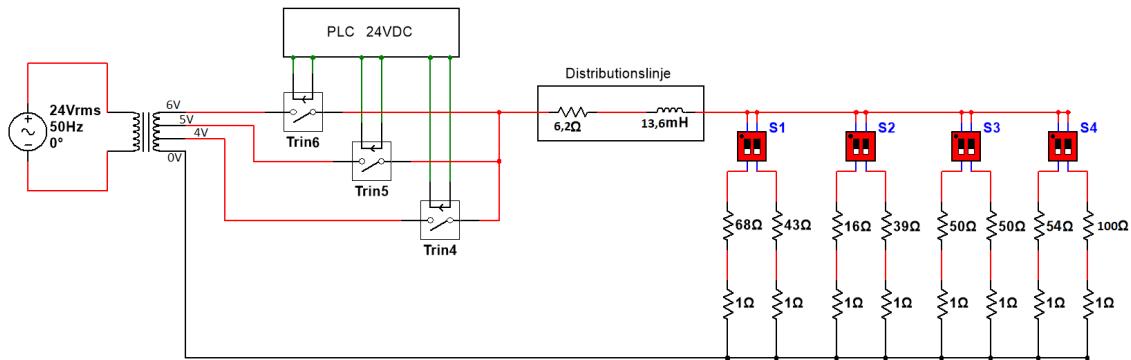
For at tjekke og understøtte disse beregninger er der lavet en simulering af systemet i Multisim. Denne simulering giver en power factor på 0,997, hvilket stemmer fint overens med forventningen fra beregningen. Simuleringen ses på figur 8.6



Figur 8.6: Simulering, der viser power factor

8.4 Trinskifter

For at kunne skifte trin og dermed spændingsniveau på transformeren er der opbygget et kredsløb med tre kontaktrelæer - et for hvert muligt trin. Det er et krav, at relæerne skal kunne styres både automatisk og manuelt fra en PLC. Udgangssignalet fra PLC'en er 24VDC, så dette skal relæerne altså kunne holde til. På værkstedet var kun en type relæ der kan holde til 24VDC styresignal, og derfor faldt valget på disse Hengstler 468 relæer, se bilag B4 for datablad. På figuren nedenfor ses tegning af relækredsløbet samt de forskellige belastninger, der kan kobles ind via kontakter. I serie med hver belastningsmodstand sidder en 1Ω modstand, som måleenheten mäter over.

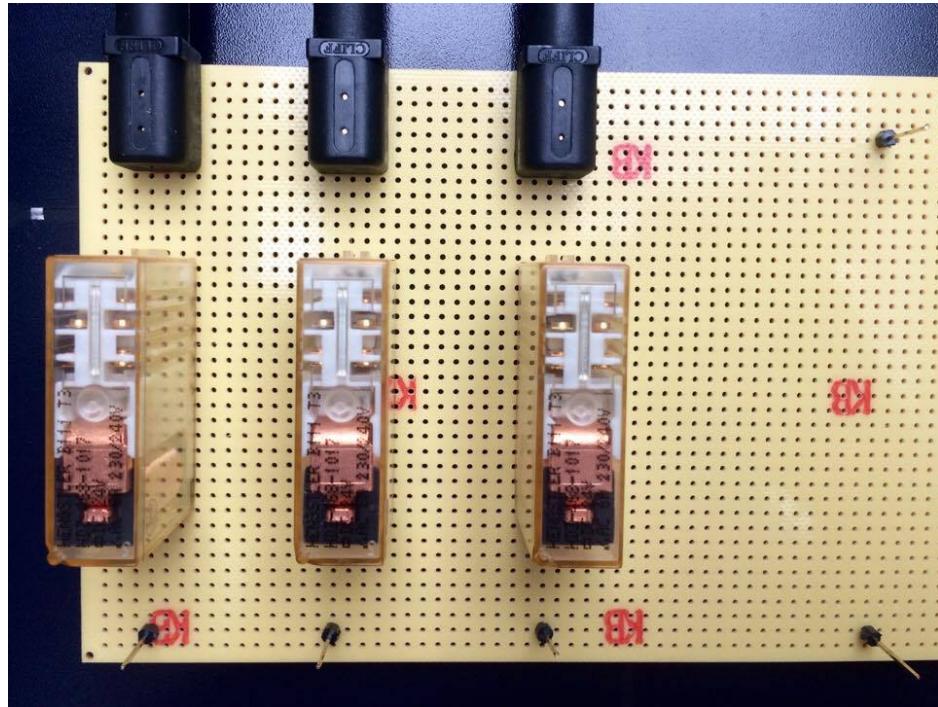


Figur 8.7: Kredsløbstegning af Distributionslinje, belastninger og Trinskifter

Som vist på tegningen er hver relæspole forbundet til hvert sin udgang på PLC, og det er herfra de modtager styresignalet. Hvert relæ er desuden forbundet til henholdsvis trin 4, 5 og 6. Gennem en Normally Open contact i hvert relæ er trinene videre forbundet ud til

distributionslinjen. Ved skift af trin fra eksempelvis 4 til 5 sendes højt signal til begge trin og herefter slukkes signalet til trin 4. Der er altså et overlap ved skift af trin, og på den måde forsvinder forsyningen til belastningen ikke undervejs.

Relækredsløbet er implementeret på et printkort, der forbindes til transformertrinene via bananstik, og til PLC og distributionslinje via pins. Det færdige kredsløb ses på figur 8.8



Figur 8.8: Relæer på printkort

8.5 Modultest

8.5.1 Spænding over belastning

For at tjekke, at distributionslinjen og de forskellige belastninger giver det spændingsfald, der forventes laves en modultest af det samlede system med distributionslinje, belastninger og trinskifter. Der vælges forskellige belastninger og trin på transformeren, hvor for hver indstilling måles spændingsfaldet over belastningen. Resultaterne heraf ses i tabel 8.1.

Trin	Belastning	Forventet resultat	Resultat
4 V AC	50Ω 50Ω	3,2 V AC over belastning	2,9 V AC målt over belastning
	50Ω 16Ω	2,6 V AC over belastning	2,0 V AC målt over belastning
	54Ω	3,58 V AC over belastning	3,38 V AC målt over belastning
	50Ω	3,6 V AC over belastning	3,5 V AC målt over belastning
	39Ω	3,5 V AC over belastning	3,4 V AC målt over belastning
	16Ω	2,8 V AC over belastning	2,5 V AC målt over belastning
5 V AC	50Ω 50Ω	4,0 V AC over belastning	3,7 V AC målt over belastning
	50Ω 16Ω	3,22 V AC over belastning	2,76 V AC målt over belastning
	50Ω	4,4 V AC over belastning	4,6 V AC målt over belastning
	39Ω	4,3 V AC over belastning	4,2 V AC målt over belastning
	16Ω	3,5 V AC over belastning	3,1 V AC målt over belastning

Tabel 8.1: Modultest for distributionslinje, belastninger og trinskifter

Det kan dermed ud fra modultesten konkluderes, at spændingen over belastningerne falder som forventet, dog med en afvigelse på maksimalt 0,5V. Afgivelsen er størst, jo lavere belastning som tilsluttes.

8.5.2 Power factor

For at teste at systemet virker som ønsket, laves en sammenligning af forventede power factor værdier og de værdier, der rent faktisk måles i systemet. Resultaterne herfra ses i tabel 8.2. Det ses at variationen i power factor er meget lille, og der ses derfor heller ingen forskel i de målte værdier.

Trin	Belastning	Simulering	Målt værdi
4 V AC	54Ω	0,997	0,999
	50Ω	0,997	0,999
	39Ω	0,996	0,999
	16Ω	0,982	0,999
5 V AC	54Ω	0,997	0,999
	50Ω	0,997	0,999
	39Ω	0,996	0,999
	16Ω	0,982	0,999

Tabel 8.2: Modultest for power factor

Kapitel 9

Design af Måleenhed

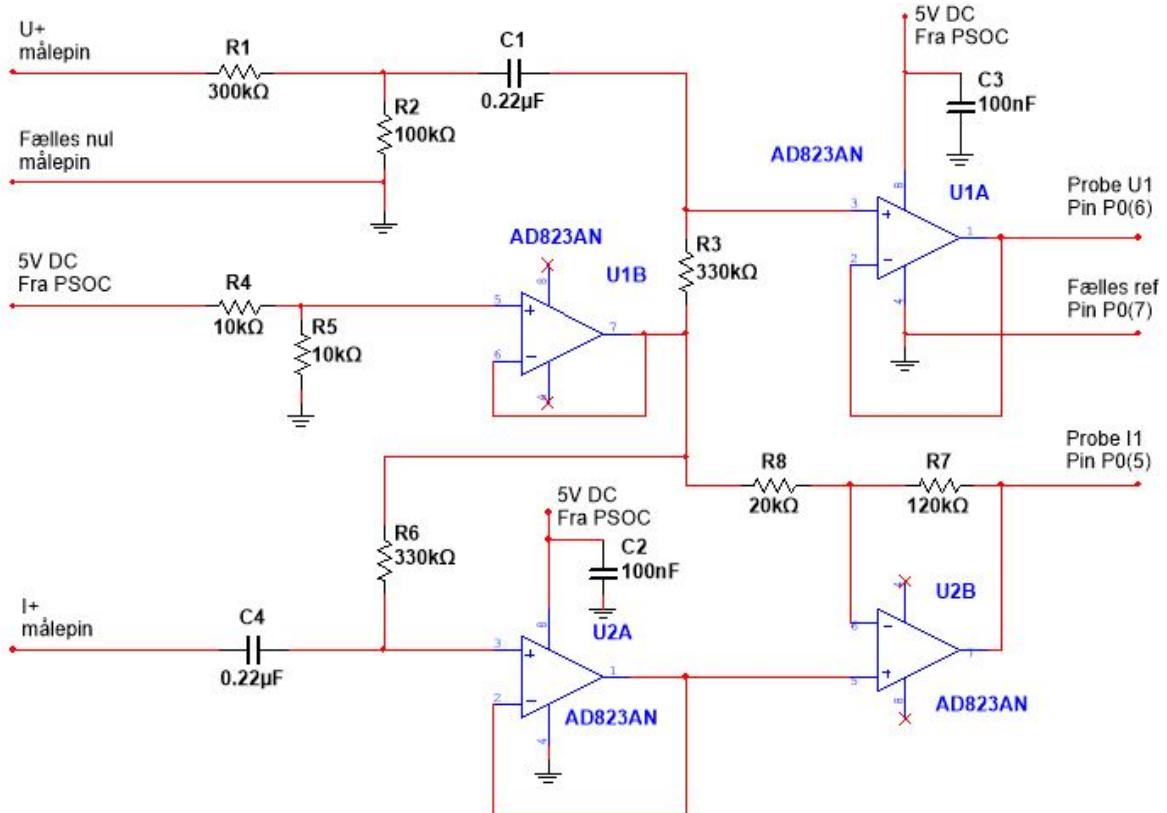
I dette afsnit beskrives, hvordan Måleenheden er opbygget. Måleenheden er designet til at kunne måle centralt og decentralt, på projektets simulering af transmissionslinje og forbrugere. Der er på netværket forberedt til at der kan tilsluttes måleenheder på forbrugerne. Spændingen kan måles direkte over forbrugeren. På hver forbruger er der tilsluttet 1 1Ω modstand i serie. På denne modstand måles spændingen, som vil være proportional med strømmen i systemet. Måleenheden er hovedsageligt software, der sampler og udregninger rms, power factor og THD. Der er dog også lavet hardware til at bearbejde signalerne inden PSOC'en.

9.1 Hardware

Måleenhedens hardware skal kunne dæmpe spændingen og forstærke strømmen så det ligger inden for PSOC'ens ADC spændingsområde. ADC kan sample signaler mellem 0 og 5Vpp, Det betyder at signalernes offset skal ændres til 2,5V.

9.1.1 Diagram

På figur 9.1 ses diagrammet for det hardware der bruges til at dæmpe spændingen, forstærke strømmen og hæve offsettet.



Figur 9.1: Hardware diagram

Spændingsmåler

For at kunne måle den maksimale spænding på 8V rms, dvs.

$$V_{pp} = 8V * \sqrt{2} = 11,3V_{pp} \quad (9.1)$$

Den mindste dæmpning skal derfor være:

$$\min_daemp = \frac{11.3}{5} = 2,26 \text{ gange} \quad (9.2)$$

Dette er valgt realiseret med en spændingsdeler ved R1 og R2. De dæmper med 3 gange og niveauet kommer derfor indenfor marginen. Derefter bliver offsettet hævet ved hjælp af signalet fra U1B. Til sidst føres signalet gennem spændingsfølgeren U1A for at sikre spændingen.

Strømmåling

Strømmålingen laves ved at måle spændingen over en 1Ω modstand. Modstanden sidder placeret ved belastningen. Iht. ohmslov vil det give strømmen i systemet .

$$I = \frac{U}{1\Omega} = U \quad (9.3)$$

Første Opamp U2A ved strømmålingen bruges til at hæve DC offset til 2,5V. Opamp U2B bruges til at forstærke strømmen så der kommer bedre præcision på målingen. Den maksimale forstærkning, for at den passer indenfor PSOC'ens ADC sample område bliver derfor:

Den maksimale Ipp der vil kunne måles:

$$I_{pp} = 0,5 * \sqrt{2} = 0,71A \quad (9.4)$$

Maksimale forstærkning:

$$\text{maks_forstaerkning} = \frac{5}{0,71} = 7 \quad (9.5)$$

Denne forstærkning ses realiseret vha. modstand R7 og R8.

PSOC tilslutning

Spændingen og strømmen bliver målt til den fælles nul i systemet, der bliver koblet på hardwarens stel forbindelse. Spænding, strøm og stel forbindelsen tilsluttes PSOC'en, som vist på figur 9.1.

9.2 Software

I dette afsnit vil softwaren for Måleenheten blive beskrevet. Herunder beskrivelser af det overordnede flow i koden, samt en uddybning af udviklede funktioner til sampling, Fourier-udregninger og Total Harmonic Distortion.

9.2.1 Overordnet beskrivelse

Softwaren til Måleenheten er allokeret på PSOC. Det overordnede flow i koden er beskrevet i Figur 9.2.

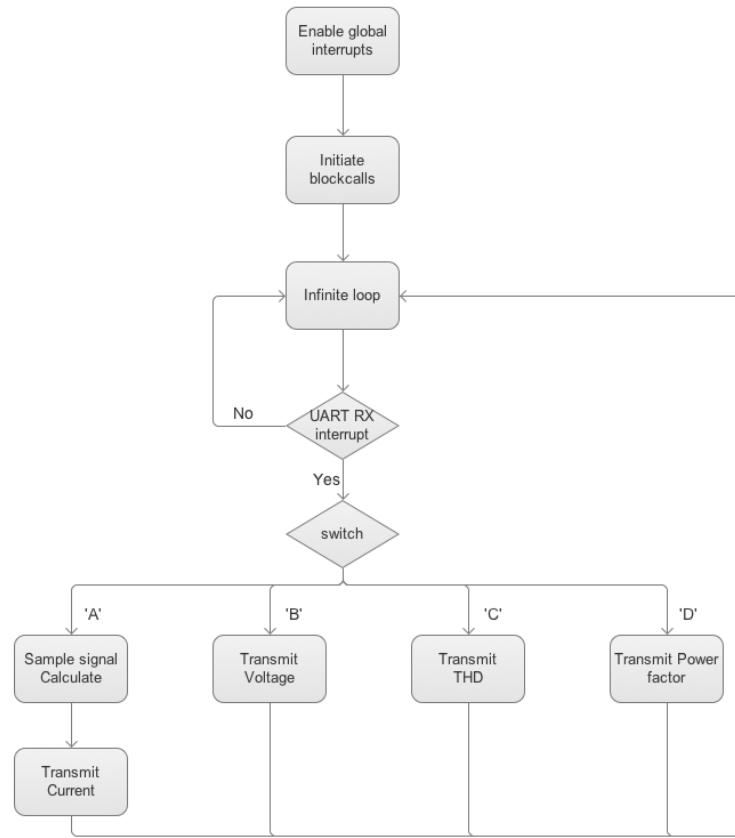
Som det første åbnes der for globale interrupts, så der kan modtages receive-interrupts fra UART forbindelsen til Styringsenheden. Herefter initieres de anvendte blokke i PSOC, herunder ADC og UART.

Efter initialiseringen går koden i et uendeligt loop hvor der ikke er nogen aktivitet. Her ventes der på et receive-interrupt fra UART forbindelsen. Ifølge protokollen for kommunikation mellem Måleenhed og Styringsenhed, se kapitel 7, forventes det at rækkefølgen af karakterer der modtages er "A-B-C-D". Denne rækkefølge er afgørende, da sampling og udregning, først starter ved modtagelse af karakteren 'A'.

Når udregningerne er udført, sendes den udregnede RMS-værdi for den målte strøm, over UART-forbindelsen. Værdien, som er en 16bit integer, sendes i to bytes, henholdsvis MSB og LSB.

Når de to bytes er sendt til Styringsenheden forlades interruptrutinen, og koden returnerer til det uendelig loop, hvor der ventes på at modtage den næste karakter i rækkefølgen.

Ved modtagelse af de efterfølgende karakterer sendes værdierne for henholdsvis spænding, THD og Power Factor.



Figur 9.2: Overordnet flowchart for software på Måleenhed

9.2.2 Analog til digital konvertering

Måleenheden omsætter to analoge spændingssignaler til digitale værdier for strøm og spænding. Denne konvertering foretages af en Delta Sigma Analog to Digital Converter (ADC) i PSOCen, se datablad for ADC i bilag A1. ADC er en komponent i PSOC, som med en read-funktion kan returnere en 16-bit værdi af det analoge signal på indgangen.

På Figur 9.3 ses blokkaldet i PSOC creator 4.0. Blokken er sat op med følgende parametre:

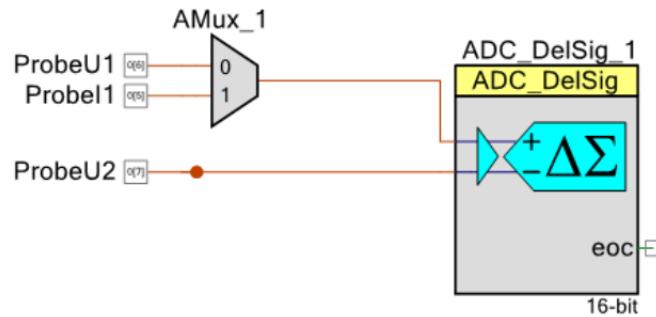
$$\text{Resolution} = 16\text{bit} \quad (9.6)$$

$$f_{sample} = 41666\text{Hz} \quad (9.7)$$

Som det fremgår af Figur 9.3 er der anvendt en multiplexer, til at skifte mellem hvilkt signal der skal samples. Dette skyldes at der kun er én ADC i PSOCen. Samplingen af signalet styres i C-koden, hvor der anvendes to af ADC-blokkens API funktioner.

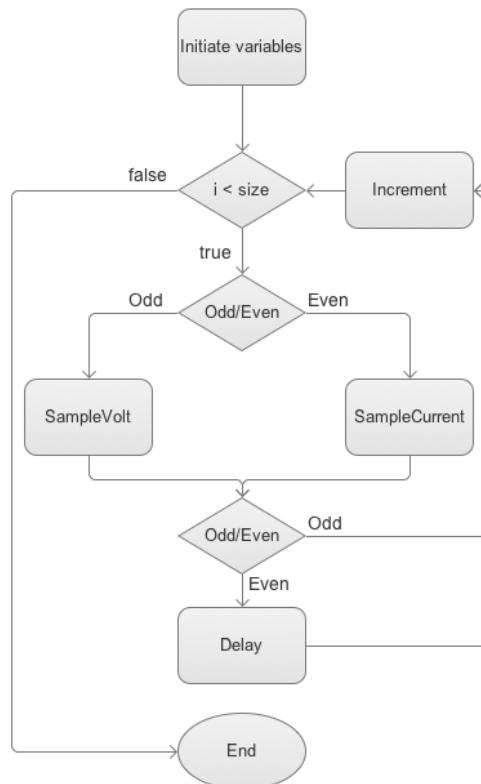
GetResult16() starter en ADC konvertering, venter på at denne er færdig, stopper konverteringen og returnerer en 16bit signed integer værdi af resultatet.

CountsTo_mVolt() konverterer resultatet af GetResult16() til en værdi i millivolt.



Figur 9.3: ADC blok fra PSOC creator 4.0

Koden til sampling af signalerne er vist grafisk i et flowchart på Figur 9.4. Samplingen kører i en for-løkke, hvor der skiftevis tages et sample af spændingen og strømmen. Efter hvert andet sample, indføres der et delay, således at sampletiden passer med én periode af et 50Hz sinussignal. Antallet af samples er 64 pr. signal, altså 128 i alt. De samplede værdier gemmes i to arrays, som bruges i de efterfølgende beregninger.



Figur 9.4: Flowchart for sample funktionen i PSOC

Realisering af samplefrekvens

Det indførte delay i for-løkken skal sikre at sampletiden svarer præcis til én periode af et 50Hz sinussignal.

$$T_{sample} = \frac{1}{50Hz} = 20 \text{ ms} \quad (9.8)$$

Det betyder at tiden mellem hvert sample skal være

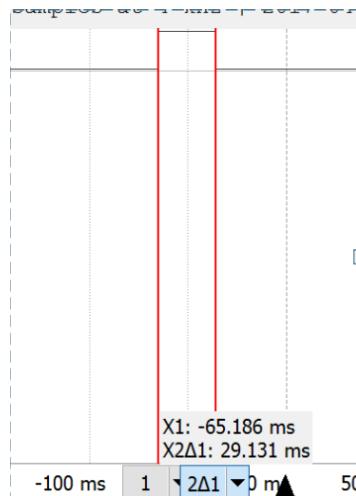
$$t = \frac{T_{sample}}{128} = 156.25 \mu\text{s} \quad (9.9)$$

Den reelle tid der er mellem samples afhænger af hvor lang tid det tager at eksekvere koden på PSOC. Denne tid er fundet ved at toggle et testben på PSOC, før og efter samplingen, og anvende Analog Discovery's Logic Analyser til at måle tiden. Tiden det tager at tage et sample fra hvert signal er aflæst til: $t_{reel} = 131.81 \mu\text{s}$. Herefter kan størrelsen på delayet udregnes.

$$resttid = T - (t_{reel} * 64) = 11.564 \text{ ms} \quad (9.10)$$

$$delay = \frac{resttid}{64} = 180.69 \mu\text{s} \quad (9.11)$$

Dette delay indsættes i koden og Logic Analyser bruges til at finde den reelle sampletid, som findes til $T_{sampleReel} = 29.131 \text{ ms}$, se Figur 9.5.

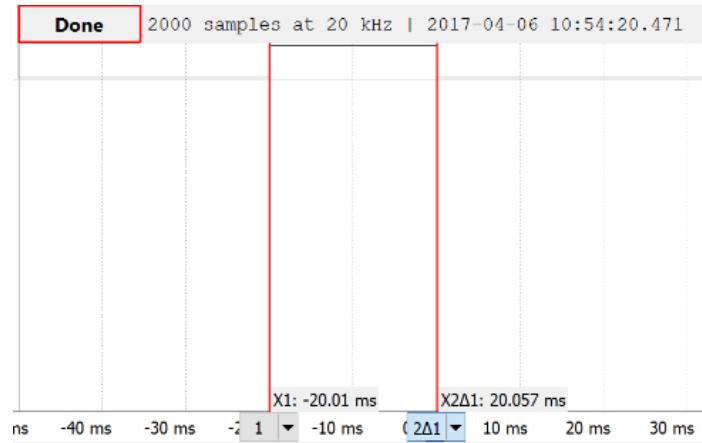


Figur 9.5: Reel sampletid med beregnet delay, fundet med Logic Analyser

Det viser sig at delayet er blevet for stort. Derfor laves endnu en beregning, hvor denne fejl fratrækkes delayet fra (9.11).

$$delay_2 = delay - \frac{29.131 \text{ ms} - 20 \text{ ms}}{64} = 38.018 \mu\text{s} \quad (9.12)$$

Ved at indsætte dette delay, kommer sampletiden meget tæt på de 20 ms, og med småjusteringer af delayet, findes den rigtige værdi til 40 μs , se Figur 9.6, hvor sampletiden er aflæst til 20.057 ms.



Figur 9.6: Reel sampletid med endeligt delay, fundet med Logic Analyser

9.2.3 Fourier algoritme

Når Strøm og spænding er samplet, laves der Fourier på dem hver især vha. funktionen 9.1

Listing 9.1: FFT prototype

```
|| void FFT(long m, double *x, double *y, double *u);
```

Funktionen kan ses i bilag A1 i sample.cpp filen.

Denne funktion dækker over en algoritme der kan lave FFT¹ på et samplet signal. Algoritmen er fundet på et Cypress forum[4]. Funktionen kan lave FFT på et array med 64 tal, hvor den laver arrayet om til de reale tal og fylder et andet array med de imaginær tal. Derudover fylder den et tredje array med absolute amplitude værdiger. Det første argument m er eksponent til 2, så det giver størrelsen af arrayet. Ved 64 bliver eksponent 6.

Algoritmen starter med at lave noget der kaldes "bit-reversal", Dette betyder at man bytter alle pladserne i arrayet på følgende måde:

Array plads	array plads binær	→	array plads binær	Array plads
0	000000	→	000000	0
1	000001	→	100000	32
2	000010	→	010000	16
3	000011	→	110000	48
4	000100	→	001000	8
5	000101	→	101000	40

Tabel 9.1: Bit-reversal example

Dette gøres for at FFT algoritmen, kan laves med et minimalt antal loops, og for at det samme array kan fyldes med de reale tal. Den næste del af funktionen, bryder DFT beregningen ned i mindre dele, som laves i nogle loops der fylder 2 arrays med hhv. de reale og imaginære værdiger. Det er en algoritme der har taget udgangspunkt i Cooley - Tukey's FFT algoritme [3], som er udviklet i 1965, men stadigvæk er en af de hurtigste FFT algoritmer. Den sidste del af funktionen beregner størrelsen af signalet ved hvert frekvensbin, med følgende formel.

¹Fast Fourier Transformation

$$X_{amplitude} = \frac{2}{N} * \sqrt{X_{Real}^2 + X_{Imag}^2} \quad (9.13)$$

For at kunne bruge fourier til noget skal der udregnes hvor langt der er mellem hver frekvensbin. For at gøre det skal samplingstiden kendes. Når der samples 64 gange over 1 periode bliver samplefrekvensen:

$$f_{sample} = 64 * 50Hz = 3200Hz \quad (9.14)$$

Hvilket betyder at der mellem hver frekvens bin bliver:

$$f_{Bin} = \frac{3200Hz}{64} = 50Hz \quad (9.15)$$

Dette betyder at der på plads nummer 2 kan findes værdiger for signaler med 50Hz, og der på plads nummer 3 kan findes værdier for signaler med 100Hz, osv.

Test af Fourier funktionen

FFT funktionen er testet ved at bruge MATLAB's FFT funktion på et array magen til. Inputtet er et samplet sinus signal med et offset på 2,5 og en amplitude på 0,5.

Frekvensbin	PSOC Real	PSOC Imag	Matlab Real	Matlab Imag
0	161607	0	161607	0
1	-2311,77	1891,26	-2311,77	1891,26
2	-4846,65	8975,04	-4846,65	8975,04
3	2909,31	-1067,90	2909,31	-1067,90
4	500,48	-3947,03	500,48	-3947,03
5	30,45	-2556,68	30,45	-2556,68

Tabel 9.2: FFT test

Resten af tabellen kan ses i bilag B1 "Måleenhed FFT sammenligning"

9.2.4 Beregning af rms og power faktor

Beregning af rms ved 50Hz

Rms kan per definition beregnes for strøm og spænding ved følgende:

$$X_{rms} = \frac{X_{amplitude}}{\sqrt{2}} \quad (9.16)$$

Derfor kan rms værdigen ved 50Hz beregnes med følgende funktion:

Listing 9.2: Funktion til beregning af 50Hz rms

```
double calculate_50Hz_RMS(double *u)
{
    return u[1]/sqrt(2);
}
```

Hvor "u" er et array med absolute værdier for spænding eller strøm med frekvensbin på 50Hz.

Beregning af Power faktor ved 50Hz

Power faktor kan per definition udregnes som cosinus til vinklen mellem strøm og spænding. Ved at lave Fourier på signalet, kan det komplekse tal anvendes til at lave en vektor. Power faktor for signalet på 50Hz bliver derfor vinklen mellem spænding og strøm på andet plads i arrayet.

Vinklen på et kompleks tal er per definition:

$$\delta = \text{atan}\left(\frac{\text{Imaginaire}}{\text{Real}}\right) \quad (9.17)$$

Power faktor kan derfor regnes som:

$$PF = \cos(\delta - \beta) \quad (9.18)$$

Måleenheden kan ikke se forskel på om det er et lag eller lead system. Det forventes næsten altid at systemet har mere induktiv end kapacitiv belastning. Derudover betyder det heller ikke noget for effektiviteten om det er lag eller lead. Derfor er det blevet nedprioriteret på måleneheden.

Listing 9.3: Funktion til beregning af 50Hz PF

```
double calculate_50Hz_PF()
{
    double angle_Volt = atan(Im_volt[1]/Re_volt[1]);
    double angle_Ampere = atan(Im_Ampere[1]/Re_Ampere[1]);
    return cos(angle_Volt - angle_Ampere);
}
```

9.2.5 Total Harmonic Distortion

Måleenheden indeholder en funktion til beregning af Total Harmonic Distortion (THD). THD er et udtryk for hvor stort indholdet af overharmoniske frekvenser er i et givent signal. Signalet som Måleneheden mäter på, består ideelt set kun af grundfrekvensen på 50Hz.

Udregningen af THD i Måleenheden udføres efter der er lavet Fourier analyse af signalet. Herefter udregnes THD ved at dele den kvadrede sum af alle overharmoniske amplituder med amplituden af den fundationale frekvens. Den generelle formel for udregning af THD er vist i Ligning 9.19.

$$THD = \frac{\sqrt{\sum_{n=2}^{\infty} V_n^2}}{V_1} \quad (9.19)$$

Måleenheder udregner THD på baggrund af de første 5 frekvensbins, inklusiv den fundationale frekvens jf. ligning 9.20.

$$THD = \frac{\sqrt{V_2^2 + V_3^2 + V_4^2 + V_5^2}}{V_1} \quad (9.20)$$

Test af THD beregning i Måleenhed

Udregningen af THD i Måleenheden er testet ved at sammenligne med resultatet med en simulering i Matlab. I Matlab er der lavet et signal der svarer til et firkantsignal i frekvensdomænet, hvorefter der er udregnet THD for de første 5 frekvensbins. Matlab koden ses i Listing 9.4. Resultatet af Matlabsimuleringen bliver

$$THD_{square_theoretical} = 0,3887 \quad (9.21)$$

Listing 9.4: Beregning af THD for firkantsignal i Matlab

```

freq = 50;

%%% double check THD of square wave using the fourier series:
n = 1:10000;
freq_vec = freq*n; % fourier series frequency vector
amp_vec = (4/pi) * (1./n).*(mod(n,2)==1);

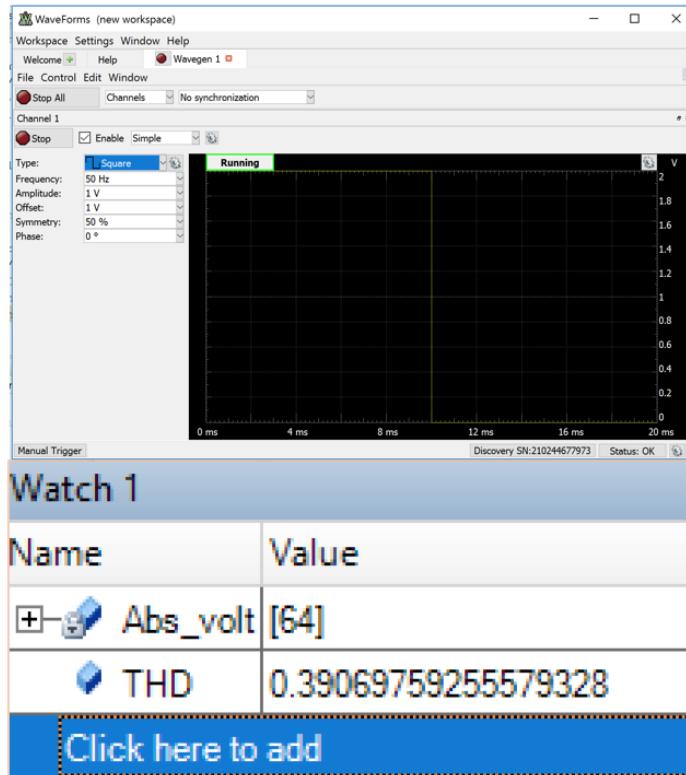
% compute THD by definition:
THD_square_theoretical = (sum(amp_vec(2:5).^2) / amp_vec(1)^2)^0.5

```

Herefter laves en test på Måleenheden. Forudsætning for testen er at sampling og Fourier-transformationen fungerer. Til testen bruges en funktionsgenerator til at lave det firkantsignal der skal måles på. Signalet forbinderes til det analoge spændings input på PSOCen, hvorefter debuggeren i PSOC creator 4.0 startes. Værdien for THD i måleenheden bliver

$$THD_{måleenhed} = 0,391 \quad (9.22)$$

hvilket er inden for de 10% præcision specificeret i Ikke Funktionelle Krav (sektion 3.4.3). På Figur 9.7 ses resultatet fra debuggeren og indstillerne på funktionsgeneratoren.



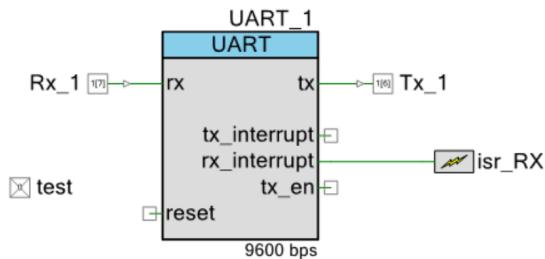
Figur 9.7: Test af THD funktion på Måleenhed

9.2.6 UART forbindelse til styringsenhed

Kommunikationen mellem Måleenheden og Styringsenheden er som det fremgår af Figur 5.4 allokeret på en UART forbindelse. UART kommunikationen foregår via en RX- og en

TX-forbindelse mellem enhederne. Opsætning af UART forbindelsen i PSOC creator 4.0 sker ved et blokkald.

Af Figur 9.8, fremgår det at RX og TX forbindelserne på UART blokken er forbundet til to hardware pins på PSOC, henholdsvis Rx_1 og Tx_1. Derudover der der forbundet en interrupt-rutine til rx_interrupt benet. Det er denne forbindelse der sikrer at Måleenheten ender i interrupt-rutinen, som er vist i Figur 9.2



Figur 9.8: UART blok fra PSOC creator 4.0

Opsætning af UART fobindelsen er foretaget i henhold til protokollen beskrevet i Kapitel 7.2

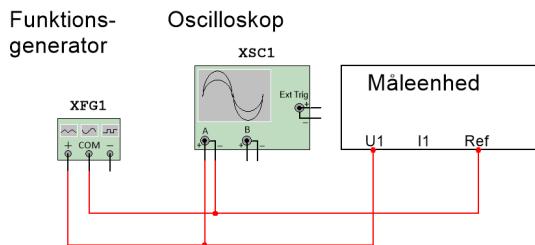
9.2.7 Kalibrering af måleenhed.

Efter test af funktionaliteten af Måleenheden laves en kalibrering, for at fjerne det offset, som er opstået i forbindelse med forstærkning/dæmpning af signal i hardwaren.

Måleopstilling

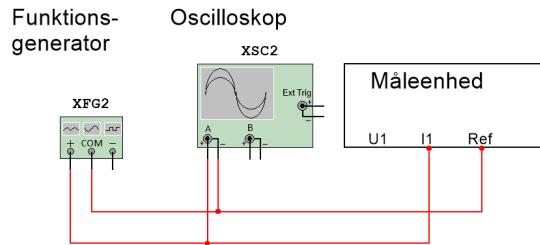
Til kalibrering af måleenheden er anvendes en funktionsgenerator, til at lave sinussignalet, som forbindes direkte til måleenheden. Til korrekt måling af spændingsværdier anvendes oscilloskop, som her antages for værende ideel. Måleenheden er tilsluttet en PC og kører i debug-mode, så værdierne for strøm og spænding kan aflæses.

Kalibrering af spændingsmåling



Figur 9.9: Måleopstilling for kalibrering af spændingsmåling

Kalibrering af strømmåling



Figur 9.10: Måleopstilling for kalibrering af strømmåling

Fremgangsmåde

Der laves to måleserier, for henholdsvis spænding- og strømmåling. Strømmålingen er baseret på en spændingsmåling over en 1Ω modstand, hvilket betyder at spændingsfaldet over modstanden lig med strømmen. Derfor kalibreres strømmålingen med et spændingssignal, på

samme måde som spændingsmålingen. Måleenheden er lavet til at måle en spænding mellem 0-8Vrms og en strøm mellem 500 mA, så der laves målinger i dette område med intervaller på 500 mV og 50 mV.

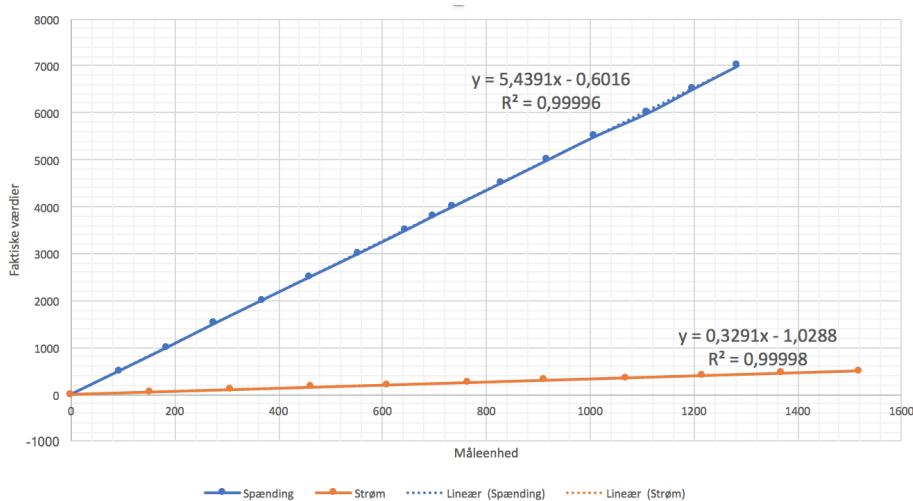
Der forventes at se en lineær sammenhæng mellem faktiske spændinger og værdierne fra debuggeren på Måleenheden. Ved at finde hældningen på denne sammenhæng findes den konstant, som skal ganges på resultatet i måleenheden, for værdierne passer med den faktiske spænding.

Resultater

Måleresultaterne fra kalibreringen findes i Tabel 9.3. Sammenhængen mellem faktiske spændinger og værdierne i Måleenheden er vist i Figur 9.11.

Tabel 9.3: Måleresultater for kalibrering af Måleenhed.

Måling nr.	Faktiske værdier		Måleenhed	
	Spænding	Strøm	Spænding	Strøm
1	0	0	0	0
2	500	50	93	152
3	1000	100	185	308
4	1509	150	276	462
5	2008	199	368	610
6	2503	250	460	764
7	2999	299	553	912
8	3502	350	645	1069
9	3808	399	699	1216
10	4006	450	736	1370
11	4505	500	828	1518
12	5000		918	
13	5499		1009	
14	6000		1111	
15	6505		1197	
16	7006		1284	



Figur 9.11: Sammenhæng mellem faktiske spændinger og værdier i Måleenheden.

Faktoren som skal ganges på resultatet i Måleenheden findes for henholdsvis spænding og strøm, ved hældningen af de to lineære funktioner på Figur 9.11.

$$a_U = 5,4391 \quad (9.23)$$

$$a_I = 0,3291 \quad (9.24)$$

9.3 Modultest

Efter udvikling af Måleenheden er der lavet test på modulet, for at sikre den lever op til kravene.

Test af spændingsmåling

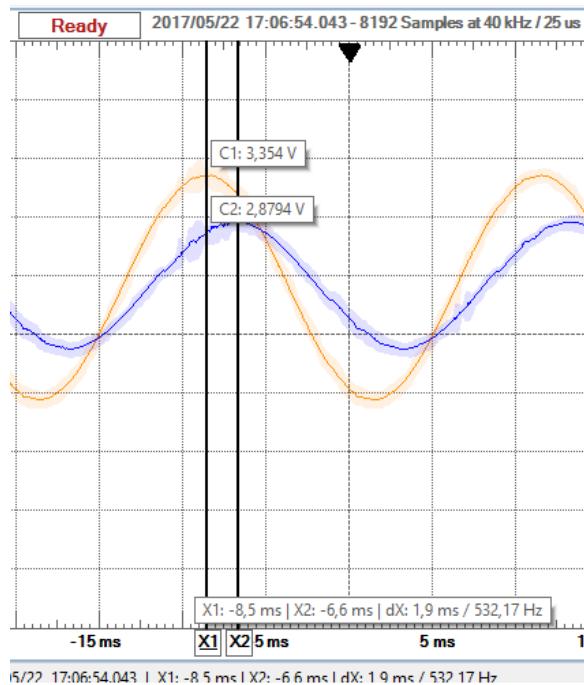
Test	Spændingsmåling
Testbeskrivelse	Der måles ved tre forskellige spændingsniveauer for at sikre at systemet kan måler i hele måleområdet
Input	Test 1: 1,02V. Test 2: 4,02V. Test 3: 8,08V
Forventet out-put	Test 1: 1,02V. Test 2: 4,02V. Test 3: 8,08V
Resultat	Test 1: 0,97V. Test 2: 4,00V. Test 3: 8,05V

Test af strømmåling

Test	Strømmåling
Testbeskrivelse	Der måles ved 3 forskellige strømniveauer for at sikre at systemet kan måler i hele måleområdet
Input	Test 1: 0,10V. Test 2: 0,30V. Test 3: 0,50V.
Forventet out-put	Test 1: 0,10V. Test 2: 0,30V. Test 3: 0,50V.
Resultat	Test 1: 0,10V. Test 2: 0,30. Test 3: 0,50V.

Test af power factor

Test	Power factor måling
Testbeskrivelse	Der måles om Power factor stemmer overens med den enlige Power factor. Dette gøres ved at måle den vinkel mellem strøm og spænding på oscilloskop, og derefter regnes power factor.
Input	2 sinus signaler med faseforskydning på 1,9ms svarende til en power factor på 0,827. Se Figur 9.12.
Forventet out-put	Power factor på 0,827
Resultat	Power factor på 0,842



Figur 9.12: Sinussignaler med faseforskydning anvendt til test af power factor.

Test af THD måling

Test af THD kan ses i kapitel 9.2.5.

Kapitel 10

Design af Styringsenhed

Styringsenheden er den del der skal sørge for at modtage data fra Måleenhederne og ud fra disse data styre trinskifteren. Til at koordinere kommunikation fra flere Målenheder til et Kontrolmodul er brugt en Arduino. Samtidig skal den informere en bruger om systemets tilstand gennem en brugergrænseflade. Her er anvendt en HMI. Kontrolmodulet er lavet på en PLC, der står for at opdatere trinskifteren og HMI ud fra de modtagende data.

10.1 Kontrolmodul

Kontrolmodulet består primært af to processer; En kommunikationsdel der kan håndtere at modtage data gennem en TCP forbindelse og konvertere disse data til noget der kan bruges i resten af programmet, herunder også HMI delen. En styringsdel der kan bruge disse data i styringen af trinskifteren.

Kontrolmodulet er programmeret på en Siemens S7-1200 PLC med en 1214C DC/DC/DC CPU og signalmodulet AQ1x12BIT. Sammen med PLC'en sidder en switch af typen CSM 1277 .

10.1.1 TCP kommunikation

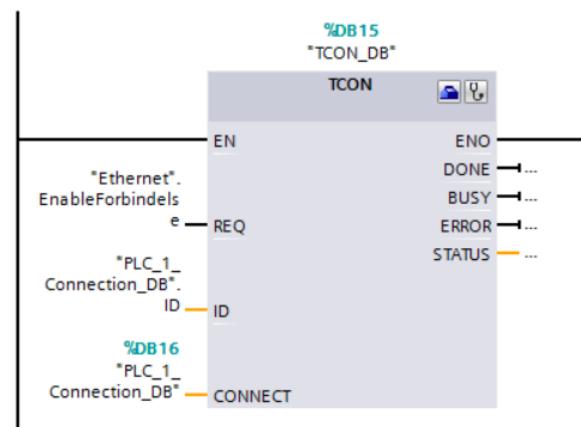
Kontrolmodulet er lavet som en client i forhold til kommunikationsmodulet, der er server. Den sender altså en forespørgelse på at modtage data til kommunikationsmodulet, hvorefter den modtager ny data fra den forespurgte enhed. Mere om selve protokollen kan findes i afsnit 7.1 TCP protokol. Det er kontrolmodulet der oprette TCP forbindelse og står for at nedlægge den igen.

TCP kommunikationen er opdelt i 4 FC'er; OpretForbindelse, SendData, ModtagData og AfslutForbindelse. Simatic TIA portal har nogle indbyggede open user communication blokke, der kan bruges til LAN kommunikation.

OpretForbindelse består af funktionsblokken TCON, som er vist på figur 10.1. Når der på REQ ses en rising edge, vil den forsøge at oprette forbindelse til en bestemt IP adresse og oprettholde den, også selvom REQ bliver sat 0 igen. Forbindelsen bliver vedligeholdt automatisk asynkront, så programmet kan udføre andet samtidigt. Hvis forbindelsen bliver brudt, skal REQ have en rising edge før TCON vil forsøge at oprette forbindelse igen. Datablokken Ethernet er oprettet med henblik på at indeholde globale statics anvendt i forbindelse med TCP kommunikationen, da FC'er ikke har nogen hukommelse fra scan til scan.

ID er identifikationen på forbindelsen internt i PLC'en, som de resterende blokke bruger til at identificere hvilken forbindelse de er tilknyttet, hvis der skulle være flere. Her er ID sat til 1, da der kun er en forbindelse til kommunikationsmodulet.

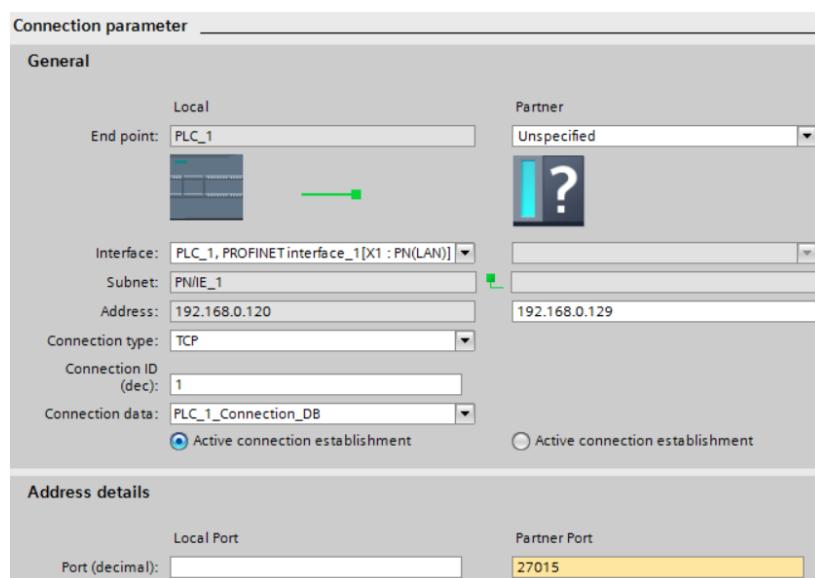
CONNECT parameteren er en pointer til data området der indeholder forbindelsesinformation. TCON muliggør at få informationer om forbindelsen tilstand gennem DONE, BUSY, ERROR og STATUS. Disse outputs ligger i den tilhørende DB TCON_DB og kan kaldes gennem den. I dette projekt er der dog ikke udviklet nogen form for fejlhåndtering og disse outputs anvendes ikke. Der henvises til bilag B-2,kapitel 11, for mere dybdegående forklaring af disse.



Figur 10.1: TCON brugt i funktionen OpretForbindelse

Selve konfigureringen af hvilke enheder(IP adresser) kommunikationen skal foregå imellem sættes under properties for TCON. På figur 10.2 kan det kan ses at på lokalsiden er PLC'en sat op til at have IP adressen 192.168.0.120, anvende TCP over subnet PN/IE_1, som er default 255.255.255.0, samt være aktiv i forhold til at oprette forbindelsen. Feltet Connection data, fortæller hvilken Datablok der indeholder data omkring forbindelse, herunder bla. ID og CONNECT.

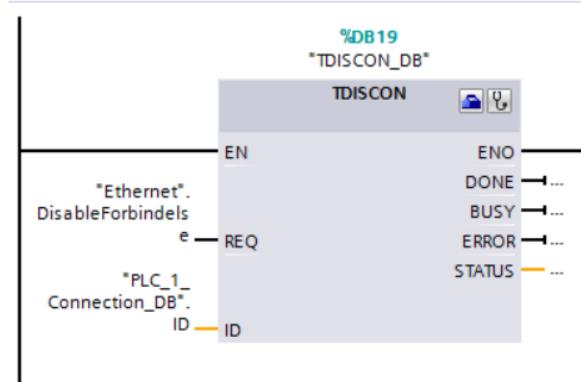
På partnersiden er Arduinoen ikke et Siemens modul og derfor unspecified. Her er valgt at kommunikationsmodulet har IP adressen 192.168.0.129 og bruger port 27015 til kommunikationen. Denne port er valgt af hensyn til Arduinoens præferencer i forhold til mulige porte.



Figur 10.2: Konfiguration af TCP forbindelse

Funktionen AfslutForbindelse står for at nedlægge forbindelse. Da der kun kommunikeres med en enhed, anvender programmet aldrig denne mulighed. Alligevel er funktionen lavet for at det er nemt at videreudvikle, med henblik på at tilføje flere TCP forbindelser.

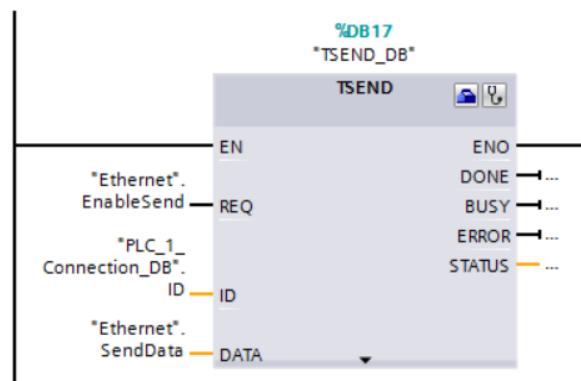
Funktionsblokken TDISCON anvendes til denne opgave. Når blokkens REQ ser en rising edge nedlægges forbindelsen og det vil kræve en rising edge på TCON's REQ for at genetablere forbindelsen. Blokken skal også bruge et ID på forbindelsen, ID'et er sat i konfigurationen af TCON. Ingen kan informationer omkring blokken fås gennem DONE, BUSY, ERROR og STATUS. Disse anvendes ikke, da der ikke er lagt vægt på fejlhåndtering i dette projekt. De kan kaldes gennem datablokken TDISCON_DB. Der henvises til bilag B-2, kapitel 11 for mere dybdegående forklaring af disse.



Figur 10.3: TDISCON brugt i funktionen AfslutForbindelse

Når forbindelsen er oprettet bruges funktionerne SendData og ModtagData til kommunikering af data. SendData anvender funktionsblokken TSEND til at afsende data placeret i den globale static SendData. Når REQ modtager en rising edge afsendes dataet. Mens ID specificerer hvilken forbindelse, der skal sendes over.

Ligesom TCON og TDISCON anvendes DONE, BUSY, ERROR og STATUS ikke. Der henvises til bilag B-2, kapitel 11 for mere dybdegående forklaring af disse.



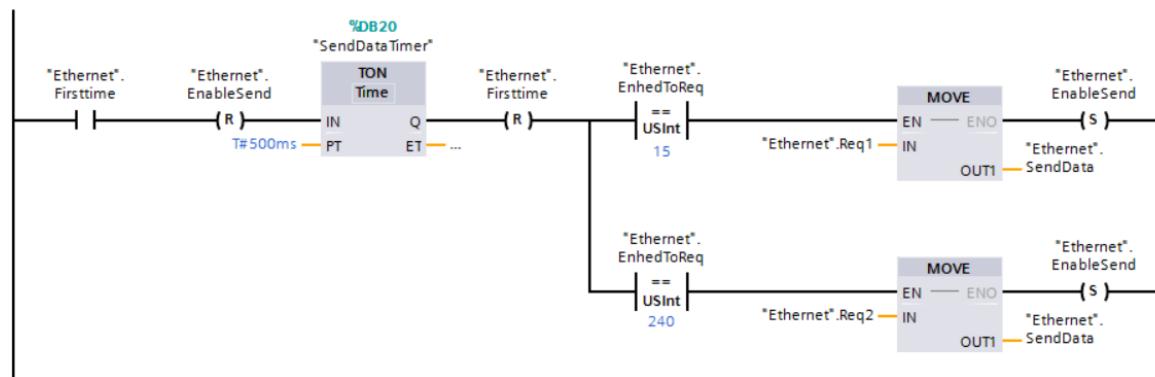
Figur 10.4: TSEND brugt i funktionen SendData

Til at styre hvor ofte der forespørges ny data og hvilken enhed der forespørges data fra, er designet et netværk, hvor en global static boolean Firsttime sættes true før send processen går igang. Firsttime bliver sat true når data er modtaget og gemt. Den er defualt true ellers vil det blokere programmet i kraft af at PLC'en ikke modtager data før den har sendt en kommando på ny data.

Når Firsttime er true igangsættes netværket vist på figur 10.5. Her resettes den static der styrer REQ på TSEND og en on delay timer, SendDataTimer, startes. Timere anvendes i programmet for kunne styre hvor ofte der hentes ny data. For systemet der udvikles er det ikke nødvendigt at dette går hurtigt. Så det blev besluttet at data fra både den centrale og den decentrale måleenhed skulle opdateres hver 2 sekunder. Derfor er der valgt en timer på 500ms i både SendData og ModtagData funktionen. Hvilket resultere i 1 sekund pr enhed og 2 sekunder samlet.

Når SendDataTimers udgang går true, resettes Firsttimer til false og den gloable static EnhedToReq sammenlignes med to konstanter 15 og 240, som svarer til Måleenhed Central og Måleenhed Decentral. Disse værdier er valgt da de er 1's komplement af hinanden. Så når de forespurgte data er modtagede og gemt kan man skifte EnhedToReq ved at inverte den unsigned integer.

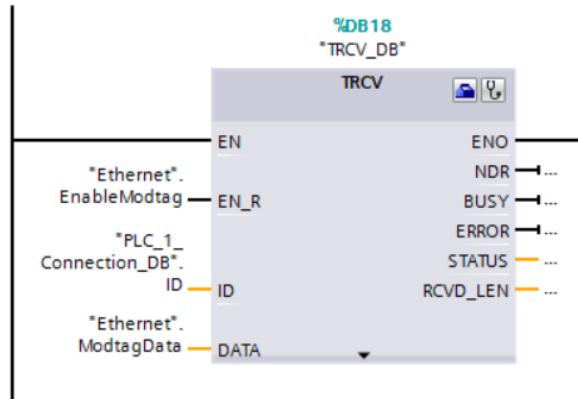
Til sidst på figur 10.5 flyttes kommandoen for den valgte enhed over i SendData, som er det char array der sidder på TSEND's DATA ben og REQ sættes true via EnableSend boolean.



Figur 10.5: Netværket der styrer hvilken enhed der forespørges data fra

ModtagData funktionen er opbygget på meget samme måde som SendData funktionen med en funktionsblok TRCV vist på figur 10.6 til selve TCP kommunikation og et netværk designet til at gemme det modtagede data. Netværket kan ses på figur 10.7. TRCV er opbygget med en EN_R, der så længe den er true vil modtage data. Den globale static EnableModtag er default true, da funktionsblokken bliver udført asynkront, så den vil ikke blokkere programmet herved. Ligeså snart en forespørgelse på ny data er sendt er programmet altså klar til at modtage et svar.

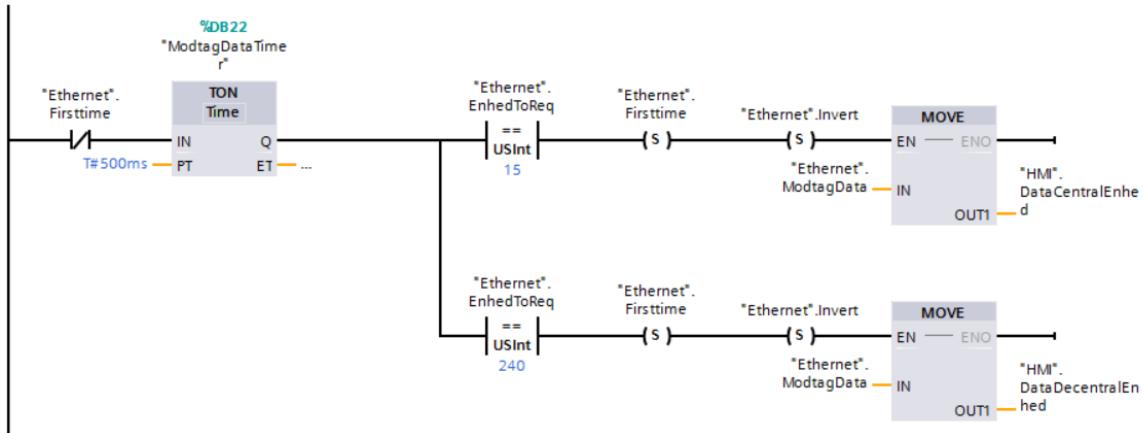
ID er det samme som for de resterende kommunikationsblokke. DATA dækker over hvor modtaget data skal gemmes. Som i de resterende blokke anvendes outputs ikke. Der henvises til bilag B-2, kapitel 11 for mere dybdegående forklaring af disse.



Figur 10.6: TRCV brugt i funktionen ModtagData

Ligeså snart SendDataTimers output går true, sættes Firsttime false, hvilket sætter gem modtagede data netværket igang. Her er placeret ModtagDataTimer, som også er en on delay timer. Når dens output går true, vælges i hvilket global static word array, DataCentralEnhed og DataDecentralEnhed, dataet skal gemmes. Disse anvendes i forbindelse med HMI'et. Se afsnit 10.2. Hvilket der bliver valgt afhænger igen af EnhedToReq. Samtidig sættes Firsttimer true, så data igen kan forespørges i SendData funktionen. Den global static boolean Invert, anvendes til den tidligere beskrevne invertering af EnhedToReq. Dette sker i et separat netværk til sidst i ModtagData funktionen, hvor en true Invert muliggør invertering ved hjælp af en funktion INV og sætter Invert false.

SendData og ModtagData funktionerne looper altså i en evig løkke, hvor Måleenheden der forespørger data fra efter hver runde skiftes.

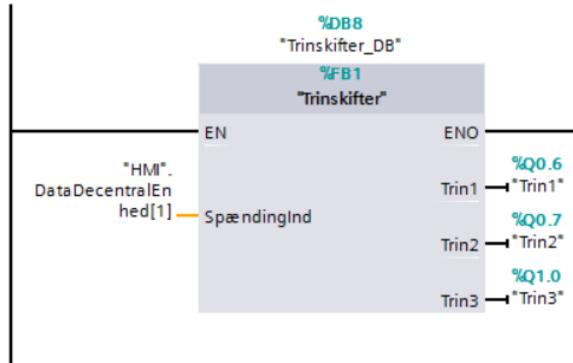


Figur 10.7: Netværket der gemmer modtagede data

I dette program er der kun en OB, OB1, der eksekveres, de fire funktioner OpretForbindelse, SendData, ModtagData og AfslutForbindelse ligger herfor i OB1 i det ene netværk af to. Det andet netværk indeholder styringen af trinskifteren og indeholder en FB; trinskifter.

10.1.2 Styring af trinskifteren

FB'en trinskifter har et input og tre outputs. Input i form af DataDecentralEnhed[1]. På plads et i word arrayet ligger information omkring spændingen målt ude ved den centrale Målenhed også benævnt Forbruger 1. Output er tre booleans Trin 1, Trin 2 og Trin 3, der hver er forbundet med et output på PLC'en, henholdsvis Q0.6, Q0.7 og Q1.0. Se figur 10.8.

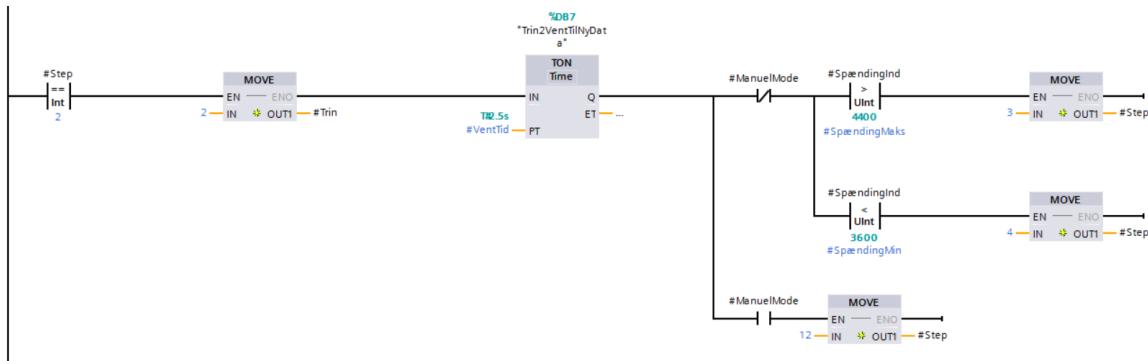


Figur 10.8: FB'en Trinskifter

Trinskifter består af 13 netværk, 1 initieringsnetværk, 5 automatisk mode netværk og 7 manuel mode netværk. Disse eksekveres i sekvenser, styret af den lokale static integer Step. Netværk 1 er et initieringsnetværk, der sætter Trin2 true og sætter Step til 2 med funktionen MOVE. Step 2 svarer til at systemet er i Trin 2 i automatisk tilstand. Trin 2 i koden er Trin 5 på trintransformeren.

På figur 10.9 ses det netværk i den automatiske del, hvor udgangen Trin 2 er sat. Trin er her en static, der bruges i HMI'et. Se afsnit 10.2. For at undgå at programmet kan nå at lave et multistep, altså f.eks. fra Trin 1 til Trin 3, er sat en on delay timer, Trin2VentTilNyData. Konstanten VentTid er sat til 2,5 sekunder, hvilket er mere end de 2 sekunder der går før der er ny data. Dermed får man ny data fra et nyt stabilt niveau efter et trinskift og programmet vil altså ikke skifte endnu et trin grundet gamle målingerne.

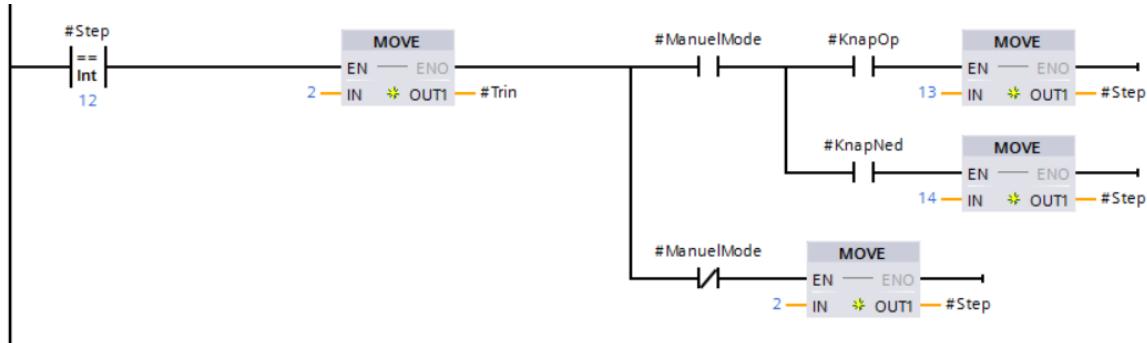
Programmet står og venter efter timeren. Her kan man ved hjælp en trykknap på HMI'et skifte til ManuelMode. Ved et tryk bliver ManuelMode true og Step bliver sat til 12, som svarer til Trin 2 i manuel mode. Se figur 10.10. Hvis en bruger ikke interagerer med systemet, vil inputtet, den unsigned integer SpændingInd, blive sammenlignet med to konstanter 4400 og 3600. Disse to værdier svarer til 10% over og under det ønskede spændingsniveau på 4000mV. Hvis målinger fra Måleenhederne skulle komme udenfor dette interval vil der bliver skiftet trin. Afhængig af om det er et trin ned eller op MOVES henholdsvis 3 eller 4 over i Step. Et trinskift op er vist på figur 10.11.



Figur 10.9: Trin 2 i automatisk mode

I manuel mode er konceptet ikke at trinskiftene skal styres af spændingsmålinger, men af brugeren gennem HMI'et. Der er derfor i stedet for sammenligningerne af SpændingInd placeret to booleans KnapOp og KnapNed. Hvis en af disse bliver true vil programmet skifte trin enten op eller ned. De manuelle skift ligger i Step 13 og 14 for trin 2. I manuel mode trin

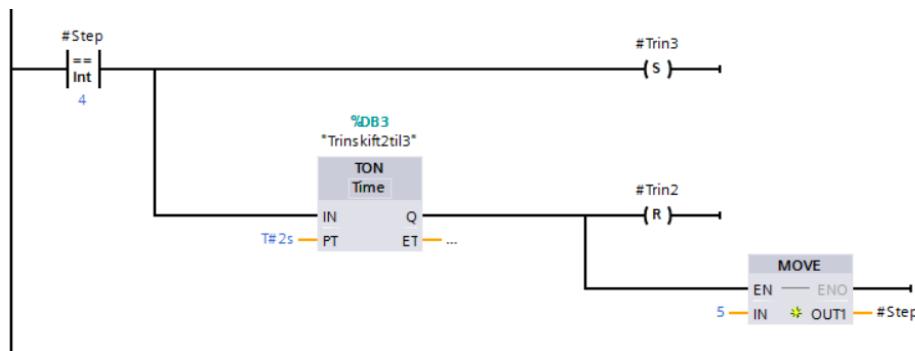
2 er det også muligt for brugeren igen at aktivere Automatisk mode og springe tilbage til Step 2.



Figur 10.10: Trin 2 i manuel mode

Hvordan et trinskift udføres i programmet er gennemgående det samme, forskellen ligger i hvilke trin der skiftes imellem og om det er et skift op eller ned. Dette gælder både for manuel og automatisk mode.

Herunder er derfor kun vist et eksempel på et trinskiftnetværk. De resterende og resten af programmet kan findes i bilag A-2. Dette eksempel er et trinskift op fra trin 2 til trin 3. For at sikre at der altid er forsyning på distributionslinjen skal der altid være et trin på transformeren der er aktivt. Derfor er en on delay timer, TrinSkift2til3, anvendt. Det nye trin, i dette tilfælde trin 3 sættes true, så snart Step bliver 4, mens det gamle trin resettes efter 2 sekunder, hvorefter Step bliver sat til 5 og systemet er nu sikkert og uden afbrydelser blevet flyttet til et nyt trin. For flere kredsløbsdetaljer om skiftet på selve transformeren, henvises til afsnit 8.4.



Figur 10.11: Trin 2 skift et trin op

Det er værd at bemærke at det kun er i trin 2 at systemet kan skifte både op og ned, fordi der kun er 3 trin i systemet. Der kan derfor kun skiftes op fra trin 1 og kun skiftes ned fra trin 3. Grunden til at der er 5 netværk i automatisk mode og 7 i manuel mode, er at trin 1's og trin 3's trinskift ligger i selve trinets netværk. Koden er meget lig det forklarede trinskift, men placeret lige efter sammenligningen af SpændingInd.

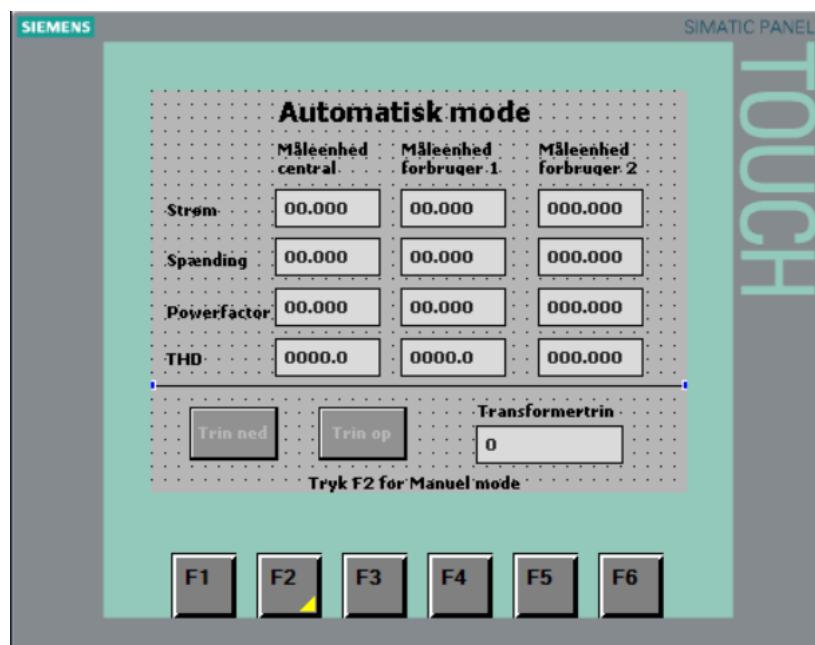
10.2 Brugergrænseflade

Brugergrænsefladen består i dette projekt af en HMI KTP 600. Denne sidder på samme board som PLC'en og kan tilgåes gennem swicthen CSM 1277. Den har to skærme; automatisk og manuel mode, hvoraf automatisk mode er startskærm ved opstart.

Det skal nævnes at systemet først var tiltænkt at have tre Måleenheder, så derfor er det klargjort til tre, men kun de to er brugt i systemet, da det er et proof of concept projekt. Felterne under Måleenhed forbruger 2 er derfor ikke aktive.

10.2.1 Automatisk mode

Formålet med HMI'et var at en bruger skulle kunne observere målinger fra Måleenhederne og følge med i hvilket trintransformer brugte. I automatisk mode, som er vist på figur 10.12, skal en bruger ikke kunne interagere med systemet udover at kunne skifte til manuel mode. Generelt var tanken det skulle være overskueligt for brugeren, derfor er måleværdier og placering opsat som rækker og kolonner. Alt der har med trinskifteren at gøre er blevet placeret nederst, under en skillelinje for at give overblik for brugeren.



Figur 10.12: Screenshot af HMI i automatisk mode

For at skifte til manuel mode trykkes på knappen F2. Knappen er valgt for at det er tydeligt at et tryk ville indebære en stor ændring i programmet. Det er også skrevet på selve skærmen for at gøre en bruger opmærksom på tilstandsskiftet. Måden skiftet er programmeret er gennem eventet Press Key og funktionen ActivateScreen, som kan ses på figur 10.13.

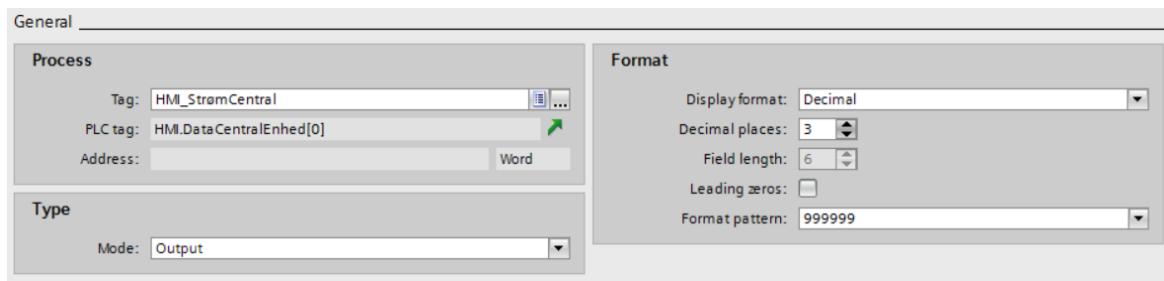


Figur 10.13: Aktiver manuel mode skærmen

Måleværdierne er koblet med de word arrays DataCentralEnhed og DataDecentralEnhed tidligere omtalt i afsnit 10.1.1. Dette sker gennem opsætning af Inoutblokkene. Et eksempel kan ses på figur 10.14 for strømmen ved den centrale måleenhed, hvor det tilknyttede PLC tag er DataCentralEnhed[0]. Plads 0 er altid strømmen, ligegyldigt hvilket array der tales om. Plads 1 er spændingen, plads 2 er THD og plads 3 er powerfactor. Mode er sat til output, dvs. at man ikke kan ændre værdien gennem skærmen. Formatet er et decimaltal med to decimaler før punktummet og tre decimaler efter punktummet. Værdien blokkene er tilknyttet

er for strøm og spaending i milliammpere og millivolt, derfor er kommaet flyttede tre pladser frem, så værdierne der vises er i ampere og volt.

Det forventes ikke at systemet under normale omstændigheder viser tal større end 7 volt, men to decimaler før kommaet er blevet valgt for at systemet kan vise selv meget store målinger, grundet eventuelle fejl på distributionslinjen. Powerfactor er i transmissionen af data også blevet gjort 1000 gange større og derfor flyttes punktummet også tre pladser her. THD er før transmissionen et decimaltal, som er forstørret 1000 gange, så her er punktummet flyttet en plads for at det kan vises i procent for brugeren.



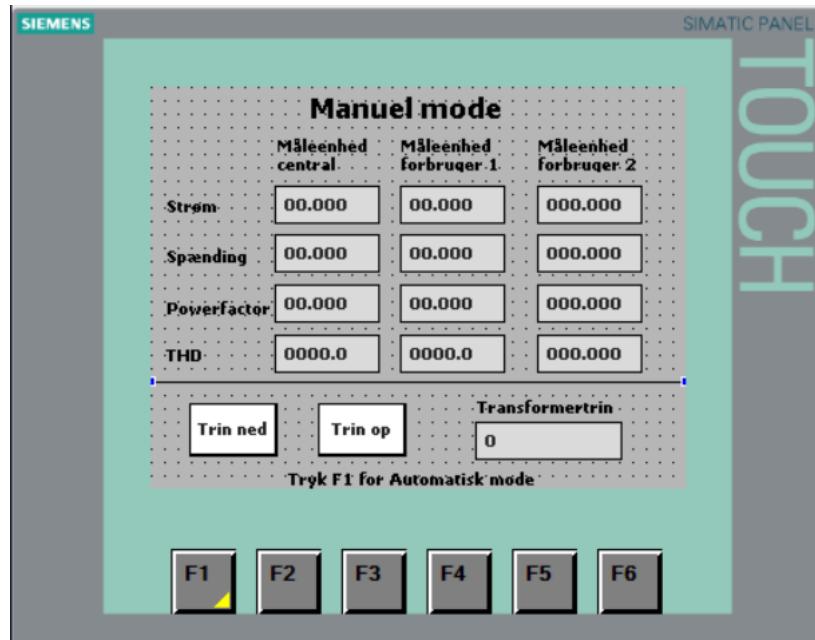
Figur 10.14: Inoutblok for strømmåling ved central Måleenhed

Til visning af transformertrin er også valgt en Inoutblok, i mode output og tilknyttet PLC tagget Trin brugt i FB'en Trinskifter. Trykknapperne Trin ned og Trin op er gråskalerede for at vise at de ikke er anvendelige i dette mode.

10.2.2 Manuel mode

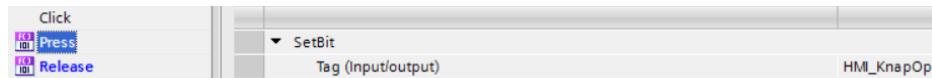
Gennemgående er den samme model brugt til Manuel mode, men der er to forskelle. Den første er at for at skifte til automatisk mode skal bruges knappen F1, hvilket også er tydeliggjort ved en tekst nederst på skærmen.

Den anden er at knapperne Trin ned og Trin Op nu er aktive og kan bruges. Valget af sort tekst på hvid baggrund gør det nemt for en bruger at spotte at knapperne kan bruges. Figur 10.15 viser Manuel mode.



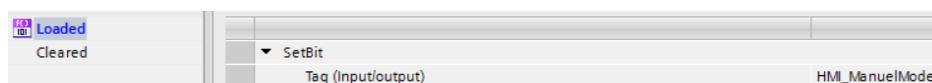
Figur 10.15: Screenshot af HMI i manuel mode

Knappen er sat til at man ved et tryk sætter et bit, her er det tagget HMI_KnapOp, der er tilknyttet PLC tagget KnapOp som anvendes i FB'en Trinskifter. Se figur 10.16. Når knappen slippes er funktionen ResetBit anvendt, som resetter HMI_KnapOp. Trin ned har samme funktionalitet, her er det bare HMI_KnapNed, der anvendes, som er tilknyttet KnapNed i FB'en Trinskifter.



Figur 10.16: Funktionalitet bag Trin op knappen

Ved tryk på knapperne F1 og F2 aktiveres skærmen for, henholdsvis automatisk og manuel mode, men dette sætter ikke programmet i de tilsvarende tilstand. Det sker i øjeblikket den pågældende skærm bliver loaded. Tilstanden styres af et bit der har HMI tagget HMI_ManuelMode og PLC tagget ManuelMode. På figur 10.17 kan man se at når manuel mode skærmen loades kaldes SetBit funktionen. For automatisk mode skærmen kaldes funktionen ResetBit på samme bit.



Figur 10.17: Aktivering af manuel mode når manuel mode skærmen loades

Generelt er skærmene udviklet med henblik på at det er nemt få overblik over dem og at de er nemme at forstå. Der er valgt at ligge vægt på få få muligheder, fordi det er et system som der skal sikres mod forkert brug, da det kan få store konsekvenser.

10.3 Kommunikationsmodul

Kommunikationsmodulets egenskaber er at indsamle data fra måleenheder og sende dem videre til kontrolmodulet. Kommunikations modulet består af en Arduino Mega2560 med

et tilknyttet Ethernet Shield R3 [1]. Dette shield er designet med udgangspunkt i W5100-microchippet, der er en standard ethernet controller chip, se bilag B-3 for datasheet på chippen. Dette ethernet shield bliver opsat fra Arduinoen igennem en SPI forbindelse, der oprettes i Arduino koden.

Kommunikationsmodulet efterspørger måledata fra måleenheden over en UART forbindelse. Da der kan kobles mange enheder op på samme UART linje, vil kommunikationsmodulet være i stand til at hente information fra så mange måleenheder som ønskes. Hvis der kobles flere måleenheder på samme UART linje kræver det blot en lille ændring i protokollen, således der ikke opstår situationer hvor 2 enheder sender data retur samtidig.

Herunder ses et eksempel af koden, hvor kommunikationsmodulet modtager data fra måleenhederne. For at få en bedre opløsning i målingerne bliver der sendt 16 bit, som er realiseret ved at dele 16bit op i to 8bits:

Listing 10.1: Modtagelse

```

    Serial1.write('A');

    while(read1!=1);
    read1 = 0;

    MCStrom = Serial1.read();
    delay(3);
    MCStrom1 = Serial1.read();

```

For hver måledata er der oprettet 2 integers, der skrives til individuelt. Dette ses med MCStrom og MCStrom1, der tilsammen giver måleenhedens strømværdi i milliampere. Den variable read1, der ses i kodeeksemplet 10.1 er interruptstyret. Interruptet er benyttet for at vente på måleenheden får samplet en hel periode. Interruptet er implementeret på følgende måde i Arduinokoden, se bilag A-3 for at se hele Arduinokoden:

Listing 10.2: Interrupt

```

// Skrevet i den globale del af koden:

int receive()
{
    read1 = 1;
}

// Skrevet i setup-delen:

attachInterrupt(digitalPinToInterruption(19), receive, RISING);

```

Funktionen attachInterrupt() er en allerede implementeret funktion i Arduino udviklingsværktøjet. Funktionen skal have 3 parametre, hvor den første er lidt speciel, da den skal hedde digitalPinToInterruption(pin), hvor pin er nummeret på arduinopinen der skal interrupts. Den anden parameter er navnet på Interrupt Service Routinen der skal køres når der fås et interrupt på pinen. Den sidste parameter bestemmer om interruptet skal triggeres på en lav værdi, et skift i tilstand, en rising edge eller falling edge. Der er i dette tilfælde valgt et interrupte på rising edge, da det er en UART-modtagelse der skal interrupts på.

Opsætningen af ethernet på Arduinoen er gjort mulig ved brug af R3 Ethernet Shieldet. W5100 chippen på Ethernet Shieldet bliver opsat igennem en SPI forbindelse der opsættes af SPI.h biblioteket. Ethernet.h biblioteket benytter denne SPI forbindelse til at opsætte socketen på Ethernet Shieldet med de parametre, der bliver givet i Arduino koden [2]. Klassen EthernetServer opretter et element ved navn Server, der får parametren på hvilken port der skal aflyttes for forbindelse. Her er der valgt port 27015. Den mac adresse der indgår i koden

skal være den der svarer til Ethernet Shield enheden. IP-adressen opsætter efter hvilken IP der ønskes, her er valgt 192.168.0.129, fordi PLC og HMI standard er på 192.168.0.xx netværket. Herunder på kodeeksempel 10.3 ses opsætningen af Ethernet kommunikation på Arduinoen, se bilag A-3 for at se hele koden:

Listing 10.3: Ethernet

```
// Skrevet i den globale del af kodens
#include <SPI.h>
#include <Ethernet.h>

byte mac[] = {0x90, 0xA2, 0xDA, 0x0F, 0x1B, 0x82}; // MAC Address
byte ip[] = {192, 168, 0, 129}; // Network Address

EthernetServer server = EthernetServer(27015);

// Skrevet i setup delen af kodens
Ethernet.begin(mac, ip);
server.begin();
```

Arduinoen modtager to forskellige kommandoer fra PLC'en én for hver af måleenhederne. De to kommandoer er henholdsvis req1 og req2. Arduinoen sammenligner hvert eneste char, for at vide hvilken måleenhed der skal sendes data fra. Se bilag A-3 for at se hele koden.

Listing 10.4: Ethernet modtagelse

```
if(client)
{
    if(client.read() == 'r')
    {
        if(client.read() == 'e')
        {
            if(client.read() == 'q')
            {
                char m = client.read();
                if(m == '1')
                {
                    client.write(MCStrom);
                    client.write(MCStrom1);
                    client.write(MCSpanding);
                    client.write(MCSpanding1);
                    client.write(MCTHD);
                    client.write(MCTHD1);
                    client.write(MCpf);
                    client.write(MCpf1);
                }

                if(m == '2')
                {
                    client.write(M1Strom);
                    client.write(M1Strom1);
                    client.write(M1Spanding);
                    client.write(M1Spanding1);
                    client.write(M1THD);
                    client.write(M1THD1);
                    client.write(M1pf);
                    client.write(M1pf1);
                }
            }
        }
    }
}
```

} } }

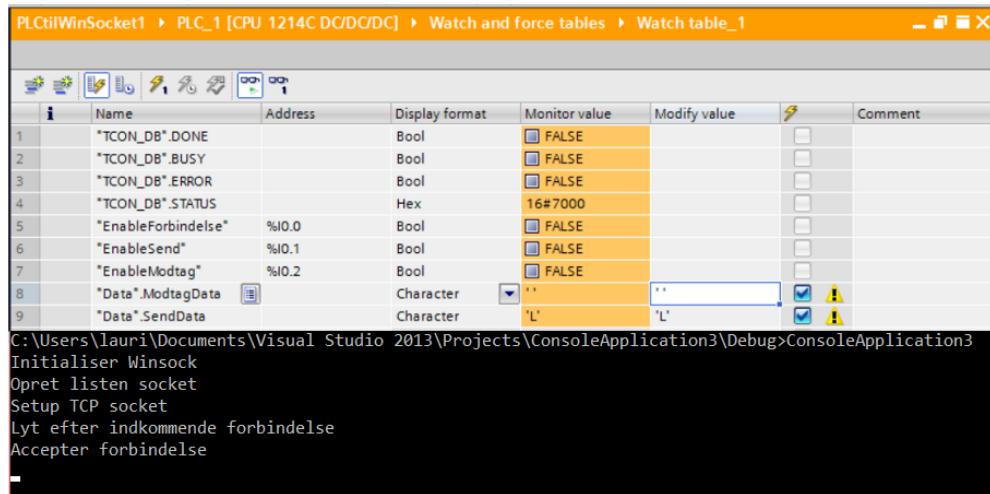
10.4 Modultest

10.4.1 Kontrolmodul

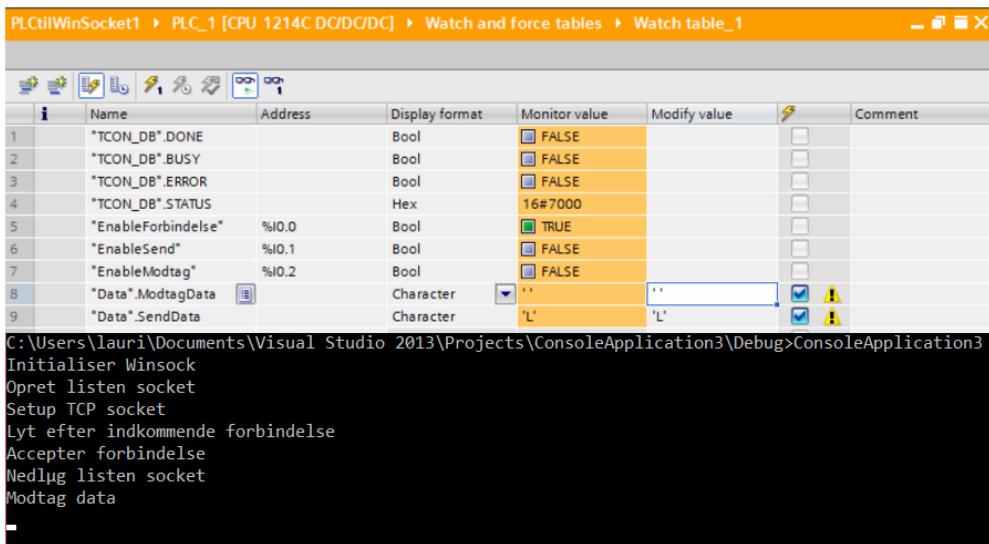
Test af PLC programmet som aktiv part i oprettelse af TCP forbindelsen

Test	Opret forbindelse fra en TCP client på PLC'en til en TCP server på en Windows PC.
Testbeskrivelse	Funktionen OpretForbindelse i PLC programmet testes ved hjælp af et en testserver WinSock. Se bilag A2[5]. Testen overvåges ved at gå online på PLC'en og gennem udskrifter fra Winsock programmet i en kommandoprompt på PC'en. EnableForbindelse styres af en switch forbundet til indgangen I0.0 i PLC'en.
Input	Et on switch på I0.0.
Forventet output	Clienten vil være aktiv i at oprette forbindelse, hvilket vil betyde at serveren udskriver at den har nedlagt listen socket og er klar til at modtage data.
Resultat	EnableForbindelse bliver sat true og i kommandoprompten udskriver serveren at listen socket er nedlagt og at den er klar til at modtage data. Se figur 10.18 og figur 10.19.

Tabel 10.1: Oprettelse af TCP forbindelse, PLC



Figur 10.18: Før I0.0 swicthes on

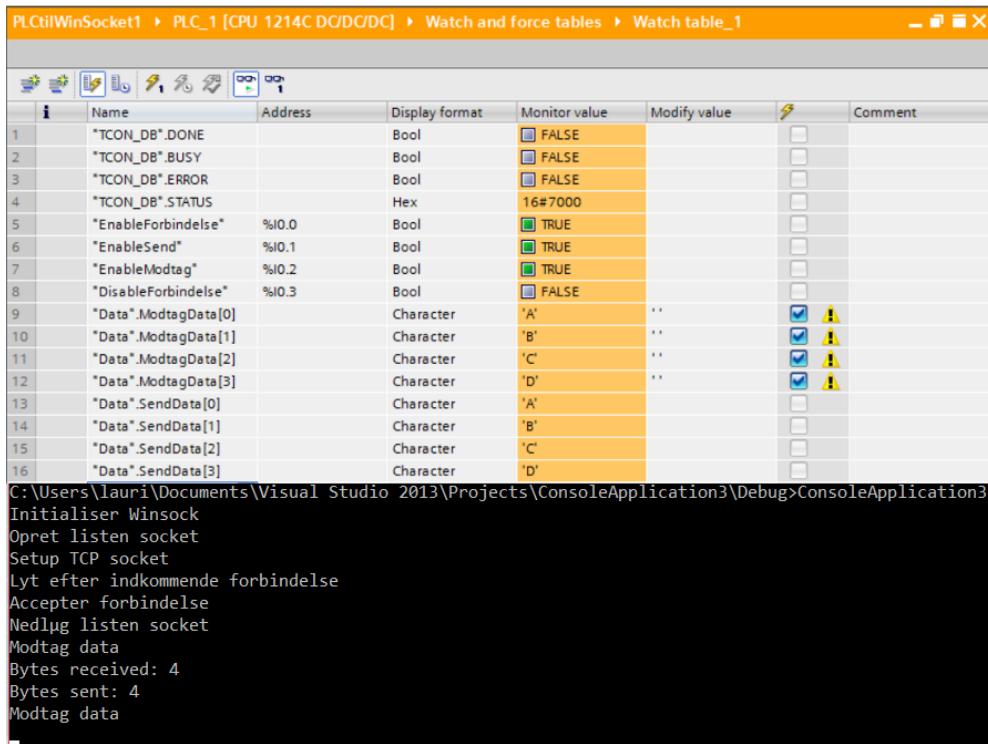


Figur 10.19: Ffter I0.0 swicthes on

Test af PLC programmets datatransmission over TCP forbindelsen

Test	Send data fra en TCP client på PLC'en til en TCP server på en Windows PC. Samme data sendes retur fra server til client.
Testbeskrivelse	Funktionerne SendData og ModtagData i PLC programmet testes ved hjælp af et en test server Winsock Se bilag A2[5]. Testen overvåges ved at gå online på PLC'en og gennem udskrifter fra Winsock programmet i en kommandoprompt på PC'en. EnableSend og EnableModtag styres af to switches forbundet til indgangene I0.1 og I0.2.
Input	Et on switch på I0.0, I0.1 og I0.2.
Forventet output	Char arrayet 'A', 'B', 'C', 'D' vil blive sendt og modtaget igen. Det kan ses på udskrifter at i kommandoprompten at 4 bytes er modtaget og sendt igen, hvorefter serveren er klar til at modtag ny data. På PLC'en kan det ses at de 4 sendte bytes matcher de 4 modtagne.
Resultat	EnableForbindelse, EnableSend og EnableModtag bliver sat true og i kommandoprompten udskriver serveren at 4 bytes er modtaget og 4 er bytes er sendt og at den er klar til at ny modtag data. PLC'en har de samme 4 bytes i ModtagDat arrayet som i SendData arrayet. Se figur 10.20

Tabel 10.2: PLC datatransmission TCP

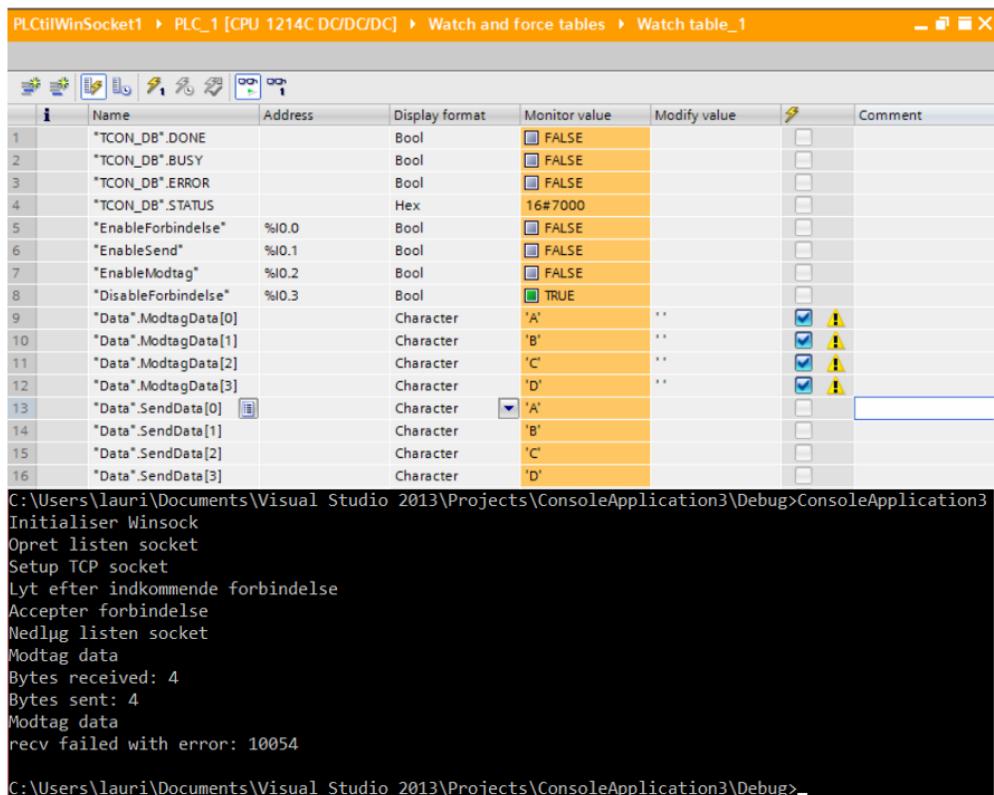


Figur 10.20: Test af ModtagData og SendData

Test af PLC program som aktiv part i nedlæggelse af TCP forbindelsen

Test	Nedlæg forbindelsen mellem en PLC client og en Windows server.
Testbeskrivelse	Funktionen AfslutForbindelse i PLC programmet testes ved hjælp af en test server Winsock Se bilag A2[5]. Testen overvåges ved at gå online på PLC'en og gennem udskrifter fra Winsock programmet i en kommandoprompt på PC'en. DisableForbindelse styres af to swicthes forbundet til indgangen I0.3.
Input	Et on switch på I0.3
Forventet output	Winsock programmet vil udskrive en fejl, fordi det har mistet forbindelsen.
Resultat	DisableForbindelse bliver sat true og i kommandoprompten udskriver serveren at der skete en fejl i modtagelse af ny data og udskriver fejlkode. Se figur 10.21

Tabel 10.3: Nedlæggelse af TCP forbindelse, PLC



Figur 10.21: Test af AfslutForbindelse

Test af PLC'ens udgange styret af digitale signaler

Test	PLC udgangene til relæer leverer 24V, som skal til for at relæerne slår.
Testbeskrivelse	Rælerne der bruges til at styre hvilket transformatortrin der benyttes skal have 24V for at trække. Derfor testes det at udgangene også giver 24V.
Input	
Forventet output	Det forventes at der kan måles 24V med et multimeter.
Resultat	Spænding målte på PLC udgang: 24.07 V, 23,97 V, 24.08 V

Tabel 10.4: Digital styring af PLC udgange

10.4.2 Brugergrænseflade

Test af HMI softwareens interaktion med PLC programmet

Test	HMI'en udskriver random generede tal der sendes til PLC'en fra PSoC'en.
Testbeskrivelse	PSoC'en er opsat med en speciel kode se figur 10.22, der genererer tilfældige værdier for strøm, spænding, powerfactor og THD.
Input	Måledata over fra PsoC'en til PLC'en, hvor HMI'en kan hente data fra og vise dem på skærmen.
Forventet output	HMI'en opdateres med nye tilfældige værdier kontinuerligt.
Resultat	Måleværdierne på skærmen bliver opdateret en gang hver 2s. 10.23

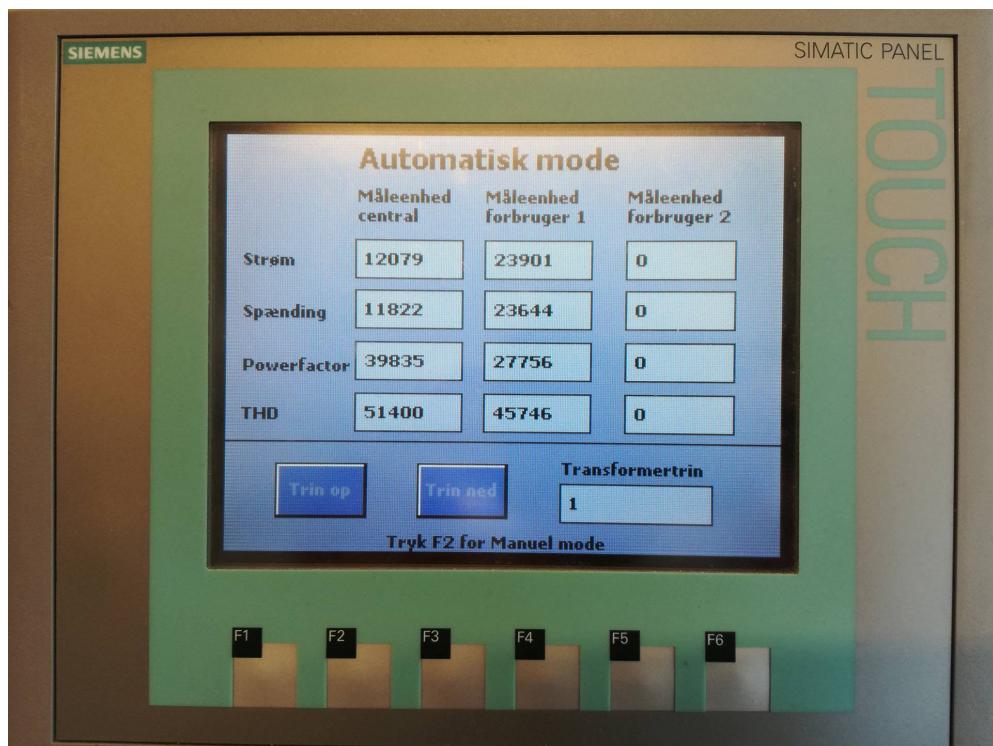
Tabel 10.5: Interaktion mellem HMI og PLC

```

54
55
56    for(;;)
57    {
58        Strom1++;
59        Spanding++;
60        pf++;
61        THD++;
62        if(Strom1 > 10) Strom1=0;
63        if(Spanding >10) Spanding=0;
64        if(pf>10) pf=0;
65        if(THD>10) THD=0;
66    }

```

Figur 10.22: PSoC tæller op og sender derfor tilfældige data



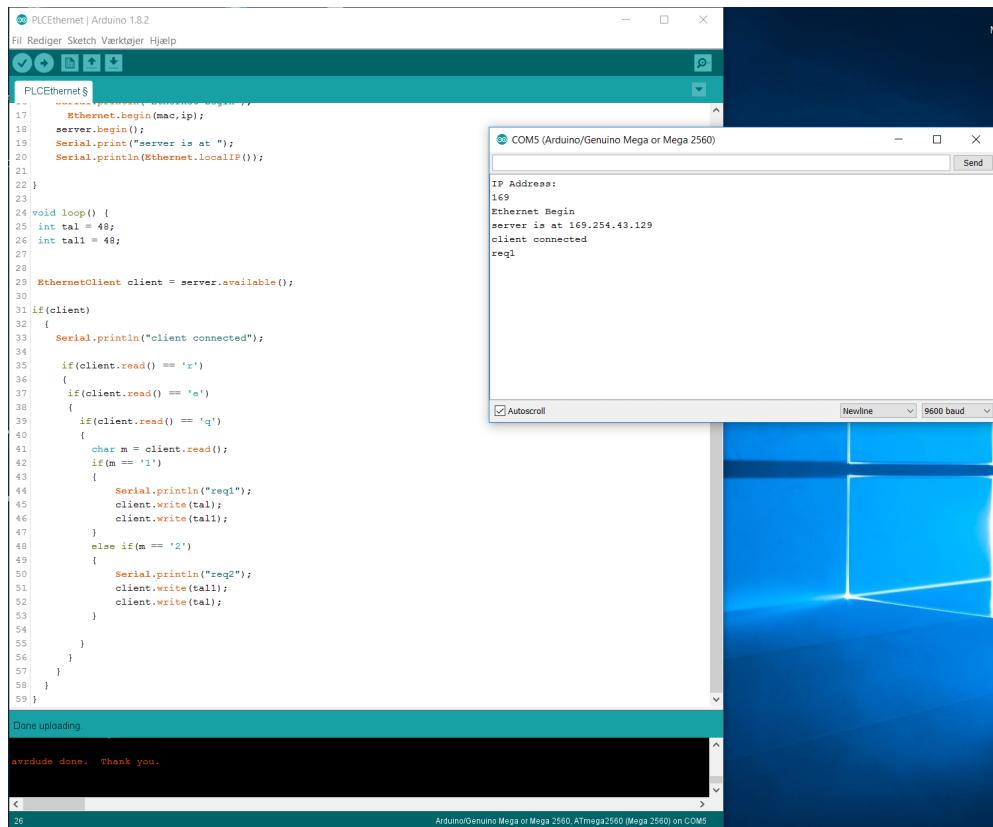
Figur 10.23: HMI'en viser tilfældigt generede værdier

Test	Arduino'en modtager 'req1' over Ethernet fra PLC'en.
Testbeskrivelse	Før Arduino'en sender værdierne til PLC'en skal den modtage et request på de nye værdier, det testes her om Arduinoen med denne request kommando rigtigt.
Input	PLC'en udsender request kommandoen 'req1' over Ethernet.
Forventet out-put	Arduinoen vil udskrive req1 på det serielle vindue, der kan åbnes i udviklingsprogrammet.
Resultat	Arduinoen modtager 'req1' og udskriver det på det serielle vindue. Se figur 10.24

Tabel 10.6: Arduinoens datatransmission TCP

10.4.3 Kommunikationsmodul

Test af Arduinoens datatransmission over TCP forbindelsen

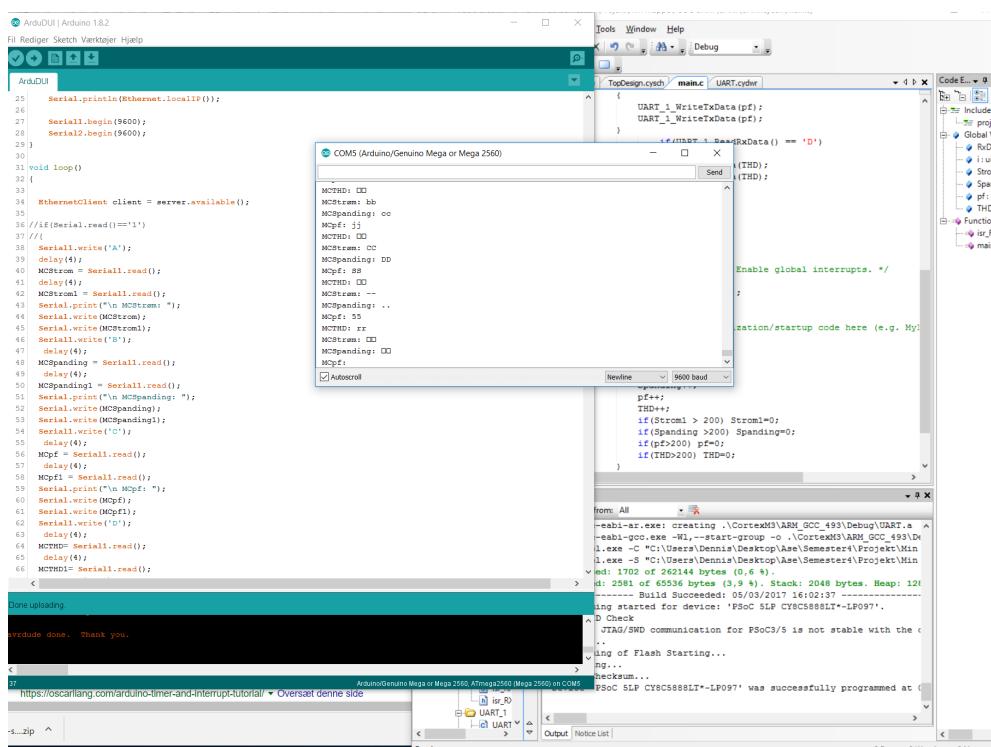


Figur 10.24: Test af request modtagelse

Test	Arduino'en kan modtage værdi over UART fra PSoC'en.
Testbeskrivelse	Arduino'en skal modtage måledata fra PSoC'en for 4 forskellige målinger. Der bliver holdt styr på hvilke måledata der kommer hvornår, ved at Arduinoen efterspørger hver enkelt måledata over UART linjen, med 'A', 'B', 'C' eller 'D'. Dette ses på figur 10.25, hvor Serial.write() skriver til terminal vinduet, og Serial1.write() skriver til PSoC'en.
Input	Arduinoen udsender 'A', 'B', 'C' og 'D' i et loop. Der modtages derfor måledata, der tilfældig genereret på PSoC'en.
Forventet out-put	Arduinoen vil udskrive de modtagede tilfældige måledata på terminalvinduet
Resultat	Arduinoen modtager udskriver måledata i terminalvinduet. Se figur 10.25

Tabel 10.7: Arduinoens datatransmission UART

Test af Arduinoens datatransmission over UART forbindelsen



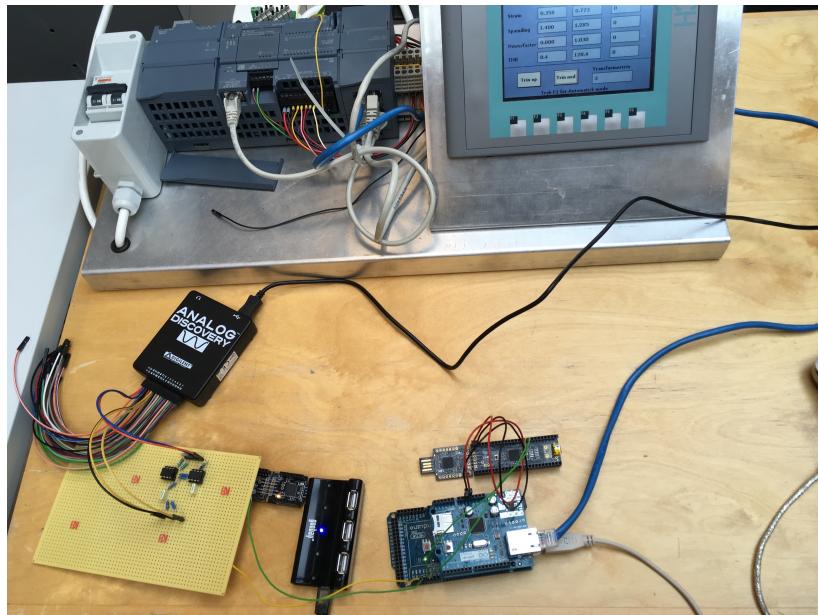
Figur 10.25: Test af måledata modtagelse

Kapitel 11

Integrationstest

11.1 Integrationstest mellem Måleenhed og Styringsenhed

I dette afsnit testes kommunikationen mellem Måleenhed og Brugergrænsefladen. Der vil blive testet af de rigtig tal vises på skærmen, hastigheden mellem opdatering af visninger samt pålideligheden af dokumentationen. Testopstilling kan ses på figur 11.1.



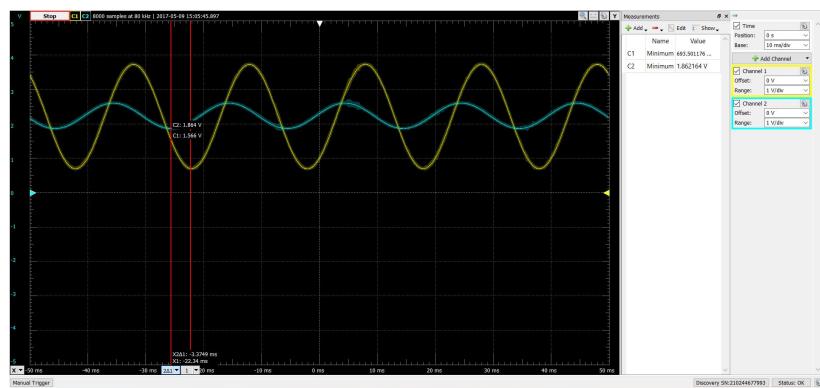
Figur 11.1: Testopstilling af integrationstest mellem Måleenhed og Styringsenhed

Til testen skal der bruges:

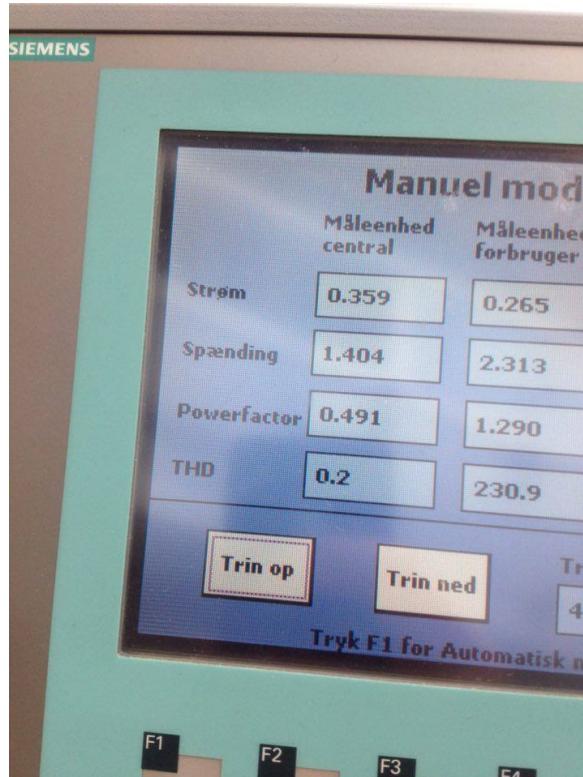
- 1 stk PSOC
- 1 stk Arduino med ethernet shield
- 1 stk PLC med hmi skærm
- Ledning fra PSOC's UART til Arduino
- Ethernet ledning fra ethernet shield til PLC
- Analog Discovery

Test	Brugergrænsefladen modtager korrekte tal
Testbeskrivelse	Der testes at de værdier, der kommer ind på Måleenheden er ens med værdierne der vises på brugergrænsefladen.
Input	Analog Discovery anvendes til at lave 4VPP sinus på spænding indgang, 1VPP sinus på strøm indgang, 3,3ms forskydning
Forventet output	1,414V spænding, 0,354A strøm, 0,509 PF se figur 11.2, 0 THD
Resultat	1,404V spænding, 0,359A strøm, 0,491, 0,2 THD, se figur 11.3 for resultat af skærm

Tabel 11.1: Integrationstest 1



Figur 11.2: Visning af 3,3ms forsinkelse mellem strøm og spænding



Figur 11.3: Resultatet af visning på brugergrænseflade

Test	Forsinkelse på brugergrænsefladen
Testbeskrivelse	Tiden der går fra mellem ændringer på skærmen
Input	4VPP
Forventet out-put	At der maks. går 2,5 sekund, pga intern delay i PLC og kommunikation
Resultat	Der er målt 5 gange: 2,01s 2,00s 1,99s 2,00s 1,95s, der alle har vist sig at være inden for kravet.

Tabel 11.2: Integrationstest 2

Test	Kommunikations pålidelighed
Testbeskrivelse	Der observeres på brugergrænsefladen i 1 min og tælles uregelmæssigheder
Input	4VPP sinus på spænding indgang, 1VPP sinus på strøm indgang
Forventet out-put	5% fejl ud af 120 inputs
Resultat	5 fejl dvs. 4.17%

Tabel 11.3: Integrationstest 3

11.2 Samlet integrationstest

Denne test indeholder alle de enkelte moduler, som systemet består af, herunder Brugergrænseflade, Måleenhed, Styringsenhed og Trinskifter.

I dette afsnit testes at udregninger for hvordan nettet opfører sig stemmer overens med det, der måles med Måleenheden, og det der vises på brugergrænsefladen. Testene laves ved at måle på nettet med Måleenheden, som er forbundet til brugergrænsefladen, så resultaterne for målinger kan aflæses på skærmen. Værdierne sammenholdes med resultaterne fra simulering og beregning.

Test	Udregning og simulering passer med systemets værdier.
Testbeskrivelse	Der testes ved at forsyne nettet med trinskifteren og derefter måle spænding, strøm og Pf med Måleenheden ved en forbruger
Input	4Vrms fra trinskifter, der tilsluttes 54Ω belastning efter distributionslinjen
Forventet out-put	At resultaterne i tabel 11.5 stemmer overens
Resultat	Se tabel 11.5

Tabel 11.4: Integrationstest 4

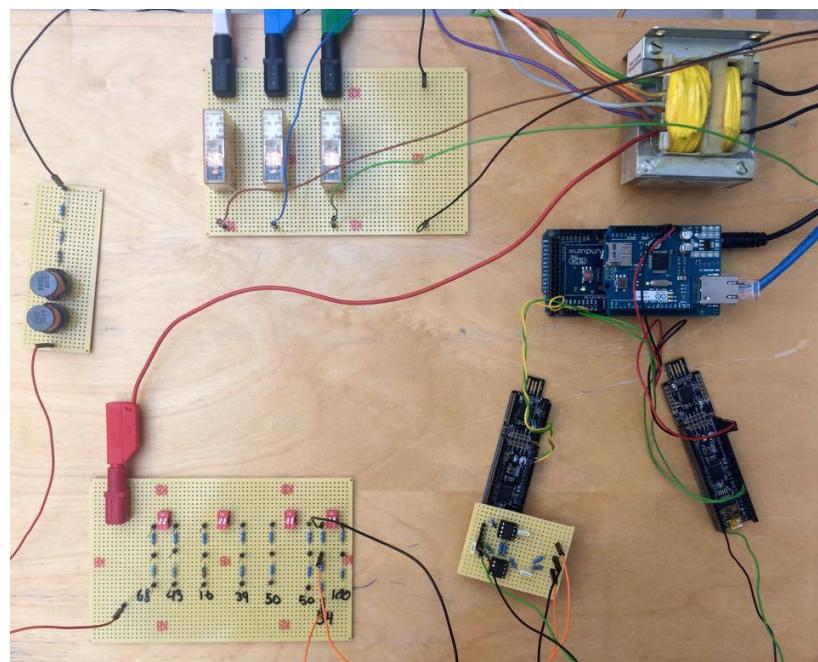
Test	Spænding	Strøm	PF
Simulering	3,580V	0,066A	0,997
Beregnet	3,600V	0,066A	0,996
Målt	3,379V	0,058A	0,999

Tabel 11.5: Resultat af Integrationstest 4

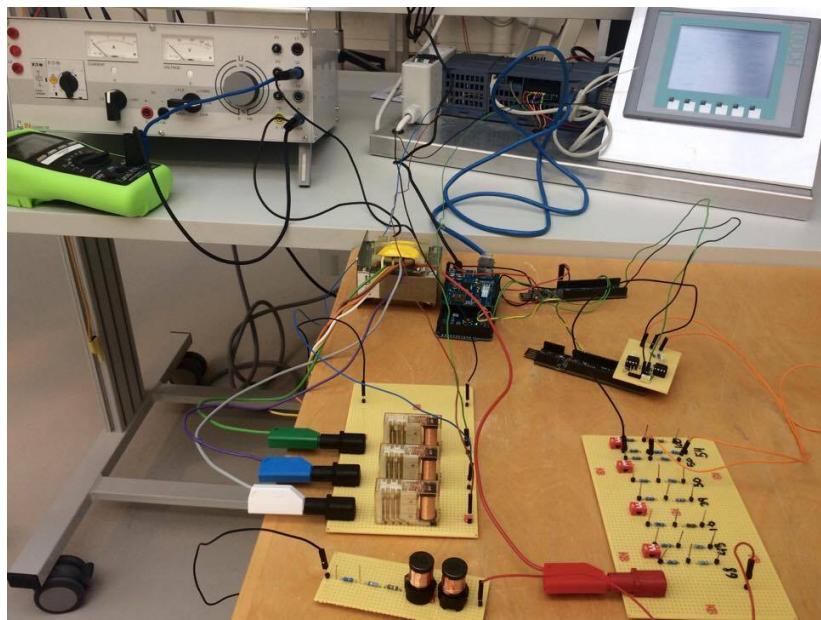


Figur 11.4: Resultat af integrationstest 4, målteværdier.

Opstillingen til den samlede integrationstest kan ses på figur 11.5 og 11.6



Figur 11.5: Opstilling af samlet system uden PLC



Figur 11.6: Opstilling af samlet system med PLC

Kapitel 12

Accepttest

12.1 Funktionelle Krav

UC1	Handling	Forventet resultat	Resultat	OK
Start manuel styring	Bruger vælger Manuel Mode	På skærmen vises Manuel Mode med mulighed for valg af trin	Skærmen viser Manuel Mode	✓

UC2	Handling	Forventet resultat	Resultat	OK
Stop manuel styring	Bruger vælger Automatisk Mode	Skærmen viser Automatisk Mode, hvor trinnapperne er deaktiverede	Der vises Automatisk Mode	✓

UC3a	Handling	Forventet resultat	Resultat	OK
Skift trin op	Systemet er i Manuel Mode. Bruger vælger Trin Op på skærmen	Systemet skifter trin op på transformeren og skærmen opdateres med nye værdier.	Transformeren skifter trin efter 2s. Derefter opdateres spænding, strøm, pf og THD på skærmen inden 2s.	✓

UC3b	Handling	Forventet resultat	Resultat	OK
Skift trin ned	Systemet er i Manuel Mode. Bruger vælger Trin Ned på skærmen	Systemet skifter trin ned på transformeren og skærmen opdateres med nye værdier.	Transformeren skifter trin efter 2s. Derefter opdateres spænding, strøm, pf og THD på skærmen inden 2s.	✓

12.2 Ikke funktionelle krav

Trintransformer	Handling	Forventet resultat	Resultat	OK
Nominel spænding på primærsiden er 24VAC	Spændingen på primærsiden måles.	Den målte værdi er 24VAC.	Der måles 23.88 VAC	✓
Nominel spænding på sekundær side er 4, 5 eller 6 VAC afhængig af trin	Spændingen på sekundær siden måles for hhv. trin 4, 5 og 6.	Den målte værdi er 4, 5 eller 6 VAC alt efter valgt trin.	Der måles hhv. 4, 5 og 6 VAC	✓
Skal minimum kunne leve 500mA .	Strømmen på sekundær siden måles.	Transformeren kan leve over 500mA	Ikke opfyldt	

Belastning	Handling	Forventet resultat	Resultat	OK
Modstandsværdi på 54Ω giver spændingsfald på 10%, når spændingen fra regulatoren er 4V.	Den givne modstand indsættes som belastning, og spændingen herover måles.	Spændingen over belastningen måles til 3,6V.	3,4V	✓

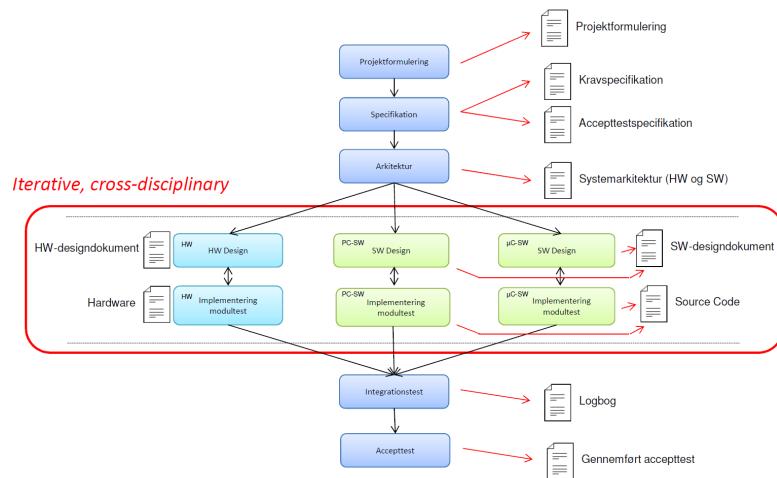
Måleenhed	Handling	Forventet resultat	Resultat	OK
Måle spændingen ved trinskifteren og forbrugerne mellem 0 og 8 Vrms	Måleenheden testes med spændinger fra 0 til 8Vrms, i intervaller af 500mV.	Korrekt spændingsmåling i hele intervallet.	Korrekte spændingsmålinger i hele intervallet	✓
Måle spændingen med en præcision på $\pm 5\%$	Måleenheden påtrykkes en spænding på 3,5Vrms. Der laves herefter ti målinger	Gennemsnits afvigelsen forventes at være under $\pm 5\%$.	Afvigelsen er fundet til 1,6%.	✓
Måle strømmen ved trinskifteren og forbrugerne mellem 0 og 500mA	Måleenheden testes med strømme fra 0 til 500mA i intervaller af 50mA	Korrekt strømmåling i hele intervallet.	Korrekt strømmåling i hele intervallet	✓
Måle strømmen med en præcision på $\pm 5\%$	Måleenheden påtrykkes en spænding på 300mVrms (Svarrende til 300mA), der laves herefter ti målinger	Gennemsnits afvigelsen forventes at være under $\pm 5\%$	Afvigelsen er fundet til 0,8%.	✓

Måle og beregne power factor med en præcision på $\pm 5\%$	Måleenheden mäter power factor over en belastning på distributionslinjen, der sammenlignes med beregnet power factor	Afvigelsen forventes at være under $\pm 5\%$	Afvigelsen er fundet til 2%	✓
Beregne THD med en præcision på $\pm 5\%$	Måleenheden påtrykkes en firkantsignal med 1V amplituder og 1V offset. Der sammenlignes med beregnet THD for firkantsignal	Afvigelsen forventes at være under $\pm 5\%$	Afvigelsen er fundet til 0,5%	✓

Kapitel 13

Metode

Til gennemførelse af dette projekt er ASE-modellen blevet anvendt som udgangspunkt. Denne model viser de forskellige faser, projektgruppen skal igennem på vejen mod et endeligt produkt og en veldokumenteret og gennemarbejdet rapport. ASE-modellen ses på figur 13.1. Gruppen har desuden anvendt en iterativ og empirisk arbejdsmetode. Der er brugt faglitteratur og teori fra undervisningen til at danne vidensgrundlag for projektet om spændingsregulatoren. Undervejs er der indsamlet nye erfaringer i forbindelse med udviklingen af produktet. I udviklingsfasen er der anvendt SysML og UML, der sammen med Use Casene har givet overblik over systemet både grafisk og skriftligt. Dette har dannet grundlag for opbygningen af systemet.



Figur 13.1: Asemodellen

For at styre og lede projektet har gruppen ladet sig inspirere af Scrum principperne. Fra starten af projektet har gruppen oprettet sprints og defineret dertilhørende opgaver. Gruppen har ikke afholdt daily scrums, men har gennemsnitligt haft 1-2 ugentlige gruppemøder, samt ugentligt vejledermøde. Dette har medført en jævn arbejdsgang og et godt flow i projektet.

Bibliografi

- [1] *Arduino Ethernet Shield R3*. URL: <https://www.arduino.cc/en/Main/ArduinoEthernetShieldV1> (sidst set 13.04.2017).
- [2] *Arduino Ethernet.h library*. URL: <https://www.arduino.cc/en/reference/ethernet> (sidst set 13.04.2017).
- [3] *Cooley and Tukey FFT algoritme*. URL: https://en.wikipedia.org/wiki/Cooley-%E2%80%93Tukey_FFT_algorithm (sidst set 20.05.2017).
- [4] *Cypress forum*. URL: <http://www.cypress.com/forum/psoc-5-device-programming/how-implement-fft-psoc5> (sidst set 10.04.2017).
- [5] *WinSock server testprogram*. URL: [https://msdn.microsoft.com/en-us/library/windows/desktop/ms737591\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/ms737591(v=vs.85).aspx) (sidst set 20.04.2017).