Høgskulen på Vestlandet

BACHELOROPPGAVE:

BO19E-24 LOKAL VENTILASJONSSTYRING TUNNEL

Andre Lyngset
Barbara Andrea Glejzer Lamela
Benedikte Martine Thorvaldsen

30.mai.2019

Dokumentkontroll

Rapportens tittel: BO19E-24 Lokal ventilasjonsstyring tunnel	*Dato/Versjon* 30. mai. 2019/1.0
	Rapportnummer: B019E-24
Forfatter(e): Andre Lyngset Barbara Andrea Glejzer Lamela Benedikte Martine Thorvaldsen	*Studieretning:* EAU16
	Antall sider m/vedlegg 54
Høgskolens veileder: Inge Vivås	*Gradering:* Åpen
Eventuelle Merknader: Vi tillater at oppgaven kan publiseres.	

Oppdragsgiver: Goodtech	*Oppdragsgivers referanse:*
Oppdragsgivers kontaktperson(er) (inklusiv kontaktinformasjon): Kenneth Herskedal	

Revisjon	**Dato**	**Status**	**Utført av**
0.1	08.02.19	Forstudie	Alle
0.2	27.02.19	Utvidelse av dokumentet Lagt til delkapitler	Alle
0.3	10.04.19	Endret rekkefølgen på ulike kapitler	Alle
0.4	15.05.19	Endret kapittelet «endelig hovedprogram»	Alle
0.5	24.05.19	Laget ferdig utkast	Alle
1.0	30.05.19	Ferdigstilling av oppgaven	Alle

Forord

Denne oppgaven markerer avslutningen på vårt bachelorprogram i automatiseringsteknikk ved Høgskulen på Vestlandet, avdeling Bergen. Arbeidet har vart siden januar 2019 og ut vårsemesteret. Vårt arbeid har i hovedsak bestått av programmering av PLS, praktisk testing og teoriarbeid. Dette er rapporten vi har skrevet under oppgaven.

I arbeidet med oppgaven har vi hatt nær kontakt med både intern og ekstern veileder. Dette er noe vi er svært takknemlige for. Vår interne veileder var Inge Vivås, som vi ønsker å takke for god oppfølging og råd underveis i arbeidet. Vår eksterne veileder var Kenneth Herskedal fra Goodtech. Vi ønsker å takke for all den kunnskapen han har delt med oss.

Videre vil vi takke Svein Borlaug, Kjetil Vatsøy og Trond-Helge Lie hos Goodtech for råd og god hjelp med ulike problemer som dukket opp underveis. Vi vil også rette en samlet takk til alle ansatte hos Goodtech på Damsgård for å ha tatt imot oss med åpne armer de gangene vi har vært der og jobbet.

Vi ønsker også å takke Lars Manger Ekroll for hjelp med å skaffe komponenter og øke forståelsen vår for virkemåten deres. Vi fikk også muligheten til å være med på lab sammen med klassen hans, dette er noe vi satte stor pris på.

Ønsker også å rette en takk til Ingrid Jørgensen og Tone Breistrand for korrekturlesning av oppgaven.

Til slutt vil vi takke våre kjære familier som har holdt ut med oss i løpet av arbeidet. Vi har mottatt mye støtte fra dem i en stressende studenthverdag.

Bergen, mai 2019.

Sammendrag

Med lokal ventilasjonsstyring i tunnel menes at det skal være lett å drifte hver enkelt tunnel, hver for seg. Styringen er også knyttet opp mot en hoved PLS som sender beskjeder fra vegtrafikksentralen og kan styres derfra.

Oppgaven beskriver at dagens løsning skal erstattes med en sikker logisk styring, som vil si en PLS, samt erstatte tavler og den gamle betjeningen av viftene med et HMI-panel. Dette er for å oppgradere systemet til å bli mer moderne, samtidig som driftssikkerheten opprettholdes.

Det første vi gjorde i arbeidet med oppgaven var å diskutere med oppdragsgiver om hvilken løsning de så for seg. Da fikk vi vite at vi kunne velge selv hvilket språk vi skulle bruke for å programmere PLS-programmet, og utforming av HMI-panelet. Hovedkravet til HMI-utformingen var at det skulle være oversiktlig og enkelt.

Videre i oppgaven undersøkte vi teori til ulike komponenter som var relevante for oppgaven. Relé, modbus, kontaktor og Mean Time Between Failure er noen av temaene vi har fordypet oss i. Vi har også fått vært med Elkraft på lab for å lære litt mer om virkemåten til kontaktorer i praksis.

Når vi hadde fått på plass relevant teori, startet arbeidet med planlegging og programmering av PLS-programmet og HMI-panelet, før vi startet med beregninger av MTBF av komponentene.

Etter arbeidet med denne oppgaven har vi fått et innblikk i hvordan man planlegger et slikt prosjekt med tanke på arbeids- og tidsfordeling. Vi har også lært å konfigurere modbus TCP/IP i PLS-programmet, der verdier blir lagret i et register. Vi har også opparbeidet oss erfaring med kontaktorer og deres virkemåte.

1 Innhold

2 Figurliste

1 Innledning og oppgavebeskrivelse

1.1 Innledning

«Automatisering er teknikken til å få systemer til å fungere uten, eller med liten grad av menneskelig medvirkning», er definisjonen av automatisering ifølge Store Norske Leksikon [1]. I dagens samfunn kan man se automatiserte systemer overalt hvor man går, alt fra når døren i bygget du skal inn i åpner seg automatisk til din månedlige lønn utbetales automatisk.

Oppdragsgiveren for oppgaven er bedriften Goodtech, som ble grunnlagt i 1913 under navnet Norsk Elektrisk Kabelfabrikk. De er en teknologileverandør innenfor automatisering, industriteknikk og miljøtenkning hvor de jobber med alt fra konsulentbaserte tjenester til totalentrepriser innenfor disse fagfeltene. De består av ca. 350 medarbeidere som jobber i Norge, Sverige og Åland [2] [3].

Mange ulike teknologier har blitt og blir fremdeles brukt i automatiseringsteknikk. Eksempler på disse er mekaniske, pneumatiske (luftdrevne) og elektroniske systemer. Den sistnevnte har vokst mye de siste tiårene og datamaskiner er blitt en viktig komponent i mange systemer. PLS er en datamaskin som blir brukt i mange ulike systemer, som for eksempel i et trafikklys eller for kontrollering av nivået i en vanntank.

Hvorfor endre et ventilasjonssystem som er basert på reléstyring og som fungerer slik det skal? For at et system eller en løsning skal fungere optimalt krever den oppdateringer, og ikke minst forbedringer der det er mulig. Lokal styring av ventilasjonen med en PLS og et HMI-panel er i dette tilfelle en nødvendig forbedring av den eksisterende løsningen, da den er mer moderne og er mindre krevende å sette opp/betjene. Dette gir mer fleksibilitet.

Det som hovedsakelig er blitt gjort i denne oppgaven er programmering av en PLS til å være lokal PLS slik at det er denne som styrer ventilasjonen i en tunnel. Et HMI-panel er også blitt programmert til å være betjeningen av denne PLSen, samtidig som flerlinjeskjema av løsningen er blitt tegnet.

1.2 Oppgavebeskrivelse

Oppgaven «Lokal ventilasjonsstyring» tar utgangspunkt i dagens løsning for ventilasjon i tunneler. Denne tradisjonelle, lokale styringen av ventilatorer i vegtunneler er basert på vendere, indikasjonslys og reléstyring. Styringen i dag består av tavler med reléstyring for hver enkelt vifte,

med betjening for å kunne kjøre viftene lokalt om hoved PLS er ute av drift. I selve styringen er det flere tidsrelé og forriglinger.

Figur 1: Lokal styring via tavle, bilde fra oppgavetekst

Oppdragsgiver ønsker at dette skal oppgraderes til en moderne styring, samtidig som driftssikkerheten opprettholdes. Betjeningen kan/skal erstattes med et HMI-panel, styringen kan/skal erstattes med en lokal PLS. Det skal også utføres beregninger for synliggjøring av at den nye styringen er sikker. Her ønsket oppdragsgiver at det skulle sees på MTBF (Mean Time Between Failure) eller SIL[1]. I oppgaveteksten ønsker oppdragsgiver også å se på produksjonstiden til tavlene som er i bruk i dag, samt se på komponentpriser og produksjonstid for den nye løsningen.

Oppgaveteksten består av mange delelementer en kunne velge å ha fokus på. I denne oppgaven er det valgt å fokusere på PLS som lokal styring med tilhørende HMI-panel for betjening. Det er også sett på beregninger ved bruk av metoden MTBF, samt priser på de ulike komponentene. I denne oppgaven er det valgt å se bort fra produksjonstid på tavlene, komponentene til denne og å lære seg å lage tavler. Dette er blitt gjort for å avgrense oppgaven til det tidsperspektivet som er gitt, samt legge fokus på lokal styring fra PLS.

[1] Hentet fra oppgavetekst

2 Teori

Dette kapittelet inneholder relevant teori for resten av oppgaven.

2.1 Kort om tunneler i forhold til oppgaven

I dette delkapittelet presenteres det kort og generelt hvordan ventilasjon i en tunnel driftes og er oppbygd i dag. Dette gjøres for å gi et lite innblikk i hvordan systemet fungerer, og det gjør det lettere å presentere løsningen senere i rapporten.

Viftesystemene i dagens tunneler er basert på vendere, indikasjonslys og reléstyring. Det er også en PLS inne i styringen. I denne oppgaven kalles denne PLSen hoved PLS, mens i systemet i tunnelene kalles den «Auto». Auto PLSen overvåker trafikk og gass i tunnelen. Det er sensorer montert inne i tunnelen som sender informasjon om CO2 til Auto. Auto ser også på ÅDT, som står for årsdøgntrafikk. ÅDT er et gjennomsnittstall for daglig trafikkmengde, det vil si hvor mange kjøretøy som passerer et gitt punkt (gjelder for begge kjøreretninger) gjennom ett år, delt på antall dager i året.

Viftene i tunnelsystemet kan kjøres begge veier, hvilken vei de skal drive luften ut av tunnelen bestemmes av sensorer som måler hvor det er mest CO2 og hvor nærmeste tunnelåpning er. Store/lange tunneler har sjakter for lufttilførsel.

2.2 Relé

Et relé er en er en type strømbryter. Et relé er elektrisk drevet og er en elektromekanisk bryter som består av en elektromagnet, et armatur, en fjær og et sett med elektriske kontakter. Virkemåten til et relé er at den mottar en liten elektrisk strøm, og dermed slår av eller på en større strøm ved enten å bryte eller slutte den elektriske kretsen. Reléer brukes når det må være elektrisk isolasjon mellom kontrollkretser, eller når flere kretser styres av et enkelt signal. Det er også noe som heter tidsrelé, som gjør at en kan forsinke et start- eller stoppsignal [4].

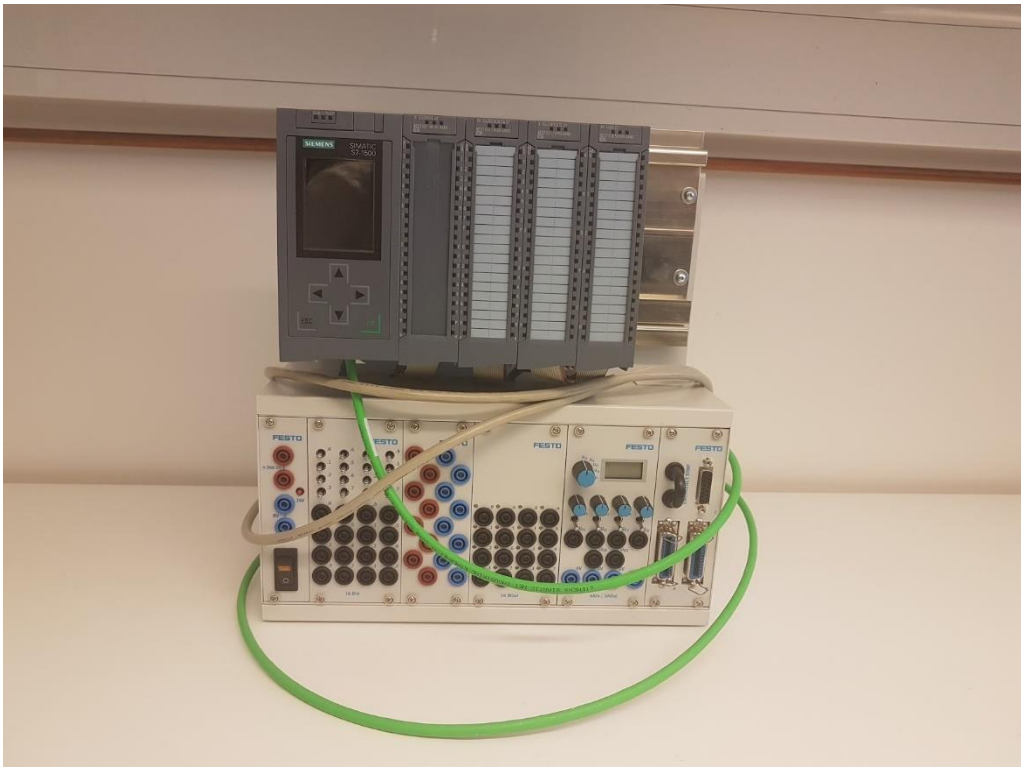
2.3 PLS

PLS står for programmerbar logisk styring, og dette er en datamaskin som brukes til styring, og regulering av prosesser, hovedsakelig i produksjonsindustrien og petroleumsindustrien. PLS fungerer slik at den mottar informasjon fra sensorer som er koblet til den, via inngangssignaler. Den behandler informasjonen som kommer til inngangene for så å aktivere utganger basert på forprogrammerte

parametere. PLS blir først programmert i programmet TIA Portal V15, deretter simuleres det i en PLS-simulator i det samme programmet. Når dette fungerer som det skal, kan det kobles til en ekte PLS, og testes [5] [6].

Det finnes flere språk en PLS kan bli programmert med, men de tre språkene som gruppen har kjennskap til fra før er FBD, SCL og Graph. Det første er Function Block Diagram og går ut på at input og output-variabler blir koblet gjennom blokker, som for eksempel «&-blokker» og «>=1-blokker» (står for «eller»). SCL står for Structured Control Language, og det er et tekstbasert programmeringsspråk.

Graph er språket som skal brukes til å programmere sekvensene til PLSen, og det har stor likhet med tilstandsdiagram. Det er blokker som blir utført etter hverandre, ut fra ulike inputs. For eksempel et program som skal motta et signal, for deretter å starte en enhet. Videre mottar programmet et annet signal for å stoppe, og dermed slår det av enheten.



Figur 2: Bilde av PLSen som er brukt i oppgaven

2.4 Modbus

Modbus er en åpen kommunikasjonsstandard som støttes av et stort utvalg av produkter og forhandlere i dagens marked. Modbus handler om kommunikasjon mellom en master og en eller

flere slaver. Master sender en forespørsel og slaven sender et svar tilbake. Disse meldingene mellom master og slave blir sendt over en RS232/RS485 seriell kommunikasjonskobling (EIA-standard).

Noen av de tekniske egenskapene ved modbus er at det kan være opptil 247 enheter i et nettverk, det er opptil en km nettverksrekkevidde, kringkasting av meldinger støttes og det er standard kabling. Modbus seriell dataoverføring kan brukes i flere applikasjoner, i mange ulike typer PLSer og de fleste Telemecanique-enheter [7] [8].

Seriell modbus-kommunikasjon er den tradisjonelle kommunikasjon, men i dette tilfellet bruker vi TCP/IP over modbus da dette gir flere fordeler. Fordeler som for eksempel at den er enkel å sette opp, har lav utviklingskostnad og det behøves lite ekstra hardware.

2.5 Mean Time Between Failure (MTBF)

MTBF[2] er et mål på hvor pålitelig et maskinvareprodukt eller -komponent er. For de fleste komponenter er målingen vanligvis i tusen eller til og med titusenvis av timer mellom feil. Dette er en beregningsmetode vi skal bruke for å finne ut hvor pålitelig løsningen vår er, og hvor pålitelig systemet og viftene er i drift. Formler for å beregne MTBF:

$$\bar{\lambda} = Feilrate = \frac{Total\ number\ of\ failures\ within\ an\ item\ population}{Total\ time\ expended\ by\ population}$$

$$MTBF = Mean\ Time\ Between\ Failures = \frac{Total\ time\ the\ monitored\ systems\ have\ been\ available}{Number\ of\ failures}$$

$$MDT = Mean\ Down\ Time = \frac{Total\ time\ the\ monitored\ systems\ have\ been\ unavailable}{Number\ of\ failures}$$

$$\bar{\lambda} = \frac{1}{MTBF}$$

$$Total\ time\ monitored = T$$

$$Total\ down\ time = TDT = Number\ of\ failures * MDT$$

$$Total\ up\ time = TUT = T - TDT$$

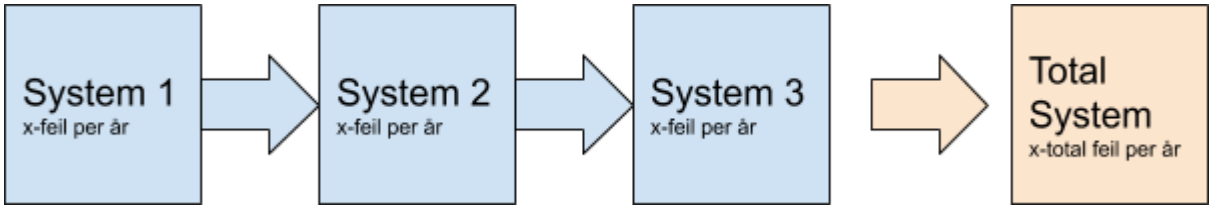
$$Mean\ Time\ between\ Failures = MTBF = \frac{TUT}{Number\ of\ failures}$$

Figur 3: Formler for utregning av MTBF 1

[2] Formlene er tatt fra forelesning i faget ELE109, og PPT med formlene er utformet av Mathias Christian Mathiesen

For å regne ut MTBF i større systemer, som gjerne består av flere undersystemer, må det tas hensyn til både serielle og parallelle feil. For å regne ut serielle og parallelle feil brukes det ulike formler, disse formlene kan også kombineres for å få en nøyaktig MTBF-beregning.

Systemer i serie: «Feil i en komponent gjør at hele systemet stanser/feiler». Systemer i serie ser slik ut som modellen under.



Figur 4: System i serie

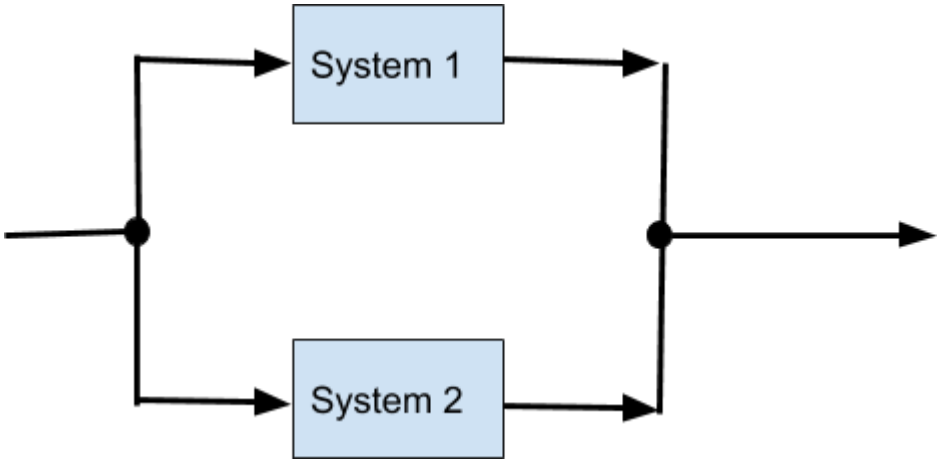
Bildet under viser formelen for systemer i serie.

$$\lambda_{tot} = \lambda_1 + \lambda_2 + \cdots + \lambda_n$$

$$\frac{1}{MTBF_{TOT}} = \frac{1}{MTBF_1} + \frac{1}{MTBF_2} + \cdots + \frac{1}{MTBF_n}$$

Figur 5: Formler for beregning av MTBF i serie

Systemer i parallell: «Feil i en komponent stopper ikke systemet». Systemer i parallell ser slik ut som i modellen under.



Figur 6: System i parallell

Bildet under viser formelen for systemer i parallell.

$$\lambda_{tot} = \lambda_1 * (\lambda_2 * MDT_1) + \lambda_2 * (\lambda_1 * MDT_2)$$

$$= \frac{1}{MTBF_1} * \frac{1}{MTBF_2} * MDT_1 + \frac{1}{MTBF_2} * \frac{1}{MTBF_1} * MDT_2$$

$$= \frac{MDT_1 + MDT_2}{MTBF_1 * MTBF_2}$$

$$MTBF_{tot} = \frac{1}{\lambda_{tot}} = \frac{MTBF_1 * MTBF_2}{MDT_1 + MDT_2}$$

Figur 7: Formler for beregning av MTBF i parallell

Parallelle systemer vil feile hvis for eksempel system 1 feiler samtidig som system 2 er nede og motsatt.

MDT i parallelle systemer.

$$MDT_{tot} = \frac{(Sum\ downtime\ from\ system\ 1) + (Sum\ downtime\ from\ system\ 2)}{Total\ number\ of\ failures}$$

$$Per\ unit\ time: MDT_{tot} = \frac{MDT_1 * \lambda_1 + MDT_2 * \lambda_2}{\lambda_1 + \lambda_2}$$

Figur 8: MDT i parallelle systemer

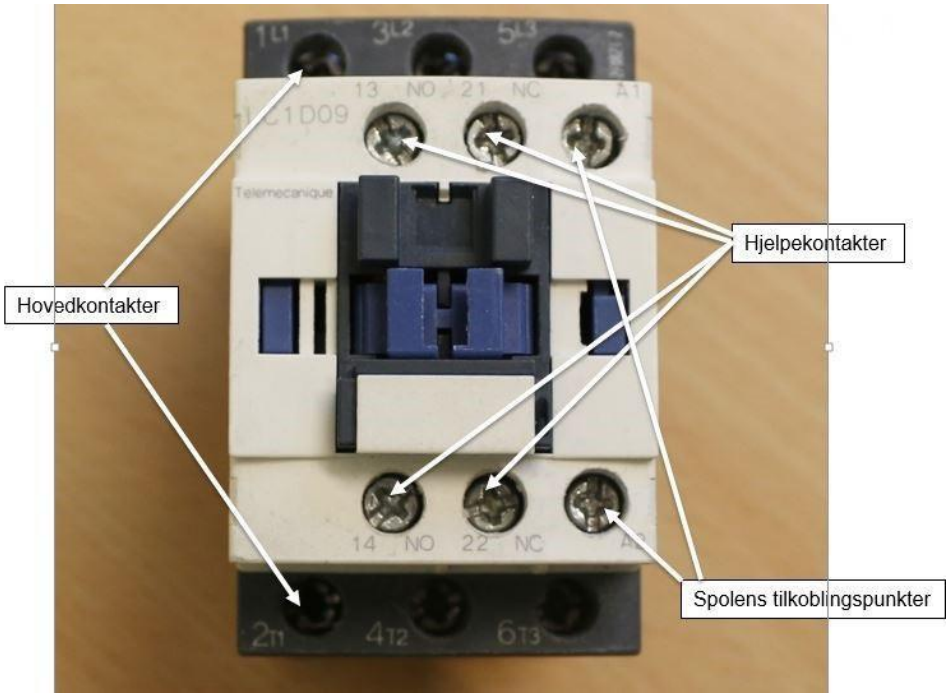
2.6 Human Machine Interface (HMI)

HMI er et brukergrensesnitt eller panel som gjør det lettere for mennesker å bruke og å styre en maskin, system eller enhet. I dette tilfellet skal HMI-panelet være koblet til en PLS for å styre denne. HMI-panelet kan programmeres og designes via TIA portal.

PLSen som HMI-panelet er koblet til, skal igjen være koblet til en hoved PLS som styrer all ventilasjon i tunnel. HMI-panelet skal være programmert slik at den overkjører alle andres styring. En HMI har eget minne, men kan også ha et eksternt minnekort. I oppgaven er HMIen koblet til PLS med TP-kabel (PROFINET), men kan også kobles via profibus [9].

Figur 9: HMI-panel brukt i løsningen

2.7 Kontaktor



Figur 10: Kontaktor, bilde fra kilde [10]

En kontaktor ligner på et relé, men den er bedre egnet til å slutte større elektriske strømmer enn et relé. En kontaktor beveger seg raskere og med større kraft, noe som gjør at det er mindre sannsynlig

at det for eksempel blir brann ved store strømmer. Kontaktoren består av spole, jernkjerne (anker), hovedkontakter og hjelpekontakter. En spole er viklinger omkring en jernkjerne. Når det drar strøm gjennom spolen, blir jernkjernen magnetisert og drar den bevegelige delen på kontaktoren til seg, og den virker som en bryter.

Hovedkontaktene på kontaktoren bryter trefasestrømmen på motorstrømmen. Hovedkontaktene er merket med 1, 3 og 5 og det er her tilførselen skal kobles til. 2, 4 og 6 er et termisk relévern tilpasset kontaktoren eller motorkablene som skal kobles til.
Dette betyr at:

- Fase 1 går gjennom punkt 1 og 2 på kontaktoren.
- Fase 2 går gjennom punkt 3 og 4 på kontaktoren.
- Fase 3 går gjennom punkt 5 og 6 på kontaktoren.

Hjelpekontaktene kalles også for styrekontakter, disse brukes til styring, alarm og signaler. Disse er ofte merket med tosifrede tall, som for eksempel 13-14, 21-22 osv. Det første tallet angir kontaktenes nummer i rekken av kontakter. Det andre sifferet angir om det er en NC (Normally Closed) eller NO (Normally Open). Hvis andre siffer er 1 og 2 er kontakten NC, og hvis det andre sifferet 3 og 4 er den NO. Normally Open vil si at kursen er brutt når kontaktoren er uten spenning, og omvendt når kontaktoren er koblet til spenning. Normally Closed vil si at kursen brytes når kontaktoren er tilkoblet spenning. Normally Closed brukes ofte som en nødstopp-knapp [10] [11].

2.8 Flerlinjeskjema

Et flerlinjeskjema viser hvordan et anlegg er koblet opp, og tegnes med to eller flere linjer. Til forskjell fra et enlinjeskjema, som viser hvilke komponenter som er med og hvor kabelen går, har et flerlinjeskjema flere detaljer som viser komponentene som er med, lederne i kabelen som er tilkoblet og hvordan lederne skal kobles.

3 Oppsett av oppgaven

3.1 Hovedidé for løsningsforslag

Hovedpunktene for løsningen vår:

- Bytte ut reléstyring med en lokal PLS som kommuniserer med hoved PLS.

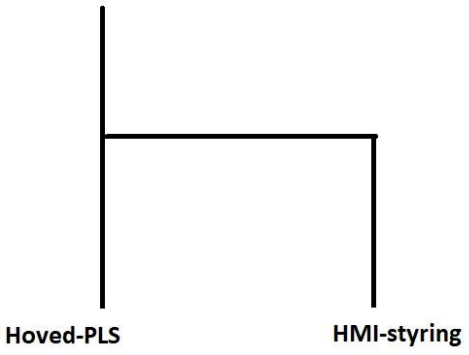
- Lage et fungerende program til ventileringen.
- Programmere HMI-panelet med enkelt design.
- Kommunikasjon mellom lokal PLS og hoved PLS skal gå via TCP/IP over modbus.

Hovedidéen for løsningen er at det skal konfigureres en lokal PLS med et eget program for å styre ventilasjonen i tunneler, denne PLSen skal også kunne kommunisere med hoved PLS i anlegget. Kommunikasjonen mellom disse PLSene skal gå via TCP/IP over modbus. Hoved PLSen i anlegget har et eget program som det skal sees bort fra, og gjøre lokal PLS klar for å motta beskjeder. Hoved PLS kan bestemme hvor mange vifter som skal kjøres ut fra sensorer og kommandoer fra vegtrafikksentralen, samtidig som lokal PLS har sitt eget program som kan kjøres.

For å løse dette i programmet lager vi alternative greiner, som vil kunne byttes mellom, alt etter hvor kommandoene for styring kommer fra. Kommer det signal fra hoved PLS vil lokal PLS kjøre det som blir mottatt, kommer det signal fra HMI-panelet, så vil lokal PLS kjøre dette i stedet.

Programmet på HMI-panelet skal være enkelt å navigere i. HMI-panelet skal vise status på viftene, symbolisert med et lys som lyser svart eller grønt, alt etter om viftene er av eller på. Programmet på HMI-panelet skal også ha mulighet til å slå hver enkelt vifte av og på.

Modell for program (tidlig fase).



Figur 11: Modell for program i tidlig fase

I startfasen av prosjektet ble det diskutert hvordan programmet skulle bygges opp, og der kom vi frem til at den beste løsningen for denne oppgaven er å lage to greiner som kjøres alternativt alt etter hvilke signaler som mottas. I startfasen ble det også diskutert om programmet skulle ta hensyn til dagmodus og nattmodus, noe vi gikk vekk fra i løpet av arbeidet.

3.2 Kravspesifikasjon

Kravene vi og oppdragsgiver har satt til oppgaven er at styringen som utvikles skal være en sikker logisk styring. Med dette mener vi at den lokale styringen skal foregå via en PLS som har et eget program som kjøres, eller mottar signaler fra hoved PLS. HMI-panel skal betjene og styre programmet som kjøres på den lokale PLSen. HMI-panelet skal også ha et enkelt design, samtidig som det er lett å navigere i. Det ble også beregnet MTBF på denne nye løsningen, og tegnet et flerlinjeskjema av løsningen.

#	Krav
1	Sikker logisk styring (PLS)
2	Enkel betjening via HMI-panel
3	Beregninger via MTBF
4	Flerlinjeskjema

Figur 12: Krav til løsningen

3.3 Tidsplan

					Uke	w2	w3	w4	w5	w6	w7	w8	w9	w10	w11	w12	w13	w14	w15	w16	w17	w18	w19	w20	w21	w22	w23	w24	w25	
					Man	7/1	14/1	21/1	28/1	4/2	11/2	18/2	25/2	4/3	11/3	18/3	25/3	1/4	8/4	15/4	22/4	29/4	6/5	13/5	20/5	27/5	3/6	10/6	17/6	
					Fre	11/1	18/1	25/1	1/2	8/2	15/2	22/2	1/3	8/3	15/3	22/3	29/3	5/4	12/4	19/4	26/4	3/5	10/5	17/5	24/5	31/5	7/6	14/6	21/6	
#	Aktivitet	Start Dato	Slutt Dato	Fram drift	Ansvarlig								SU?							Påske									
1	Innhenting av oppgaver	1/9	1/11	100 %	alle																								
2	Presentasjon av oppgaver for stud	13/11		100 %	Institutt																								
3	Innlevering av oppgave ønsker	20/11		100 %	Oss																								
4	Endelig tildeling av oppgaver	27/11		100 %	Institutt																								
5	Forstudie arbeid	7/1	31/1	100 %	Oss	~100/200/300/400 time																								
6	Metode undervisning	8/1		100 %	institutt		2t	2t	2t	2t				2t			2t													
7	Forstudie innlevering	12/2		90 %	Oss																								absolutt frist
	Første møte med Inge Vivås	21/1	21/1	100 %	Oss																								
8	Første møte med Goodtech	29/1	29/1	100 %	Oss																								
9	Bachelor oppgave arbeid			0 %	Oss						~400/800/1200/1600 timer																		
10	Midtveis presentasjon	1/4	5/4	0 %	Oss																								avtales med veileder
11	Bachelor oppgave seminar/ Siste UV dag	10/5		0 %	Oss																								10 min/gruppe
12	Ordinær eksamen	13/5	11/6	0 %	Oss																								
13	Konte eksamen	5/6	11/6	0 %	Oss																								
14	Bachelor oppgave innlevering	31/5		0 %	Oss																								absolutt frist
15	Bachelor oppgave presentasjon	5/6	12/6	0 %	Oss																								avtales med veileder
16	EXPO / Avslutningsfest	13/6		0 %	Oss																								alle må stille

3.4 Planlegging

I arbeidet med oppgaven har gruppen fått mulighet til å sitte på Goodtech sine lokaler på Damsgård. Dette har vi benyttet oss av, ved å sitte hos dem i gjennomsnitt to dager i uken i løpet av semesteret. Her har vi hatt nær kontakt med ekstern veileder og fått mulighet til å få hjelp av andre ansatte i

bedriften. Vi har også brukt en del tid på skolen for å benytte oss av skolens PLSer og HMI-panel for programmering og testing av løsningen, samtidig som intern veileder også er tilgjengelig.

3.5 Analyse av problemet

Etter stilte krav til oppgaven og begrensning av arbeidsomfanget, vil det være naturlig å gå for kun én løsning.

3.5.1 Utforming av mulige løsninger

Oppgaveteksten sier at for å erstatte reléstyring må denne byttes ut med en sikker logisk styring, som vil si en PLS. Denne PLSen vil stå for den lokale styringen, og vil samtidig motta signaler fra hoved PLS i anlegget. Oppdragsgiver ønsker at denne kommunikasjonen skal være modbus TCP/IP, de ønsker også å ha et HMI-panel for betjening.

Kravene fra oppdragsgiver gjør at det kun er en type løsning på oppgaven, men det er fritt valg hvilket språk som brukes i programmeringen av PLSen. I denne oppgaven er det valgt språket GRAPH, siden det er dette språket gruppa har mest kjennskap til. Det er også blitt brukt function blocks for å lese og skrive informasjonen til modbusregisteret.

Det har også blitt laget en liten modell av løsningen som skal vises frem på EXPO på skolen (se kapittel 6).

3.5.2 Vurderinger i forhold til verktøy og HW-/SW-komponenter

Hardware som er blitt brukt i oppgaven er PLS og et HMI-panel koblet til denne. PLSen vi har brukt er modellen S7-1500 fra Siemens, og HMI-panelet er også fra Siemens, modell TP700 Comfort. I modellen til EXPO bruker vi komponentene som nevnt over, men for å slå av/på viften bruker vi kontaktorer.

Softwaren som har blitt brukt heter TIA portal, og dette brukes for å programmere programmet til PLSen og HMI-panelet. En annen software vi har brukt er PCSCHEMATIC. PCSCHEMATIC er et EL-dokumentasjonsprogram, et tegneprogram for elektriske installasjoner [12]. I oppgaven er dette programmet brukt for å tegne flerlinjeskjema av løsningen. Dette var noe oppdragsgiver ønsket at skulle følge med den nye løsningen. Programmet har mange komponenter i sitt bibliotek som gjorde det enkelt å finne frem komponentene som ble brukt i oppgaven.

4 Program

I dette kapittelet presenteres programmet som blir utformet. Her vises variabler og hvordan programmet ble bygget. Som nevnt så brukes programmeringsspråket GRAPH i denne løsningen. Enkelte steder i programmet er det lagt inn kommentarer, slik at det er lett å huske hva som har blitt gjort, hva ulike funksjoner gjør og små forklaringer.

4.1 Variabler

For å tilordne innganger og utganger forskjellige egenskaper, må vi ha navn, datatype og en adresse på dem. Dette kalles variabler/tags. Navnene på variablene bidrar til et oversiktlig program, der det er lett å se hva inngangene og utgangene skal gjøre. I programmet TIA Portal er det også en seksjon for å skrive små kommentarer til variablene. Bildet nedenfor er et eksempel på en tabell med ulike variabler.

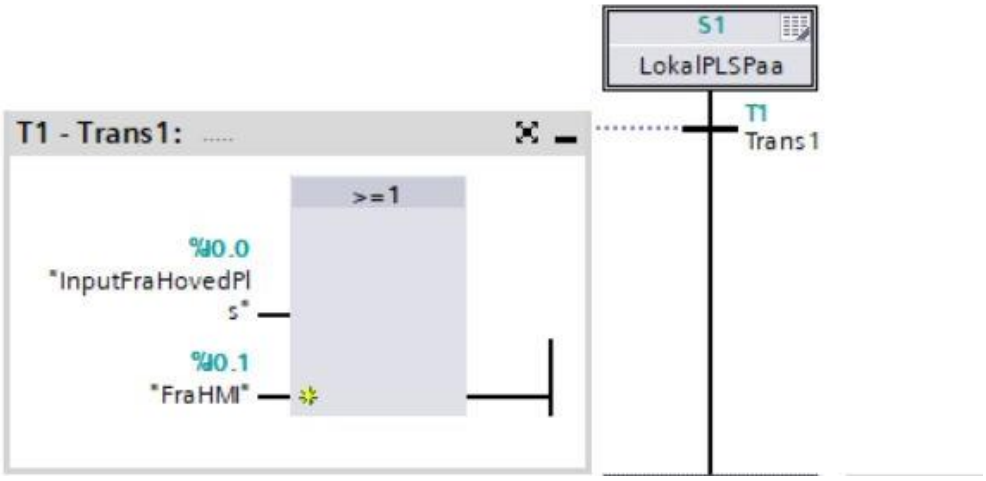
	Name	Data type	Address	Retain	Acces...	Writa...	Visibl...	Supervis...	Comment
1	InputFraHovedPls	Bool	%I0.0	☐	☑	☑	☑		Meldinger fra HovedPLS-en
2	FraHMI	Bool	%I0.1	☐	☑	☑	☑		Meldinger fra HMI-skjerm
3	Vifte1	Bool	%Q0.2	☐	☑	☑	☑		Slå av/på vifte
4	Vifte2	Bool	%Q0.3	☐	☑	☑	☑		Slå av/på vifte
5	Vifte3	Bool	%Q0.4	☐	☑	☑	☑		Slå av/på vifte
6	Vifte4	Bool	%Q0.5	☐	☑	☑	☑		Slå av/på vifte
7	Vender/kontaktorHMI	Bool	%I0.6	☐	☑	☑	☑		Aktiv høy
8	Vender/kontaktorHPLS	Bool	%I0.7	☐	☑	☑	☑		Aktiv høy
9	Vender/kontaktorNATTPLS	Bool	%I1.0	☐	☑	☑	☑		Aktiv høy
10	Vender/kontaktorNATTHMI	Bool	%I1.1	☐	☑	☑	☑		Aktiv høy
11	<Add new>			☐	☑	☑	☑		

Figur 13: Eksempelbilde variabler

I bildet over er det lagt til flere variabler, noen er innganger og noen er utganger. Datatypen på variablene i bildet er «Bool», som betyr at de kun sender «0» og «1», kan også kalles aktiv lav og aktiv høy. Verdiene som blir sendt/mottatt via inngangene og utgangene blir lagret i variabelen. Bildet over er fra startfasen av programmet, og brukes her kun som et eksempel for å beskrive hvordan variablene kan være. Variablene kan ha ulike funksjoner, i denne oppgaven er det mest fornuftig å bruke bool, da det kun skal sendes om inngangene og utgangene er av eller på.

4.2 Hovedprogram i tidlig fase

Dette delkapittelet viser hvordan programmet så ut i en tidlig fase av prosjektet, dette var en idé for hvordan programmet kunne ha blitt bygd opp.

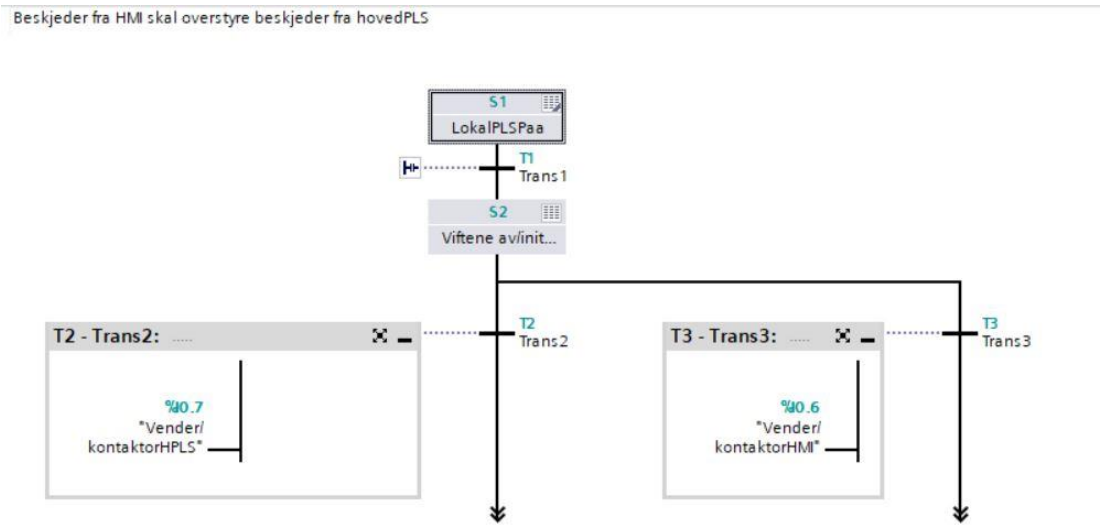


Figur 14: Program i tidlig fase 1

Figur 15: Program i tidlig fase 2

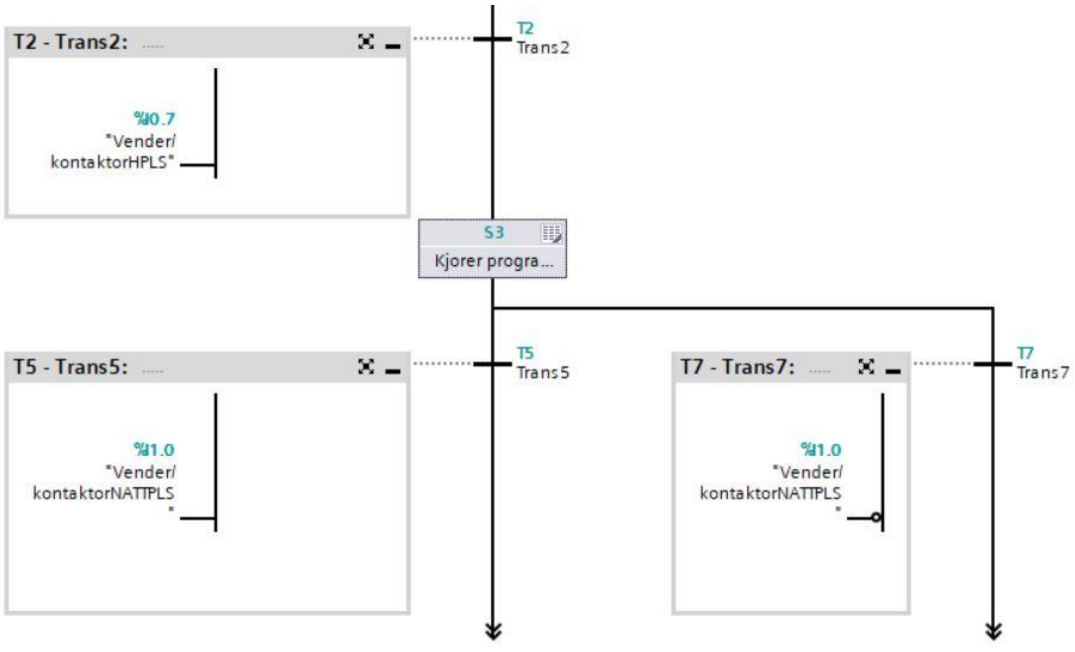
Step 1 blir kalt LokalPLSPaa og bruker dette som et initialiserings-step for å starte programmet, der kan viftene få på-signaler fra hoved PLS eller HMI-panelet.

Step 2 er en forlengelse av step 1, kalles også sikkerhets-step, der viftene er avslått frem til det kommer noen annen beskjed. Dette er lurt å ha for å vite med sikkerhet at viftene er avslått.



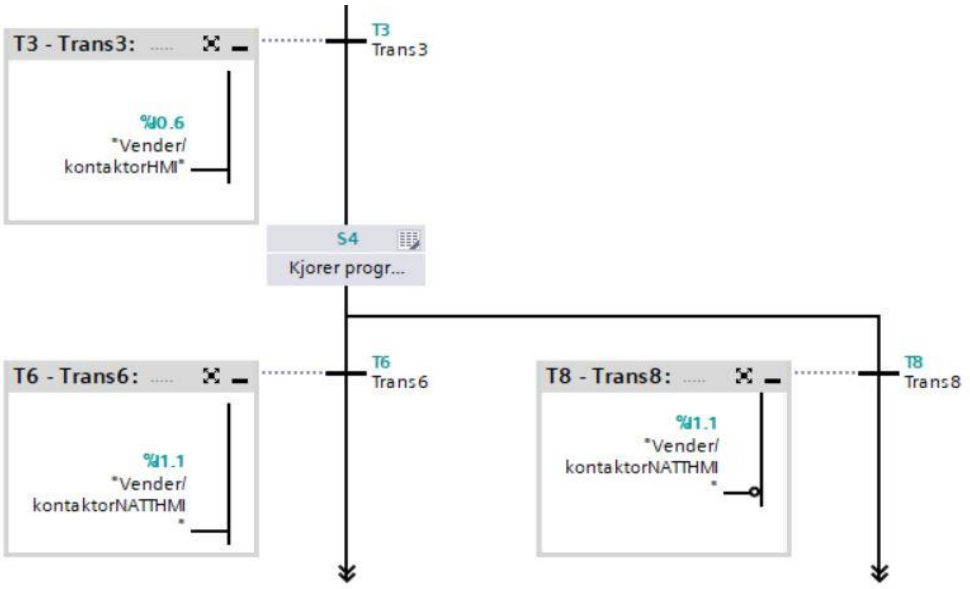
Figur 16: Program i tidlig fase 3

Her er det blitt lagt inn en alternativ grein, der veivalget blir mellom beskjeder fra hoved PLS eller HMI-panelet. Dette gjøres fordi oppgaven går ut på å lage styring for en lokal PLS, med mulighet for å motta beskjeder fra både hoved PLS eller HMI-panel.



Figur 17: Program i tidlig fase 4

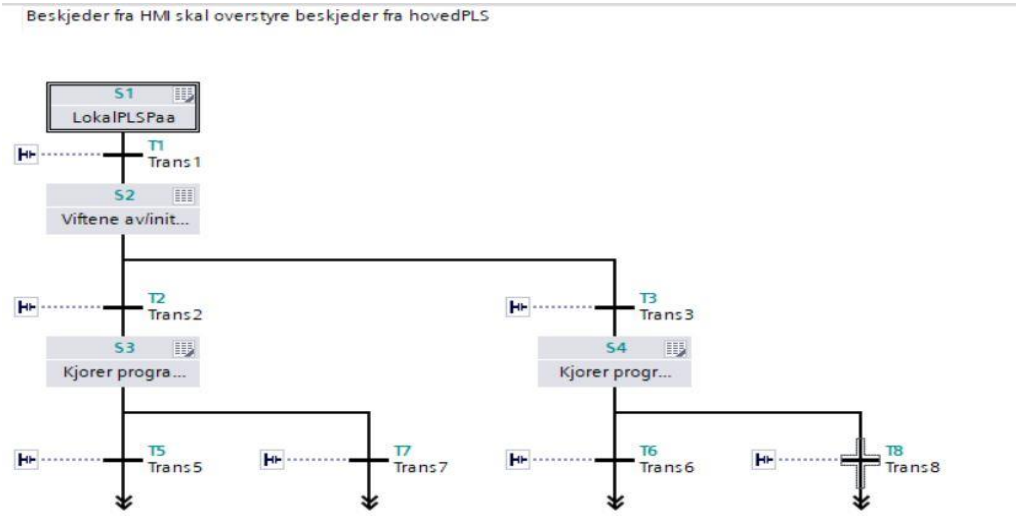
Bildet over viser dag- og nattmodus for hoved PLS, og bildet under viser dag- og nattmodus for HMI-panelet.



Figur 18: Program i tidlig fase 5

I denne tidlige versjonen av programmet ble det tatt med dagmodus og nattmodus for å kunne skille mellom disse ulike modusene, da det normalt er mer trafikk på dagtid enn på natten. Det er ofte mer stillegående køer på dagtid på grunn av rushtrafikk.

Bildet under viser et oversiktsbilde av det tidlige programmet.



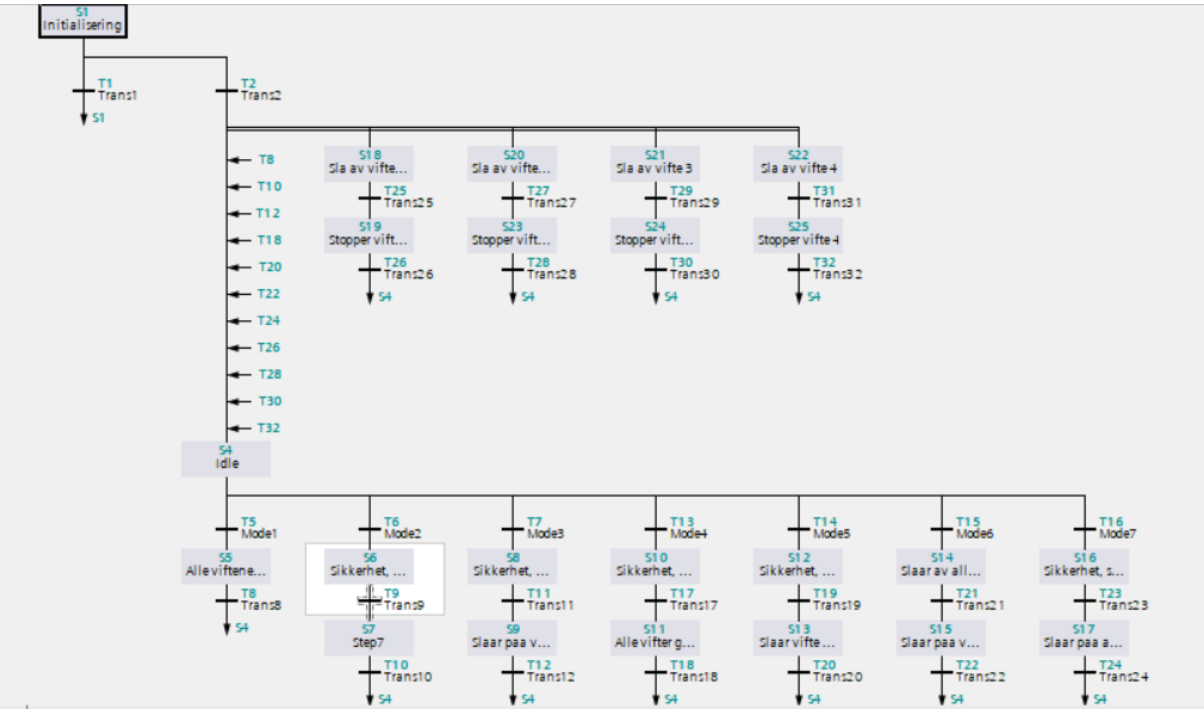
Figur 19: Oversiktsbilde av program i tidlig fase

Etter en diskusjon med oppdragsgiver og veiledere har vi sett bort fra denne versjonen av programmet og omstrukturert det for å få det kompatibelt med kommunikasjon via TCP/IP over modbus. Det tas heller ikke hensyn til dag- og nattmodus, da dette er noe som hoved PLSen ordner

selv, og ikke noe oppgaven skal ta hensyn til. Den nye versjonen av programmet skal ta utgangspunkt i denne oppbyggingen av programmet ved bruk av språket GRAPH og bruken av alternative greiner og simultane greiner, men se bort fra dag- og nattmodus. Det tas utgangspunkt i at hoved PLS har programmet vårt.

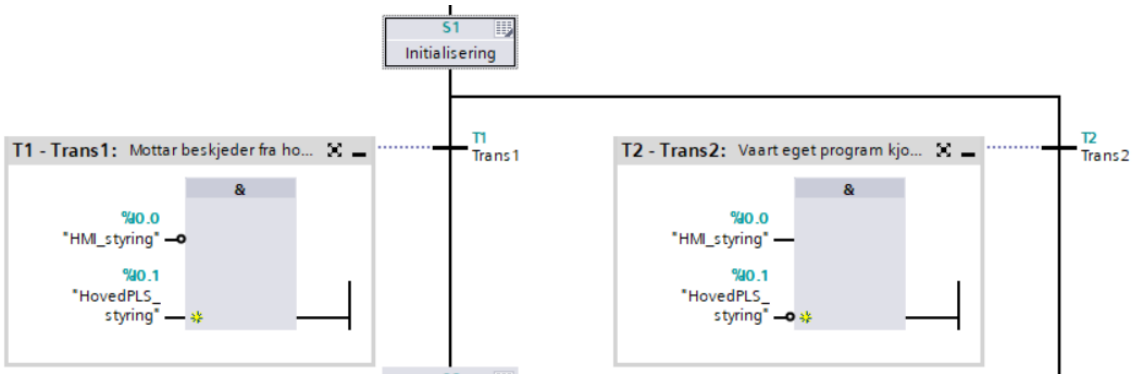
4.3 Endelig hovedprogram

Nedenfor kan man se et oversiktsbilde av hovedprogrammet.



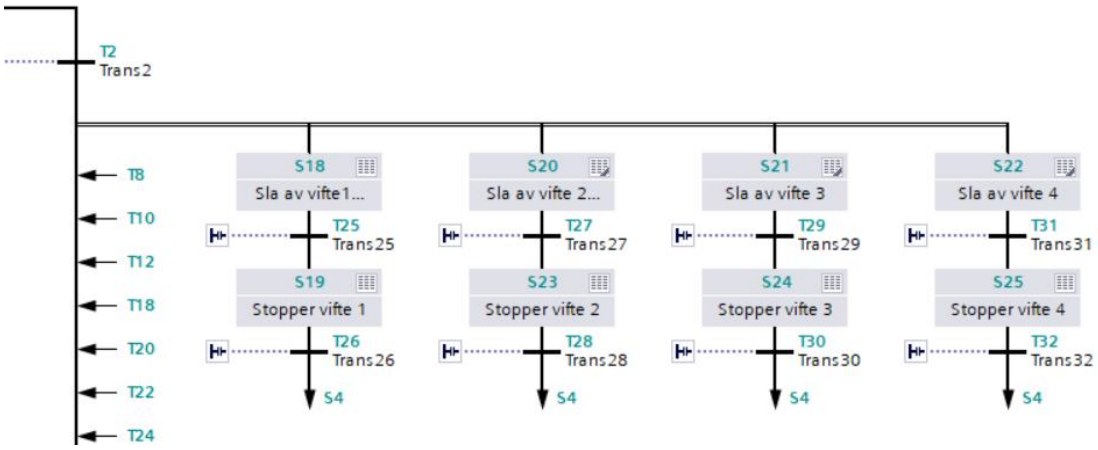
Figur 20: Oversiktsbilde av endelig program

I bildet under begynner programmet med et step som heter «initialisering», så deles programmet i to. Dette fordi lokal PLS skal kunne bli styrt både av HMI-panelet som er som nevnt, prioritet nummer 1 og av hoved PLS som er prioritet nummer 2. Helt til venstre vises transition 1 som er den delen som skal bli styrt av hoved PLSen, derfor er variabelen HMI_styring invertert. Dette skal vi i utgangspunktet ikke gjøre noe med og derfor går vi ut i fra at hoved PLS er programmert på samme måte, bare at den sender informasjon til lokal PLS. Hoved PLS mottar status på viftene. Helt til høyre vises transition 2 som er begynnelsen av den delen av programmet som skal bli styrt av HMI-panelet og bli programmert av oss.



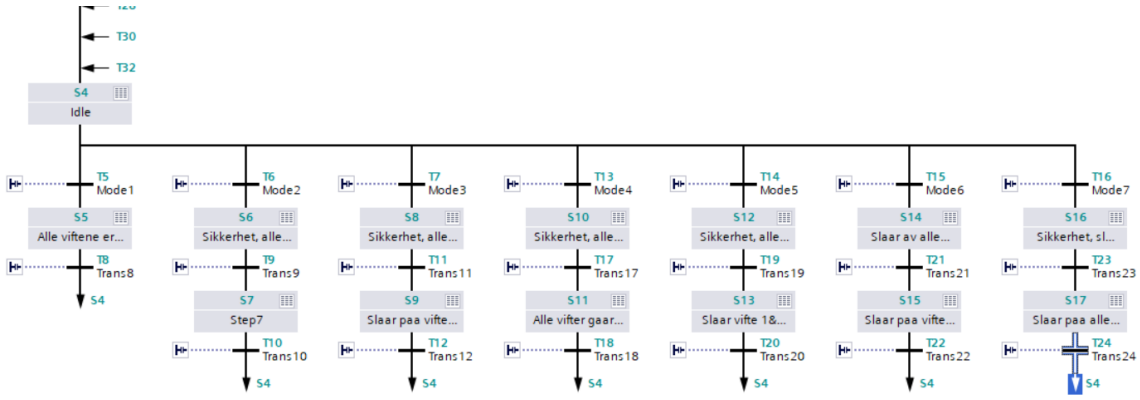
Figur 21: Endelig program bilde 1

Bildet under viser fire ulike greiner, som tilsvarer de fire viftene i tunnelen. I den første greina kan man slå av vifte 1, i grein 2 kan man slå av vifte 2, i grein 3 kan man slå av vifte 3 og i den siste greina kan man slå av vifte 4. Hver vifte skal kunne slåes av uavhengig av hvilken modus som er på. Dette tilsvarer vedlikeholdsbildet i HMI-panelet.



Figur 22: Endelig program bilde 2

Bildet under viser de syv ulike modes/alternativer som er blitt konfigurert på PLS og HMI-panelet.

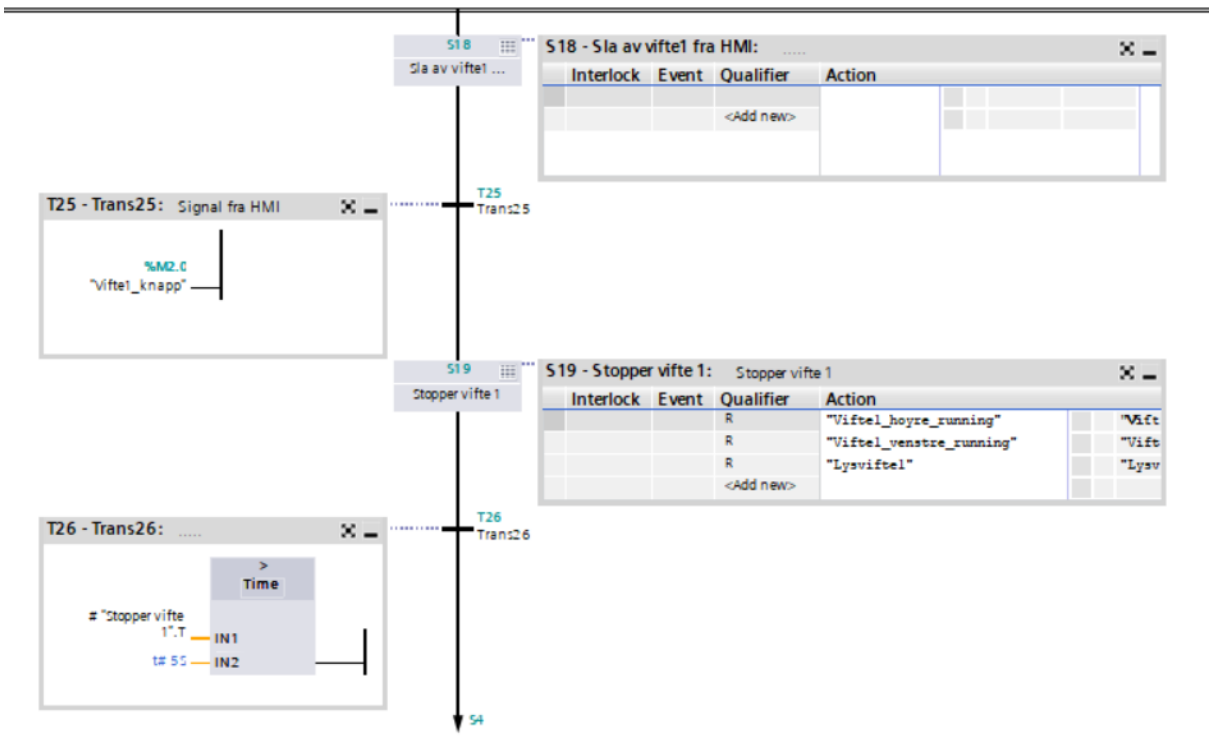


Figur 23: Endelig program bilde 3

- Alternativ 1: Alle viftene er slått av.
- Alternativ 2: Vifte 1 og 2 er slått på mot venstre.
- Alternativ 3: Vifte 3 og 4 er slått på mot venstre.
- Alternativ 4: Alle viftene er slått på mot venstre.
- Alternativ 5: Vifte 1 og 2 er slått på mot høyre.
- Alternativ 6: Vifte 3 og 4 er slått på mot høyre.
- Alternativ 7: Alle viftene er slått på mot høyre.

Dette tilsvarer lokalstyringsbildet i HMI-panelet. Senere i dette kapittelet skal vi ta for oss hvordan de ulike alternativene er programmert i TIA-PORTAL V15-programmet.

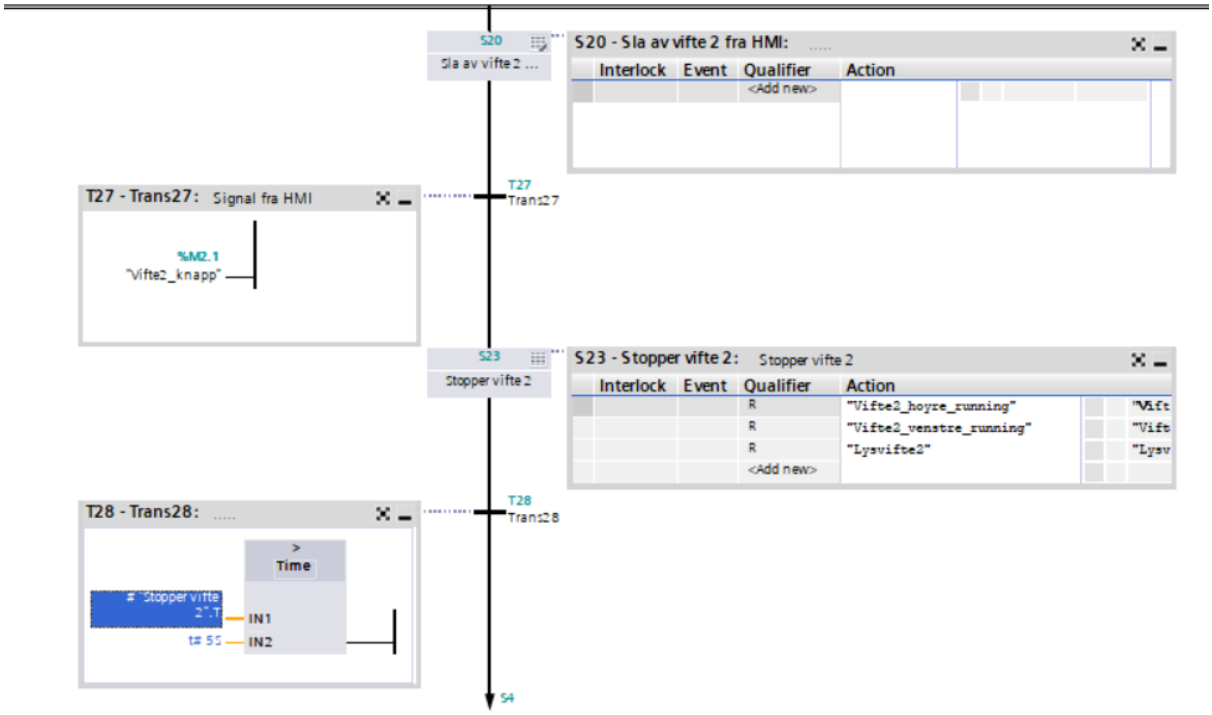
Starter med forklaring på PLS-programmet som handler om vedlikehold av viftene. Det første man ser er en tom step-boks, så kommer det en transition med signal fra HMI-panelet, dette signalet blir sendt når knappen «vifte1_knapp» blir trykket på. Når dette skjer kommer det en step-boks som viser at begge retninger (mot venstre og mot høyre) blir slått av i vifte 1. Hver vifte har to variabler, en for retning mot venstre og en mot høyre, disse blir slått av ved å sette de til Reset, altså til 0.



Figur 24: Endelig program bilde 4

Man kan si at en transition-boks er en slags «input»-boks som i dette tilfellet får signal fra HMI-panelet, mens step-boksene er «output»-bokser. Til slutt er det en timer på 5 sekunder og så hopper programmet tilbake til S4, der lokal PLS blir styrt av HMI-panelet. Dette fordi den skal bli kjørt i en «loop» slik at de ulike alternativene kan benyttes til forskjellige tider.

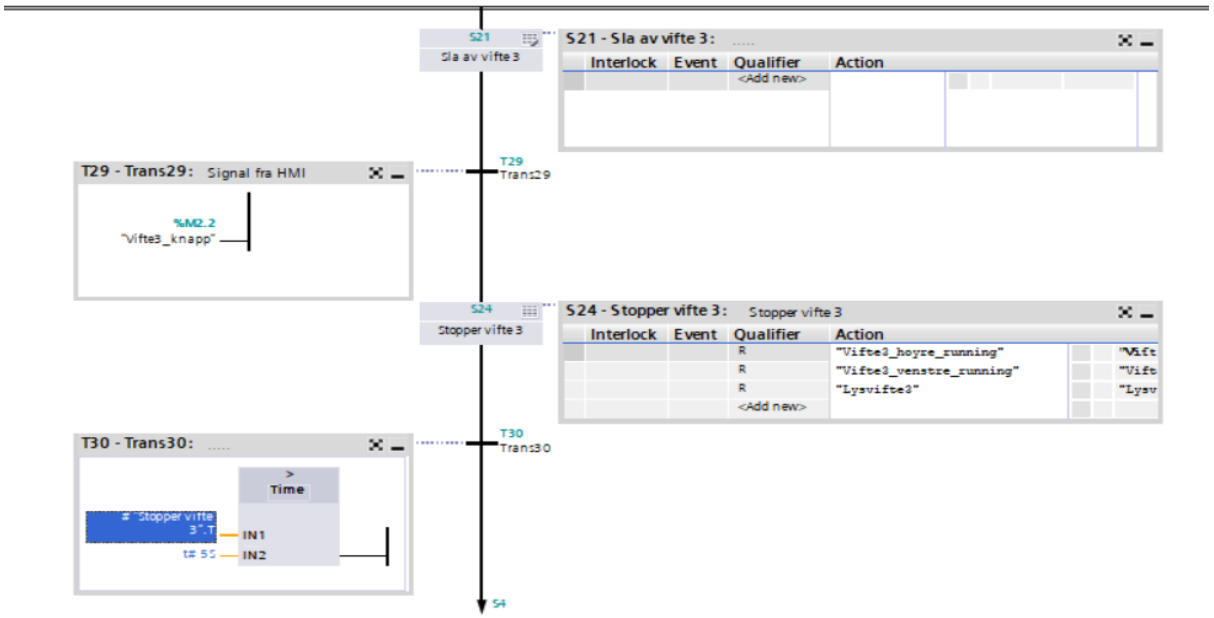
Bildet under viser neste grein. Det første vi ser er en tom step-boks, så kommer det en transition med signal fra HMI-panelet. Signalet blir sendt når knappen som heter «vifte2_knapp» blir trykket på. Når dette skjer, blir vifte 2 slått av i begge retninger, dette er noe man kan se i step-boksen. Dette blir gjort ved å sette variablene «vifte2_høyre_running» og «vifte2_venstre_running» til R, altså til 0. Til slutt er det en timer på 5 sekunder, for å sikre en fin overgang mellom alternativer i tilfelle man først trykker på for eksempel alternativ 3 i HMI-panelet og så alternativ 1. Deretter hopper programmet tilbake til begynnelsen, der HMI-panelet styrer den lokale PLSen.



Figur 25: Endelig program bilde 5

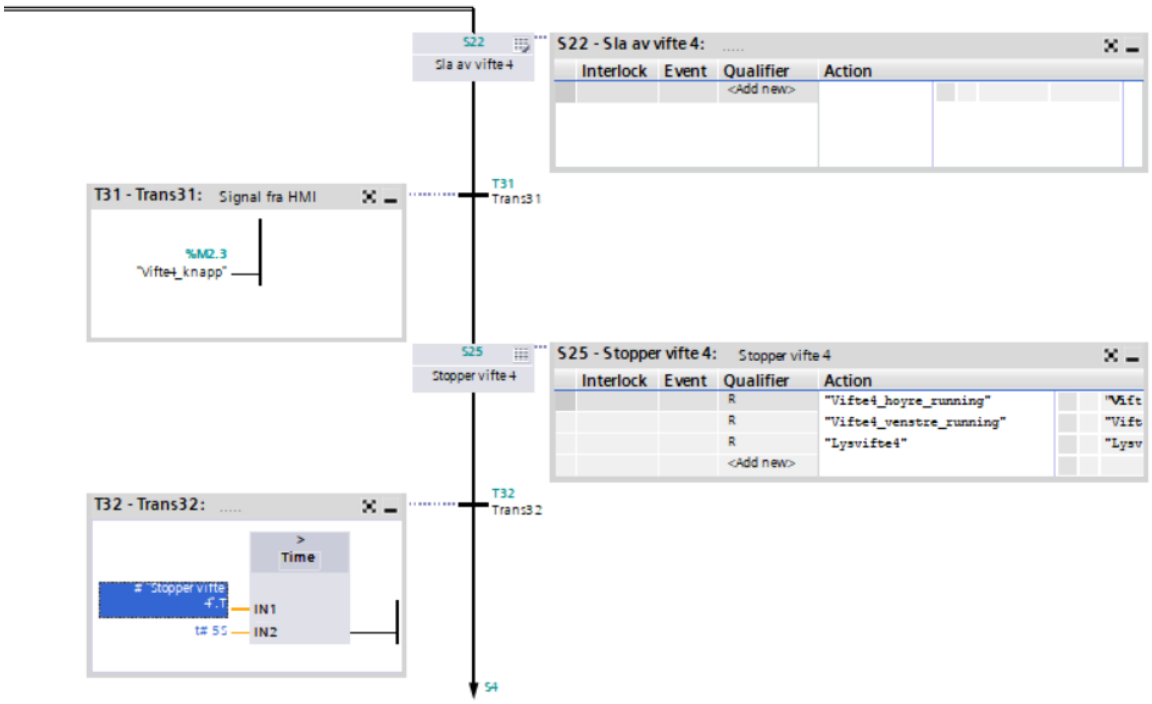
Bildet under viser neste grein. Det første vi ser er en tom step-boks, så kommer det en transition-boks med signal fra HMI-panelet. Det er en knapp som heter «vifte3_knapp», som blir trykket på, når dette skjer så kommer det en step-boks som viser at begge retninger blir slått av i vifte 3. Dette blir gjort ved å sette variablene «vifte3_høyre_running» og «vifte3_venstre_running» til 0, altså aktiv lav. Deretter kommer det en timer på 5 sekunder for å ha en sikrere overgang i tilfelle lokal PLSen får

beskjed om å hoppe til et annet alternativ. Til slutt hopper programmet tilbake til starten der HMI-panelet begynner å styre.



Figur 26: Endelig program bilde 6

Neste bilde viser siste grein. Først er det en tom step-boks som på de andre greinene, så kommer det en transition-boks med signal fra HMI-panelet. Når knappen «vifte4_knapp» blir trykket på, er det en step-boks som gjør at begge retninger blir slått av i vifte 4. Til slutt, som de på andre viftene, er det en timer og programmet hopper tilbake til starten.



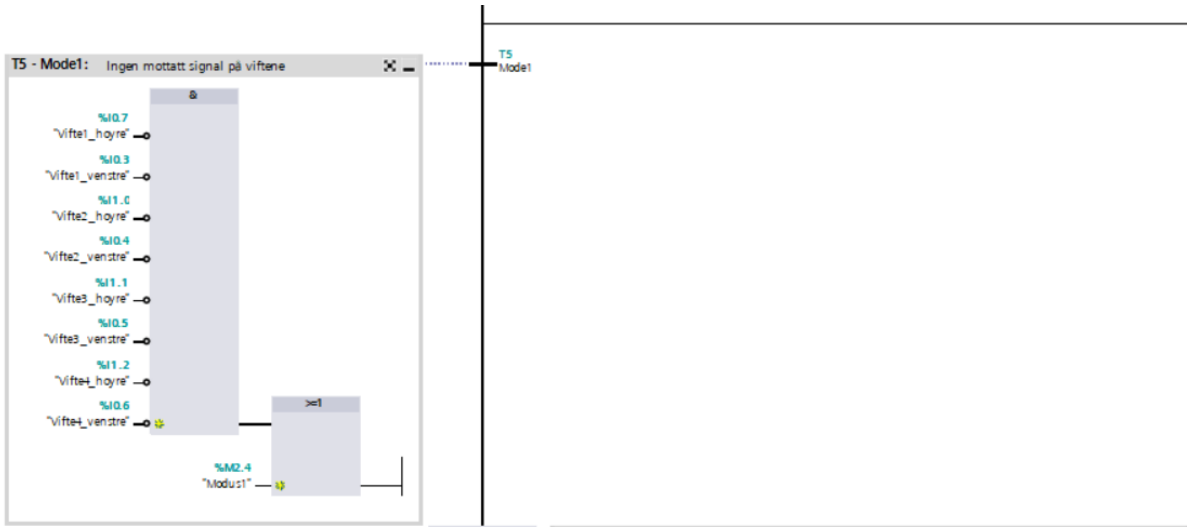
Figur 27: Endelig program bilde 7

Her under ser vi idle - boksen, hvor alle viftene blir initialisert og venter på aktivering. Alle viftene blir satt til R, reset, altså «Set til 0».

Figur 28: Bilde av Idle-boks

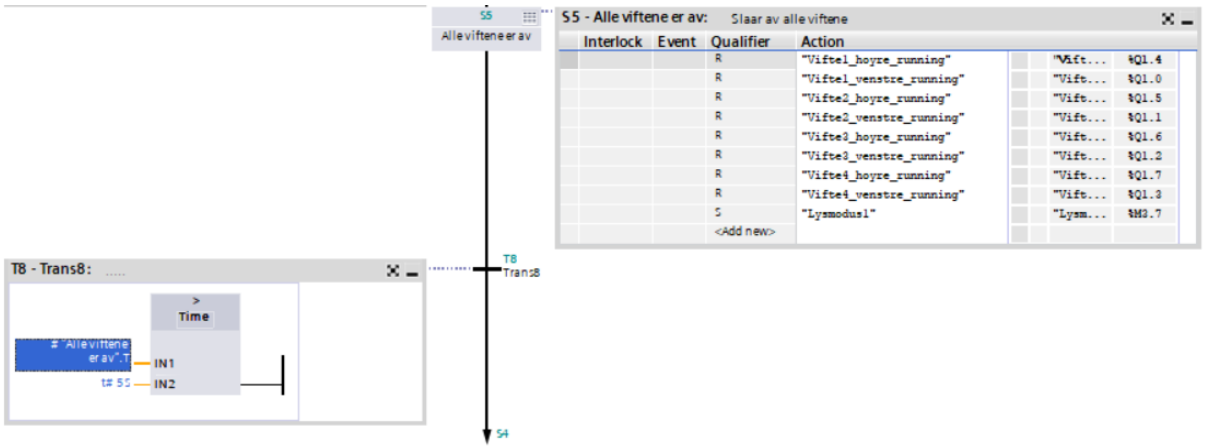
Modus 1:

Som nevnt så er det i denne modusen alle viftene skal være slått av. Dette er bygd slik at det begynner med en transition-boks, der det først er en «&-boks» etterfulgt av en «>=1-boks». «&-boksen» fungerer slik at den går videre til «>=1» (eller)-boksen hvis alle viftene er slått av begge retningene. En bedre forklaring på dette er hvis enten «&-boksen» eller HMI-panelet sender signal om at modus 1-knappen har blitt trykket på, så kan step-boksen gjøre det den skal gjøre.



Figur 29: Modus 1 bilde 1

Dette går videre til en step-boks der alle viftene blir slått av, begge retninger. De blir satt i R, altså reset (aktiv lav). Samtidig så blir «lysmodus1»-variablen satt til S, altså Set (aktiv høy). Dette betyr at lyset endres fra svart til grønt på HMI-panelet, både på vedlikeholdsbildet og lokalstyringsbildet. Dette lyset er ment for å informere en eventuell fremtidig medarbeider som skal ordne noe ved viftene eller skal bruke HMI-panelet for å styre lokalstyringen, om at alle viftene er slått av. Deretter kommer det en ny transition-boks der en timer på 5 sekunder blir satt på, for å få en fin overgang mellom viftene som blir slått på og de som blir slått av.



Figur 30: Modus 1 bilde 2

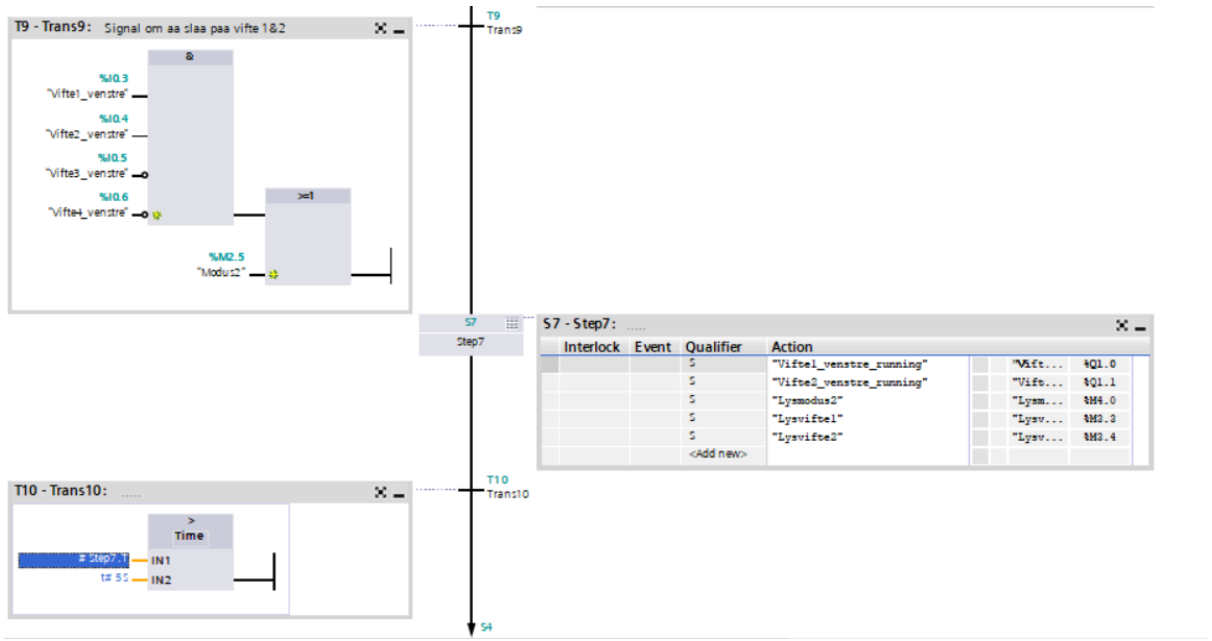
Modus 2:

Her er det vifte 1 og 2 som er slått på i retning mot venstre. Hele modusen begynner med en transition som inneholder en timer på 1 minutt for å gjøre overgangen mellom ulike moduser så lite problematisk som mulig. Deretter er det en step-boks som er blitt satt der for sikkerhet, der blir alle

viftene slått av. Dette er noe som måtte slettes i etterkant da simuleringen ble testet, fordi programmet ellers ikke ville fungere. Det måtte bli slettet fra alle moduser.

Figur 31: Modus 2 bilde 1

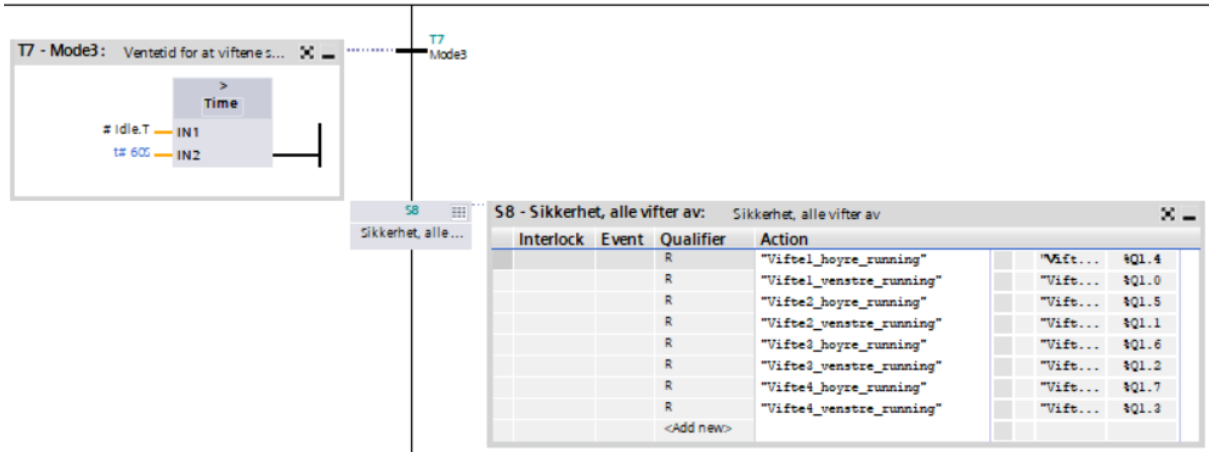
Det neste som kommer er en ny transition som begynner med en &-blokk som vil si at alle variablene må være oppfylt for at den skal gå videre til «>=1»-blokken. Signaler for både vifte 1 og 2 må være slått på mot venstre og vifte 3 og 4 mot venstre må være slått av. Hvis dette skjer eller knappen modus 2 blir trykket på i HMI-panelet, så går da signalet videre i programmet. Dette fører da videre til en ny step-boks der viftene 1 og 2 blir slått på mot venstre, samtidig som lysene både for modus 2 og lysene for viftene 1 og 2 blir slått på. Til slutt så er det en timer, av samme grunn som på de alle andre moduser. Programmet hopper så tilbake til begynnelsen.



Figur 32: Modus 2 bilde 2

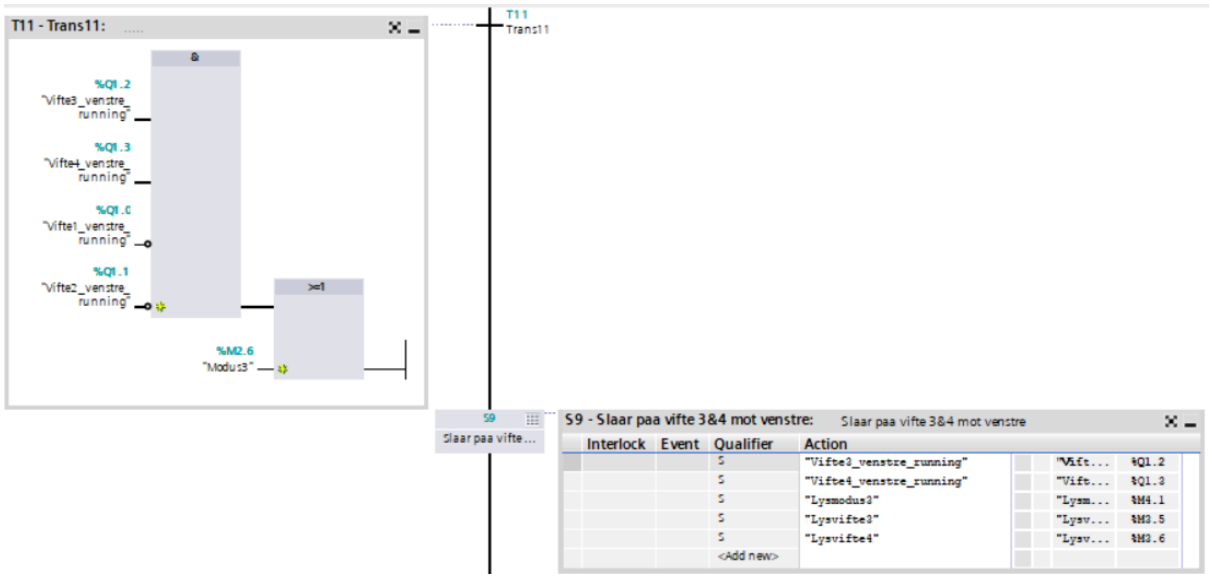
Modus 3:

Her er det vifte 3 og 4 som skal slås på mot venstre. Som alle andre moduser er det først en timer på 1 minutt. Deretter er det et sikkerhets-step som modus 2 også hadde, og en ny transition.



Figur 33: Modus 3 bilde 1

Denne transition inneholder det samme som transition etter sikkerhets-step i modus 2. Forskjellen her er at det er vifte 3 og 4 som er slått på mot venstre i stedet for vifte 1 og 2. I step-boksen som kommer etter det blir alle viftene slått av utenom de som skal være på, samtidig som lysene for modus 3 og lys for vifte 3 og 4 blir slått på. Alle disse lysene blir da vist i HMI-panelet. Til slutt er det en timer på 5 sekunder som på de andre moduser. Programmet hopper så tilbake til starten.

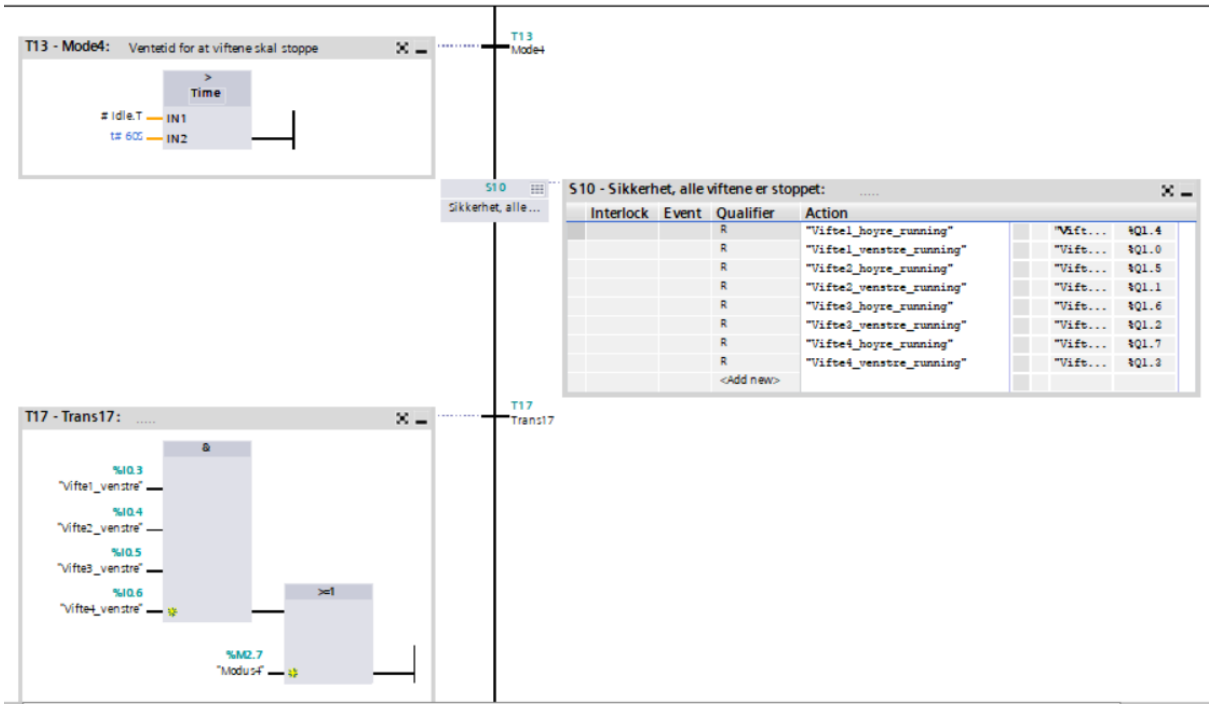


Figur 34: Modus 3 bilde 2

Figur 35: Modus 3 bilde 3

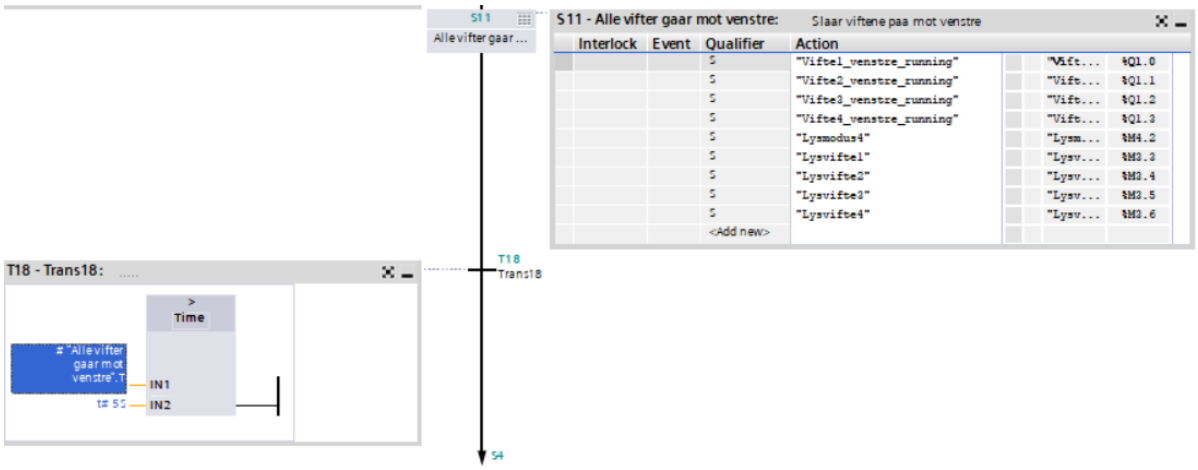
Modus 4:

Denne modusen er som nevnt tidligere i oppgaven når alle viftene er slått på i retning mot venstre. Oppbyggingen av denne modusen begynner som de alle andre, med en timer på 1 minutt. Deretter kommer sikkerhets-stepet og en ny transition. Denne transition inneholder en «&»-blokk og en «(>=1)»-blokk som de andre moduser. Forskjellen her er at det blir sendt signaler for at alle viftene skal bli slått på i &-blokken (som betyr «og»).



Figur 36: Modus 4 bilde 1

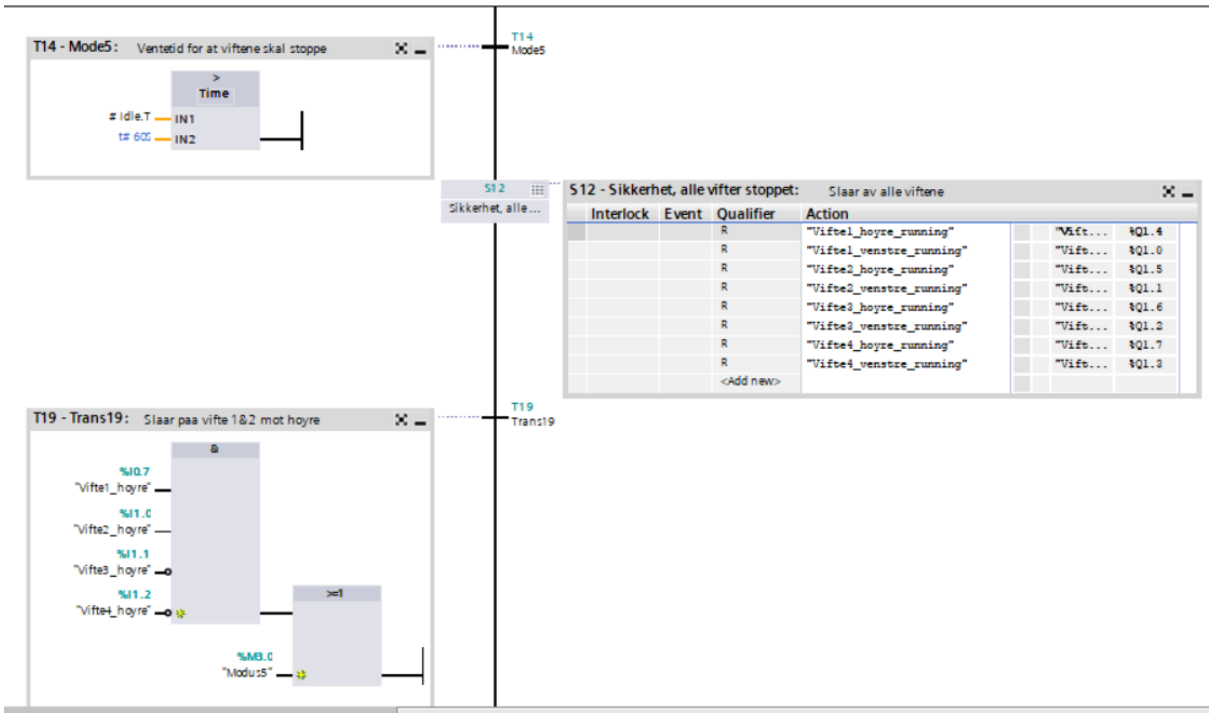
Deretter blir alle viftene slått på mot venstre, samtidig som lyset for modus 4 og lys for alle 4 viftene blir slått på. Så er det en timer og programmet hopper tilbake til starten.



Figur 37: Modus 4 bilde 2

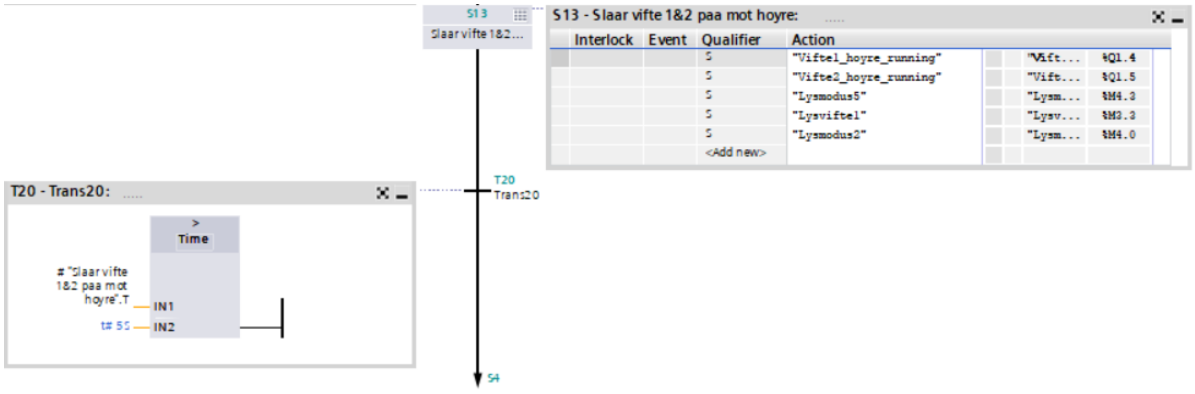
Modus 5:

I denne modusen er det vifte 1 og 2 som skal være på, men denne gangen i retning mot høyre. Dette er blitt bygget i programmet på samme måte som de andre moduser. Det som er annerledes er transition (trans19) etter sikkerhets-step og step (S13) som kommer etter det. Transition19 består av en og-blokk(&) og en eller-blokk(8>=1). I og-blokken er det fire signaler, to av de som er for vifte 1 og 2 mot høyre og de to andre er signaler for vifte 3 og 4 mot høyre, men disse er invertert. Hvis alle disse signalene er mottatt på denne måten eller knappen for modus 5 blir trykket på så går signalene videre i programmet.



Figur 38: Modus 5 bilde 1

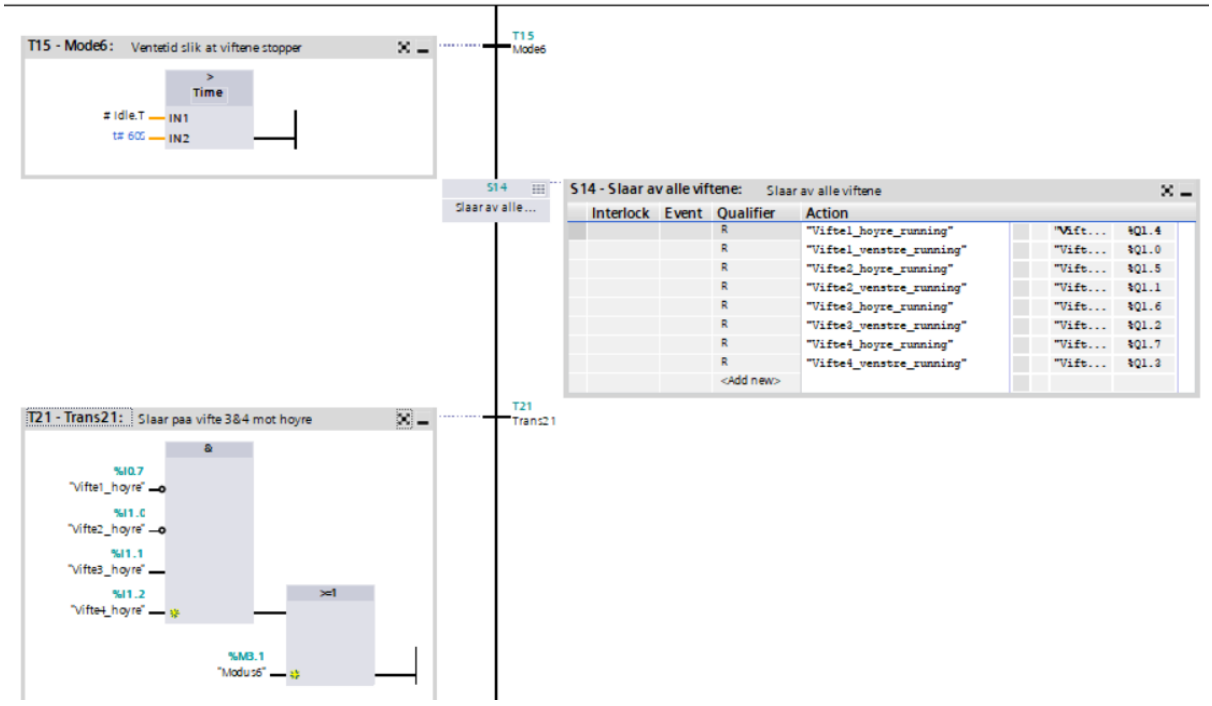
Deretter ser vi i step13 at viftene 1 og 2 blir slått på mot høyre, samtidig som lysene for modus 5 og for viftene 1 og 2 skifter til grønn. Så er det en timer på 5 sekunder og programmet hopper tilbake til starten.



Figur 39: Modus 5 bilde 2

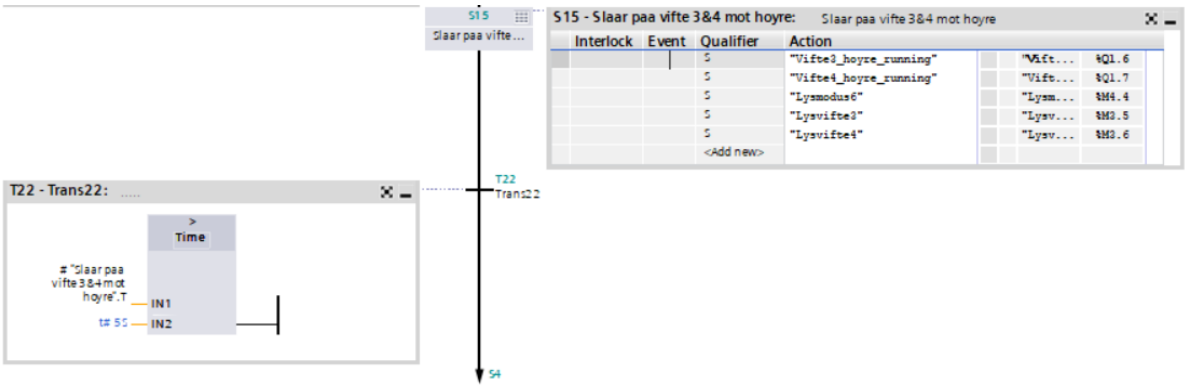
Modus 6:

Her er det vifte 3 og 4 som skal være slått på mot høyre. Denne modusen begynner med en timer på 1 min og et sikkerhets-step. Transition21 består av en og-blokk og en eller-blokk. Her er det enten signaler for at vifte 3 og 4 skal slå seg på mot høyre eller knappen modus 6 blir trykket på i HMI-panelet.



Figur 40: Modus 6 bilde 1

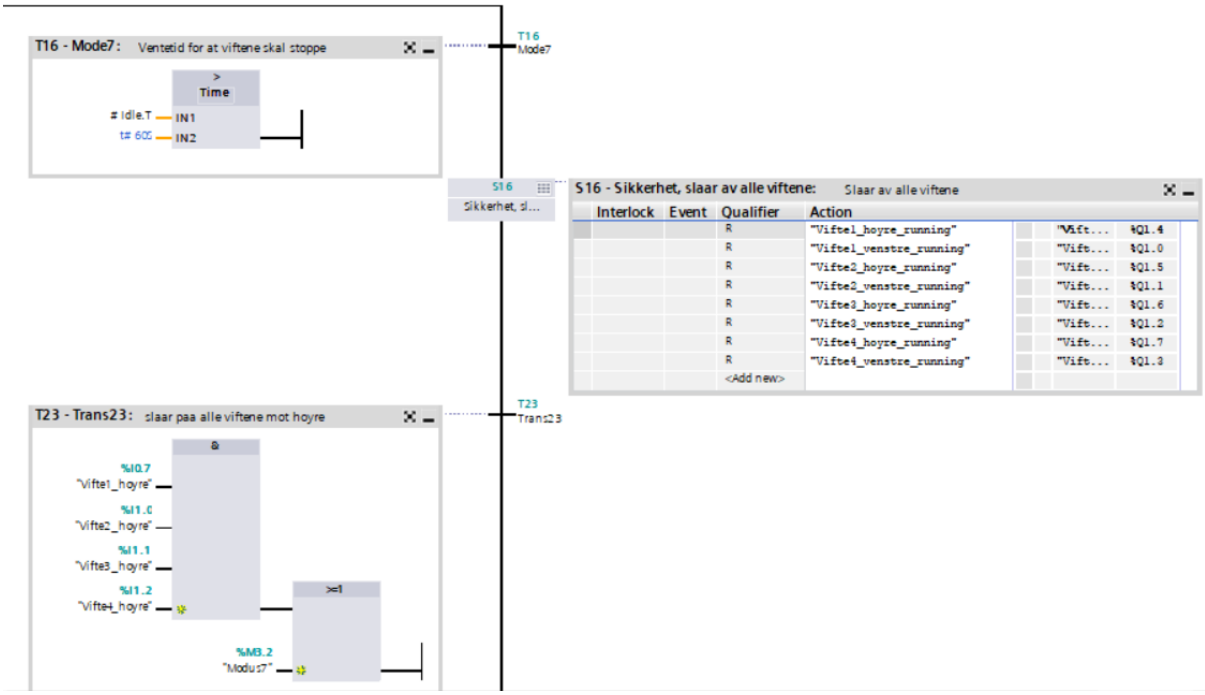
Deretter kommer step15 som utfører dette samtidig som lyset for modus 6 og lysene for vifte 3 og 4 blir slått på.



Figur 41: Modus 6 bilde 2

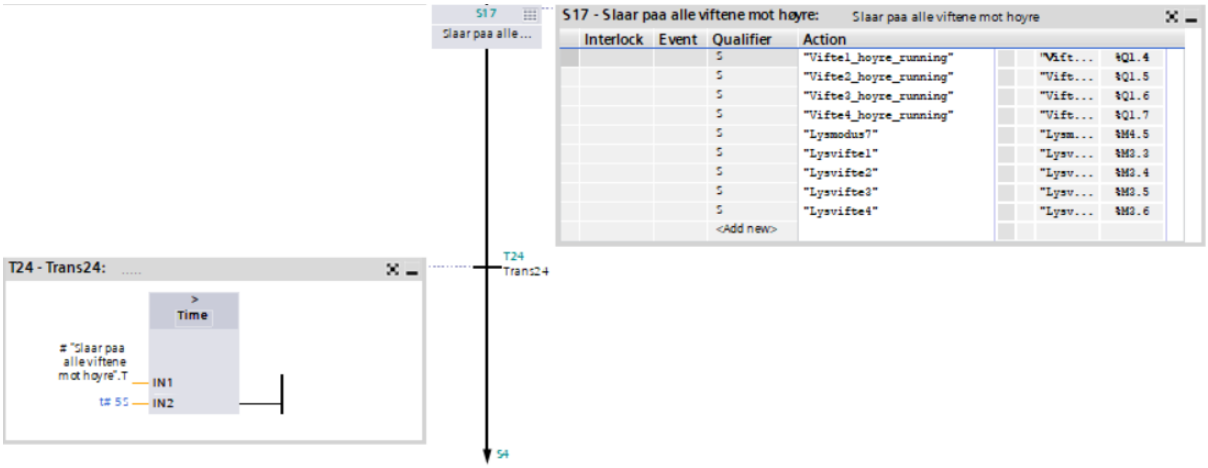
Modus 7:

Her skal alle viftene være slått på i retning mot høyre. Først er det en timer på 1 minutt og sikkerhets-step. Transition23 er likt på som de andre moduser, bare at det signalet som går videre er enten alle viftene som blir slått på mot høyre eller knappen for modus 7 blir trykket på i HMI-panelet.



Figur 42: Modus 7 bilde 1

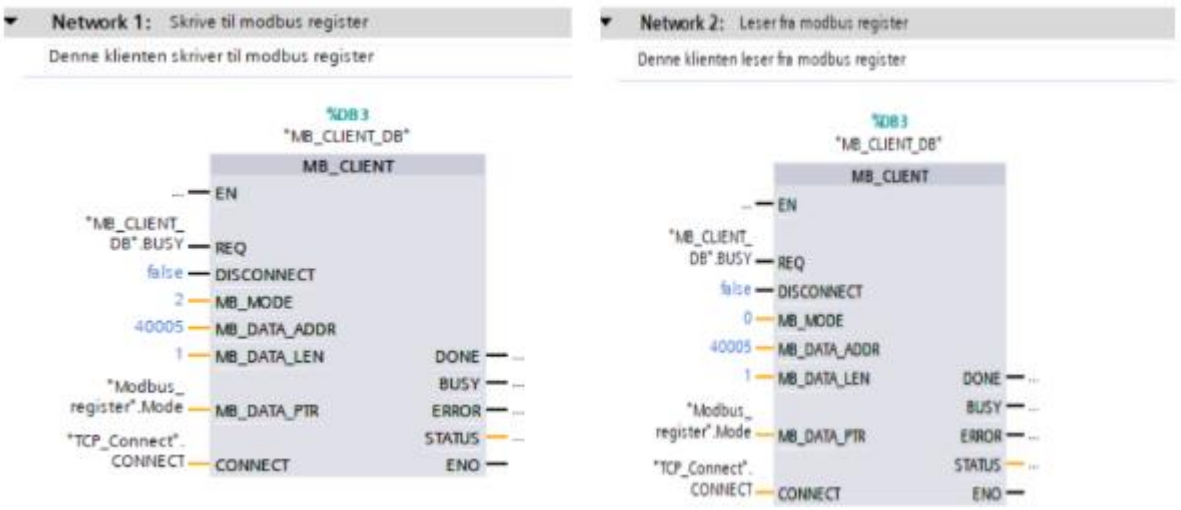
Deretter kommer step17 som viser at alle viftene blir slått på, samtidig som lysene for alle viftene og for modus 7 blir slått på. Til slutt er det en timer på 5 sekunder og programmet hopper som vanlig tilbake til starten.



Figur 43: Modus 7 bilde 2

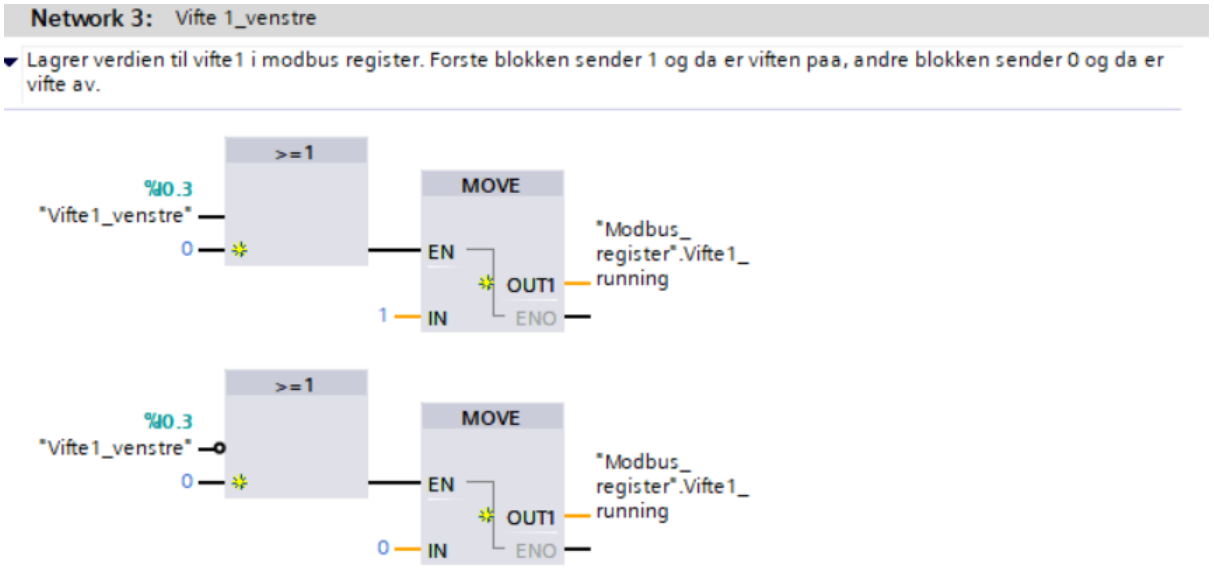
4.3.1 Kommunikasjon via modbus over TCP/IP

Kommunikasjonen skal, som nevnt tidligere i oppgaven, foregå ved hjelp av protokollen TCP/IP over modbus. Dette kjøres parallelt som hovedprogrammet. Modbuskommunikasjon fungerer slik at det er en master og en eller flere klienter. I dette tilfellet så er det 2 klienter som er vist i bilder under. En klient for skriving til modbusregisteret og en for lesing fra modbusregisteret. Modbusregisteret er en type database, der enten 0 eller 1 blir lagret. Hvis 0 blir lagret så skal viften slås av og hvis 1 blir lagret så skal viften slås på. Masteren er hoved PLS. Figur 44 viser kommunikasjonen fra lokal PLS til hoved PLS.



Figur 44: Funksjonsblokker som skriver og leser til modbusregisteret til hoved PLS

Bildet under viser hvordan 0 eller 1 blir lagret i modbusregisteret, i dette tilfellet er det vifte nummer 1 i retning mot venstre. Dette har blitt gjort med alle fire viftene i begge retninger.



Figur 45: Skriving til modbus register

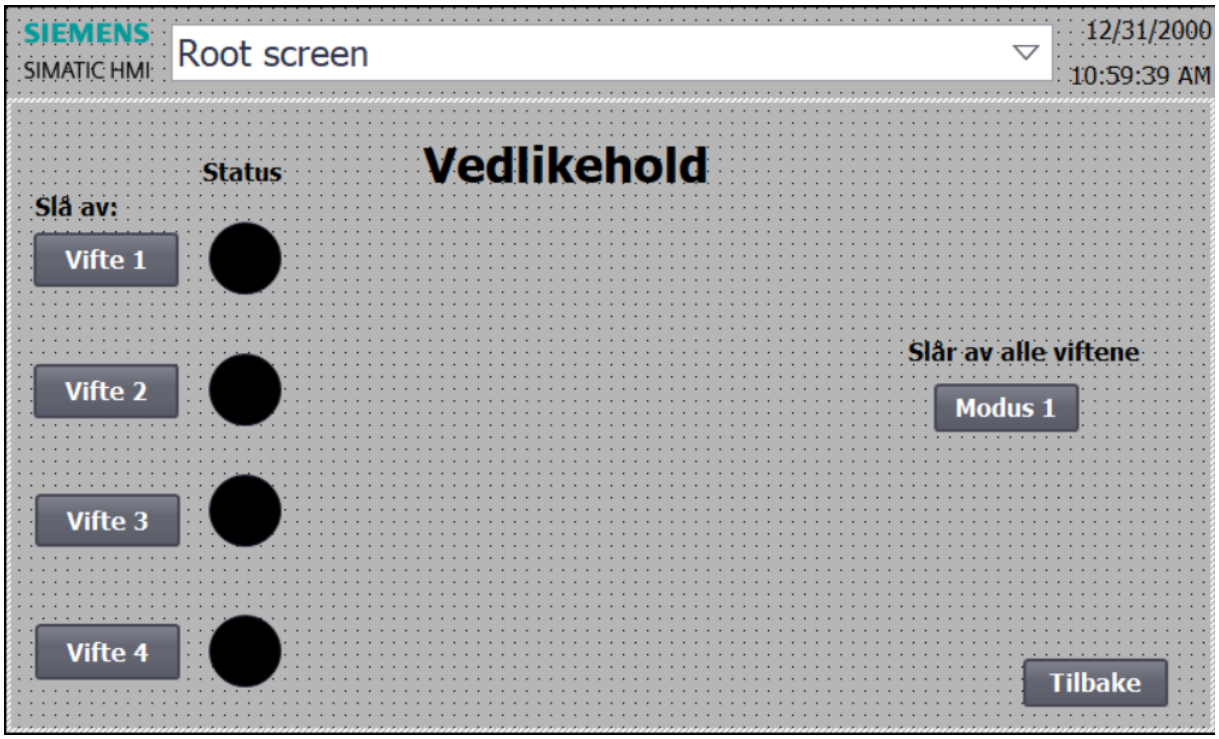
I den første «>=1»-blokk er det signal fra vifte1_venstre eller 0 som blir sendt videre til move-blokken, og dette fører til at 1 blir lagret i modbusregisteret. Dette kan man se i inngangen «IN» og i utgang «OUT1» i MOVE-blokken. Det samme skjer med signal fra vifte1_venstre, men denne gangen invertert slik at det er 0 som blir lagret i modbusregisteret.

4.3.2 Programmering av HMI-panelet

I dette delkapittelet forklares valg av utseende og hvordan HMI-panelet ble programmert. Figur 46 under viser startbildet/hjemskjermen til HMI-panelet, også kalt «Root screen». Her kan operatøren velge hva som skal skje med ventileringen i tunnelen. I oppgaven har vi valgt å klassifisere to valg. Det ene valget er vedlikehold og det andre valget er lokal styring. Det er tatt utgangspunkt i et enkelt design som er lett å navigere i etter ønske fra oppdragsgiver.

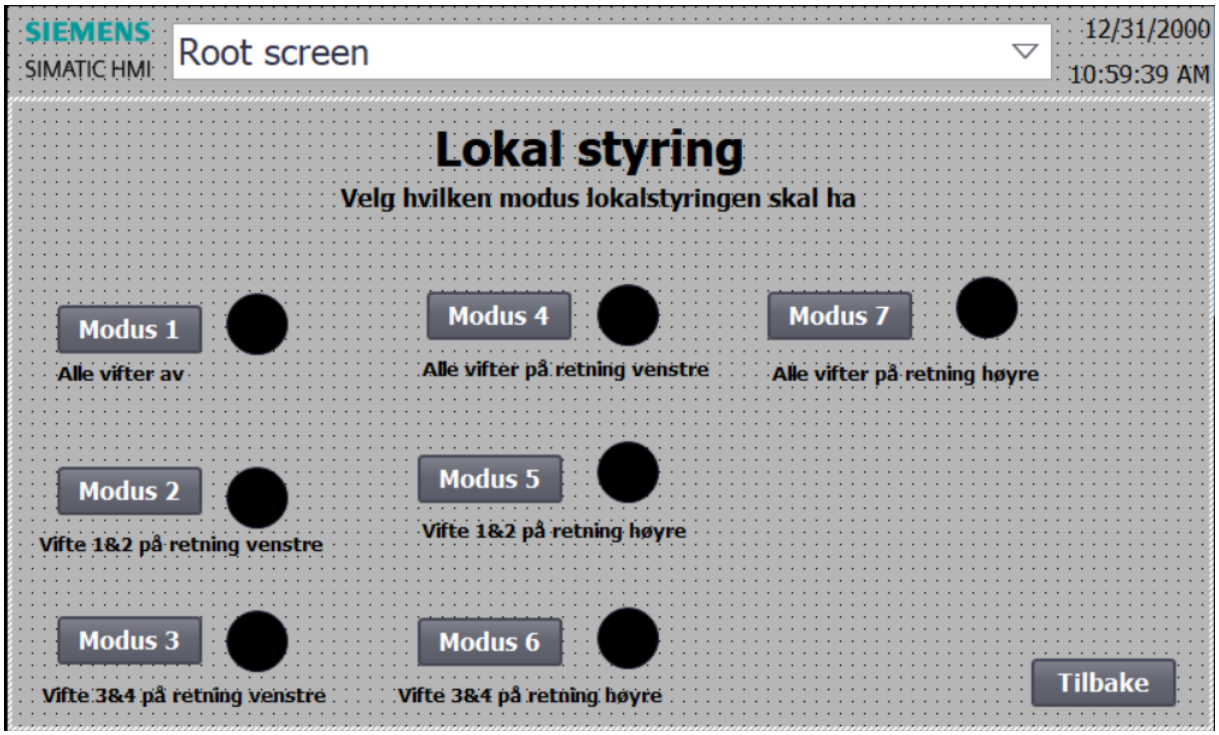
Figur 46: Hjem-skjerm

Når en trykker inn på knappen «Vedlikehold» kommer en ny skjerm opp. Dette skjermbildet heter vedlikehold. Her er det lagt inn knapper for å stoppe hver enkelt vifte, en knapp for å slå av alle viftene samtidig, en tilbake-knapp (som fører tilbake til startbildet/hjemskjerm), og statuslys. Statuslysene vil lyse grønt ved de viftene som er i drift, og vil være svarte dersom viftene ikke er i drift. Dette gjør det enklere å se hvilke vifter som faktisk er i gang, samtidig som det gjør løsningen mer sikker, da operatør kan se hvilke vifter som er i drift, før et eventuelt vedlikehold starter.



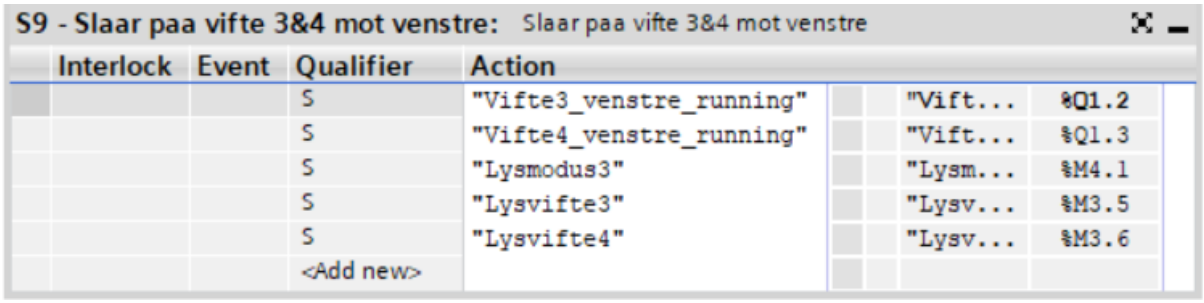
Figur 47: Vedlikehold

Bilde 48 under viser skjermen som kommer opp når man trykker inn på knappen «Lokal styring». Dette skjermbildet heter lokal styring. Her er det lagt inn ulike moduser (som nevnt i kapittel 4.3), og tilhørende statuslys. Statuslysene vil også her lyse ved den modusen som kjøres, dette gjør det enkelt for operatør å se hvilken modus som kjører, og gjør det lett å bytte om det skulle være ønskelig.



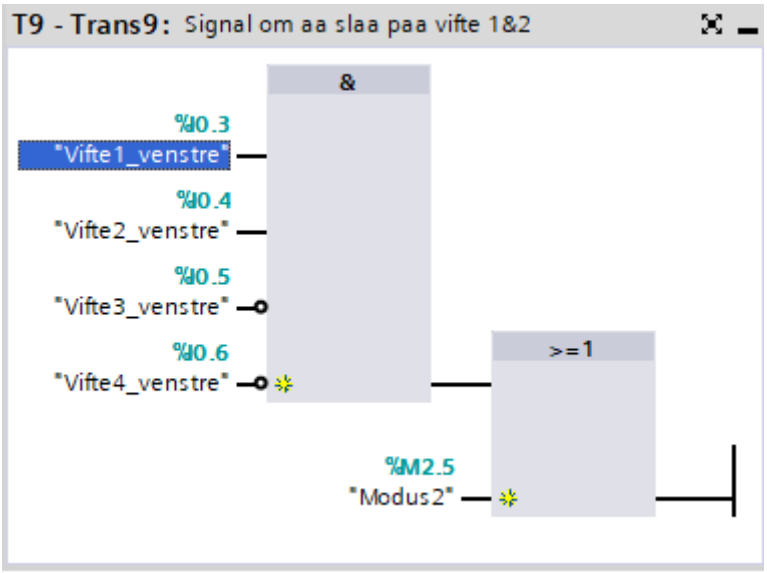
Figur 48: Lokal styring

Programmeringen av HMI-panelet er gjort slik at den er knyttet opp mot hovedprogrammet. Det ble opprettet variabler/tags for lysene i tabellen for hovedprogrammet, og det ble lagt inn i ulike steps/transitions i hver grein i programmet. Bildet under er et eksempel på hvordan statuslysene er knyttet opp mot programmet.



Figur 49: Eksempelbilde med statuslys

Her vil statuslys for modus 3 lyse inne på lokalstyringsskjermen på HMI-panelet, samtidig vil lys for vifte 3 og 4 lyse inne på vedlikeholdsskjermen. Grunnen til at lampene vil lyse på begge steder er for at operatør skal kunne se hvilke vifter som går uavhengig av hvilket valg som blir trykket på i hjemskjermen. For at dette skulle fungere ble variablene som er opprettet i hovedprogrammet lagt inn som aktivatorer for statuslysene, så når det blir sendt en «logisk 1» eller aktiv høy, så vil lysene slå seg på og lyse grønt, og motsatt hvis det blir sendt en «logisk 0» eller aktiv lav, da vil de lyse svart. Variablene for lysene er lagt inn i step-blokkene i programmet, siden de er signaler som skal utføre en funksjon.



Figur 50: Eksempelbilde knappaktivering

Bildet over er et eksempel på hvordan knappene er implementert i programmet. Variablene for knappene er også opprettet i tabellen til hovedprogrammet, men tillagt egenskaper inne i HMI-panelet. Når en knapp blir trykket på, genererer dette en «logisk 1» eller aktiv høy, og sender dette til hovedprogrammet. Signal fra knappene i HMI-panelet mottas i transition-delen av programmet, siden de er signaler som skal aktivere en funksjon videre.

5 Beregninger, kostnader og flerlinjeskjema

5.1 Beregninger

I dette delkapittelet utfører vi beregninger ved hjelp av Mean Time Between Failure. Vi tar utgangspunkt i at systemet er koblet i serie, og beregner ved hjelp av formelen for MTBF i serie (se kapittel 2.5). Utregningene er kun for PLS-systemet.

Strømforsyning:
Produktnummer: PS 25W 24VDC 6ES7 505-0KA00-0AB0
MTBF: 28 år = 245 280 timer

PLS:
Produktnummer: PLC 6ES7 516-3FN01-0AB0
MTBF: 25,6 år = 224 256 timer

Innganger:
Produktnummer: DI 32x24 VDC HF_1 6ES7 521-1BL00-0AB0
MTBF: 90,53 år = 793 043 timer

Utganger:
Produktnummer: DQ 32x24 VDC/0.5A ST_1 6ES7 522-1BL00-0AB0
MTBF: 94,66 år = 829 222 timer

HMI Panel:
Produktnummer: TP700 Comfort
MTBF: 7,7 år = 67 453 timer

Formelen under viser beregning av MTBF for alle komponentene.

$$\frac{1}{MTBF_{TOT}} = \frac{1}{245280} + \frac{1}{224356} + \frac{1}{793043} + \frac{1}{829222} + \frac{1}{67453}$$

$MTBF_{TOT}$ = 38 720 timer = 4,4 år

Figur 51: Beregning av MTBF

Tallene for hver enkelt komponent er hentet fra ulike bedrifter sine hjemmesider [13] [14] [15].

5.2 Kostnader

Som nevnt i kapittel 2.1 ønsker oppdragsgiver at det skal sees på produksjonstiden til tavlene som er i bruk i dag, samt se på komponentpriser og produksjonstiden på systemet. Det vi har valgt å gjøre i denne oppgaven på grunn av tids- og oppgavebegrensning, er å se på komponentprisene til de komponentene vi bruker og har sett på i vår løsning. Det er viktig at totalkostnaden for løsningen er konkurransedyktig. Komponenter som må regnes med er:

Komponent	**Pris pr stk** ***verdier regnet ut fra dollar og euro**
PLS S7-1500 (Uten CPU) [16]	30592,10 kr
CPU[3]digitale innganger [17]	3034,07 kr
CPU digitale utganger [18]	4393,59 kr
HMI-panel TP700 Comfort [19]	8632,95 kr
Kontaktor TeSys-035477 [20]	656,81 kr
Lisens til TIA portal[4] [21]	22429,84 kr
SUM	**69736,36 kr**

Figur 52: Tabell over kostnader

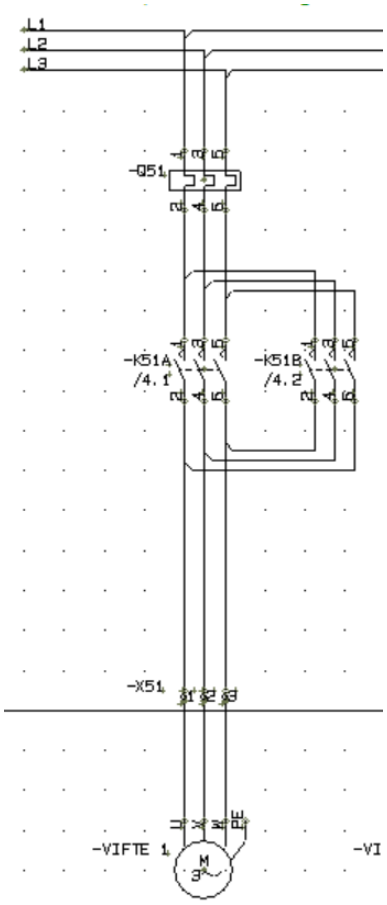
Tabellen over viser prisene for de ulike komponentene i løsningen, og prisen er per stykk. Vi har tatt utgangspunkt i Siemens sine PLS-komponenter, da det er disse vi bruker i løsningen vår. Som tabellen viser koster en komplett PLS en god del, og dette vil da være en god investering siden PLSen har lang

[3] CPU – digitale innganger og utganger
[4] Lisens til TIA portal kjøpes kun en gang

levetid (se kapittel 5.1). En PLS-løsning er ganske drift sikker, da den vil kjøre det programmet som er lastet opp på den, frem til den mottar beskjeder fra hoved PLS om å gjøre noe annet, eventuelt at det blir byttet vifte-modus via HMI-panelet.

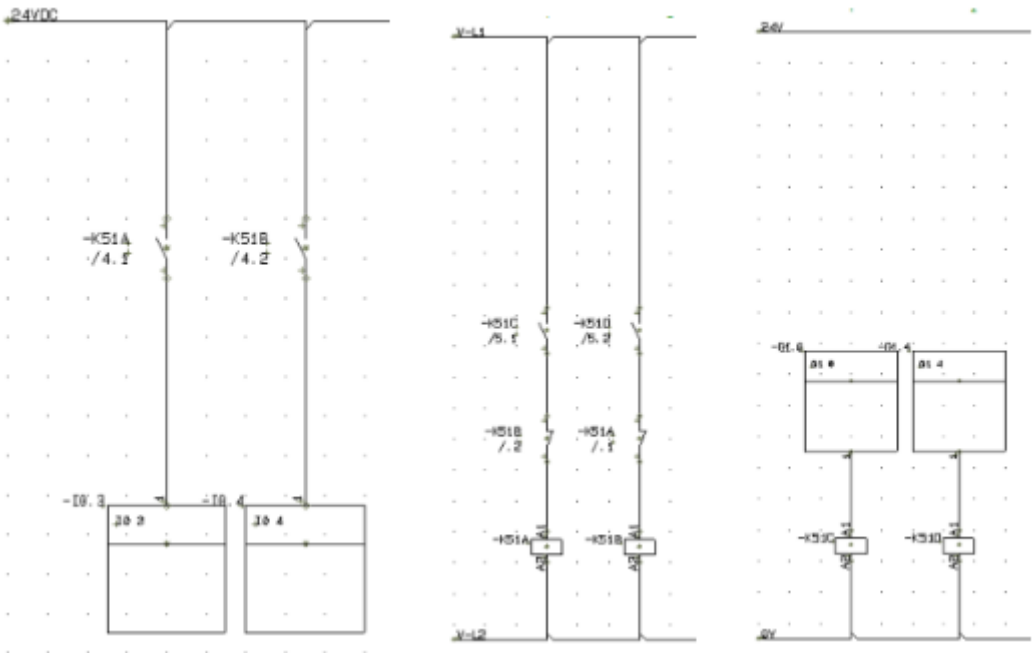
5.3 Flerlinjeskjema

I dette kapittelet presenterer vi flerlinjeskjema for løsningen vår etter ønske fra oppdragsgiver.



Figur 53: Flerlinjeskjema bilde 1

Bildet over viser hovedstrømskjema for en av de fire viftene. Komponent Q51 er motorvernet, også kalt termisk vern. Denne typen automatisk bryter benyttes i forbindelse med elektriske motorer for å hindre at viklingene inne i motoren blir ødelagt dersom strømmen blir større enn innstilt strømstyrke. Komponentene K51A og K51B er kontaktorer som er koblet slik av motoren til viftene kan kjøres begge veier. X51 er rekkeklemmer som kobles fra kontrollskapet til viftemotoren.



Figur 54: Flerlinjeskjema bilde 2

Bildene over viser et utklipp fra styrestrømsskjema for en av de fire viftene. I0.3 og I0.4 er inngangene til PLSen, Q1.0 og Q1.4 er utgangene. Bildet i midten inneholder forriglinger vi har lagt inn i systemet. En forrigling er en metode for å forebygge uønskede tilstander i et system, i vårt tilfelle vil dette være at en viftemotor blir startet i begge retninger samtidig. Det har også blitt lagt til et hjelperelé på hver av utgangene til PLSen for å legge inn kontaktorene.

6 Expo

Expo er en utstilling der studentene kan vise frem bachelorprosjektene sine. Til utstillingen har vi laget en liten modell som simulerer løsningen vår. Dette var noe vi ferdigstilte etter at vi var ferdige med programmeringen av hovedprogrammet og HMI-panelet. For at løsningen skulle bli så realistisk som mulig, og ligne på et ferdig system, trengte vi en motor, vifte/propell og kontaktorer (24V DC). Vi måtte bruke tre kontaktorer for å kunne styre motoren. Motoren og viften fungerer som en vifte i en tunnel. Vi måtte også lage et mindre program for å få dette til å fungere på modellen vår.

7 Analyse av oppgaven

I dette kapittelet utfører vi en kort analyse av beregningen og kostnadene, der enkelte komponenter belyses og trekkes frem som positive oppgraderinger.

Beregninger av det komplette PLS-systemet viser at det vil ha en MTBF på ca. 38 720 timer. Dette vil si at det i gjennomsnitt vil gå ca. 4,4 år mellom hver gang hele systemet vårt er nede eller at det blir noe feil med det, eventuelt at det trenger vedlikehold.

Enkeltkomponenter som PLS og HMI-panel har også en egen MTBF. MTBF for PLS er på 25,6 år og for HMI-panelet er MTBF på 7,7 år. For PLS som tar over styringen er dette en veldig lang levetid mellom feil, noe som vil si at om systemet svikter, vil det mest sannsynlig ikke være på grunn av PLSen, men heller svikt et annet sted i systemet.

Betjeningen, altså HMI-panelet har som nevnt en levetid på 7,7 år ved daglig bruk, noe som er relativt lang tid for et panel med touch-skjerm. HMI-panelet vil mest sannsynlig ha lengre levetid enn MTBF-beregningene viser, siden dette er noe som ikke blir brukt hver dag, men kun til lokal styring av ventilasjonen.

En PLS uten ekstra CPUer koster ca. 30 592,10 kr, og hvis vi deler dette på levetiden får vi:

$$\frac{30592{,}10kr}{25{,}6år} \approx 1195{,}004\ kr\ per\ år$$

Figur 55: Beregning av årspris, PLS

Hvis PLSen lever i 25,6 år så vil årsprisen for PLSen være ca. 1200kr, noe som er en veldig god langsiktig investering (ser bort fra avkastning og nedskrivninger).

HMI-panelet koster 8632,95 og har en levetid på 7,7 år. Dette vil gi en årspris på:

$$\frac{8632{,}95kr}{7{,}7år} \approx 1121{,}2\ kr\ per\ år$$

Figur 56: Beregning av årspris, HMI

Dette er også en god langsiktig investering, da prisen for hvert år er på ca. 1121 kr (ser også her bort fra avkastning og nedskrivninger).

Det nye systemet er en dyr, men samtidig en bra oppgradering av komponentene med tanke på levetid. Hvis komponentene holder seg fungerende i de årene MTBF viser, vil dette være en god investering, siden årsprisen for komponentene blir lav.

8 Konklusjon

Vi vil konkludere med å si at vi har, etter vår beste evne, prøvd å lage et mer moderne og enklere system for lokal ventilasjonsstyring, samtidig som den er sikker og har lang levetid. Vi har programmert en PLS til å gjøre det samme som flere tidsrelé og forriglinger gjør i den gamle løsningen. Samtidig har vi også programmert et HMI-panel til å gjøre det samme som en tavle med reléstyring for hver enkel vifte. Vi har programmert begge delene i TIA Portal V15.

Gruppa har fått en bedre forståelse for hvordan PLS og HMI-panel skal programmeres, både hver for seg og sammen. Vi har også fått en bedre forståelse for hvordan det er å knytte variabler fra PLS-programmet til knapper som er blitt opprettet i HMI-panelet. Vi har lært hvordan kommunikasjonen modbus over TCP/IP fungerer og hvordan dette implementeres i et PLS-program. Samtidig har vi også lært hvordan kontaktorer fungerer, hvordan disse brukes i ulike systemer og hvordan disse kan fungere som bryter for å slå av og på en vifte.

Vi har dessverre ikke fått tid til å gjøre alt som var planlagt i utgangspunktet, som var å teste programmet i en ekte tunnel og å lære oss hvordan tavler blir designet og bygget og hvordan få ned produksjonstiden på disse. Vi har heller ikke regnet ut levetid på den gamle løsningen for så å kunne sammenligne med levetiden på vår løsning. Alt dette er noe som kan være en del av en fremtidig bacheloroppgave der fremtidige studenter setter seg inn i hvordan tavler fungerer, får ned produksjonstiden på dem og sammenligner levetider på begge løsninger.

9 Utfordringer

I dette kapittelet forteller vi kort om ulike utfordringer vi har støtt på under arbeidet med oppgaven.

Testing av løsningen i TIA Portal var en av de største utfordringene vi møtte på under arbeidet med oppgaven. Når programmet var ferdig programmert med sekvensene, kommunikasjon over modbus TCP/IP, og HMI-panelet, skulle dette testes i simulatoren i TIA Portal. Ved de første forsøkene var det ingen ting som skjedde, og sekvensen sto kun i initialiseringsstepet, og kom seg ikke videre. Etter feilsøking og hjelp fra Goodtech fikk vi til å kjøre sekvensene manuelt ved å hoppe til de ulike steppene og transitions som vi ville teste og deretter manipulere inngangsverdiene. Da fikk vi bekreftet at programmet fungerte slik vi ville.

Etter dette begynte vi å undersøke selv hvorfor simuleringen ikke fungerte. Da fant vi ut at vi må ha et tomt step inne på hoved PLS sin grein, for at TIA Portal skal kunne kjøre sekvensen automatisk. Vi måtte også fjerne sikkerhetsstepene for å kunne sende inn logisk 1 og logisk 0, slik at viftene ble simulert som på og av. Etter disse endringene fikk vi testet programmet med de ulike modusene, og da fungerte alt slik det skulle. Vi har valgt å ha med sikkerhetsstepene i kapittelet 4.3, siden disse er viktige for å forsikre seg om at de riktige viftene går, og at de andre er avslått før en ny modus starter.

En annen utfordring var å finne tallene for å beregne MTBF, dette var noe vi også måtte få hjelp til fra Goodtech. Mange av tallene ligger gjemt i dokumenter vi ikke hadde tilgang til, og noen av dokumentene var vanskelig å tolke.

10 Referanser

[1] P. B. Andersen, «Store Norske Leksikon,» 26 August 2018. [Internett]. Available: https://snl.no/automatisering. [Funnet 22 Mai 2019].

[2] Goodtech, «Goodtech,» [Internett]. Available: https://www.goodtech.no/nb-NO/about-goodtech/history/. [Funnet 6 Februar 2019].

[3] Wikipedia, «Wikipedia, den frie encyklopedi,» 2 November 2017. [Internett]. Available: https://no.wikipedia.org/wiki/Goodtech. [Funnet 6 Februar 2019].

[4] Techopedia, «Techopedia,» [Internett]. Available: https://www.techopedia.com/definition/10318/relay. [Funnet 6 Februar 2019].

[5] K. A. Rosvold, «Store Norske Leksikon,» 20 Juni 2018. [Internett]. Available: https://snl.no/PLS_-_prosessdatamaskin. [Funnet 6 Februar 2019].

[6] Unitronics, «Unitronics,» [Internett]. Available: https://unitronicsplc.com/what-is-plc-programmable-logic-controller/. [Funnet 8 Februar 2019].

[7] S. Electric, «Schneider Electric,» [Internett]. Available: https://www.se.com/no/no/product-range-presentation/574-modbus/. [Funnet 6 Februar 2019].

[8] S. Electric, «Schneider Electric,» [Internett]. Available: https://www.schneider-electric.co.uk/en/faqs/FA293162/. [Funnet 13 Februar 2019].

[9] I. automation, «Inductive automation,» [Internett]. Available: https://www.inductiveautomation.com/resources/article/what-is-hmi. [Funnet 13 Februar 2019].

[10] S. Olsen, «Nasjonal Digital Læringsarena,» 23 Mars 2018. [Internett]. Available: https://ndla.no/subjects/subject:16/topic:1:140012/topic:1:194156/resource:1:142180. [Funnet 13 Februar 2019].

[11] Wikipedia, «Wikipedia, den frie encyklopedi,» 14 September 2017. [Internett]. Available: https://no.wikipedia.org/wiki/Kontaktor. [Funnet 13 Februar 2018].

[12] PCSCHEMATIC, «PCSCHEMATIC,» [Internett]. Available: 1.

[13] I. O. S. SIEMENS, «SIEMENS,» [Internett]. Available: https://support.industry.siemens.com/cs/document/16818490/mean-time-between-failures-(mtbf)-list-for-simatic-products?dti=0&lc=en-WW. [Funnet 21 Mai 2019].

[14] P.-d. ABB, «ABB,» [Internett]. Available: https://new.abb.com/products/en/1SAM550000R1008/ms495-75-manual-motor-starter. [Funnet 21 Mai 2019].

[15] S. Electric, «Schneider Electric,» [Internett]. Available: https://www.schneider-electric.com/en/product/LC1D18BD/tesys-d-contactor---3p%283-no%29---ac-3---%3C%3D-440-v-18-a---24-v-dc-coil/. [Funnet 21 Mai 2019].

[16] P. City, «PLC City,» [Internett]. Available: https://www.plc-city.com/shop/en/siemens-simatic-s7-1500-cpu/6es7516-3an01-0ab0.html. [Funnet 21 Mai 2019].

[17] N. Instrument, «NEX Instrument,» [Internett]. Available: https://www.nexinstrument.com/6ES7521-1BL00-0AB0. [Funnet 21 Mai 2019].

[18] L. &. E. G. LEG, «LegElectrical,» [Internett]. Available: http://legelectrical.com/brand-siemens/simatic-s7-300%2Cdigital-module-sm-327/sku-V4319-6es73271bh000ab0. [Funnet 21 Mai 2019].

[19] P. City, «PLC City,» [Internett]. Available: https://www.plc-city.com/shop/en/siemens-simatic-hmi-comfort-panels/6av2124-0gc01-0ax0.html. [Funnet 21 Mai 2019].

[20] R. Components, «RS Components,» [Internett]. Available: https://no.rs-online.com/web/p/products/3949807/?grossPrice=Y&cm_mmc=NO-PLA-DS3A-_-google-_-CSS_NO_NO_Automation_And_Control_Gear-_-Contactors%7CContactors-_-PRODUCT_GROUP&matchtype=&pla-394874423478&gclid=Cj0KCQjww47nBRDlARIsAEJ34bmEtTRh8Gi1ybR1kzjjsKgoX. [Funnet 21 Mai 2019].

[21] TPAutomation, «TPAutomation,» [Internett]. Available: http://www.tpautomation.de/Automation-systems/SIMATIC-Software/TIA-Portal-STEP-7/6ES7822-1AA05-0YA5-STEP-7-Professional-V15-1::33646.html?language=en. [Funnet 21 Mai 2019].

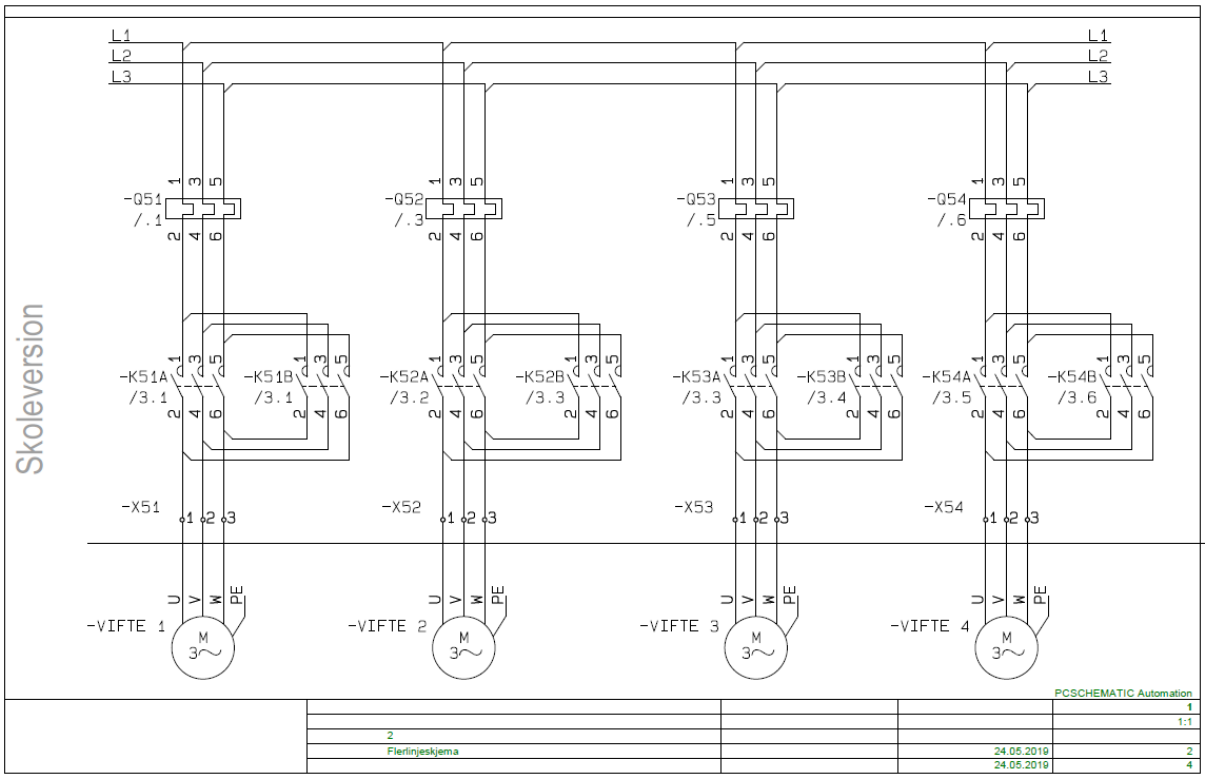
Appendiks A Vedlegg

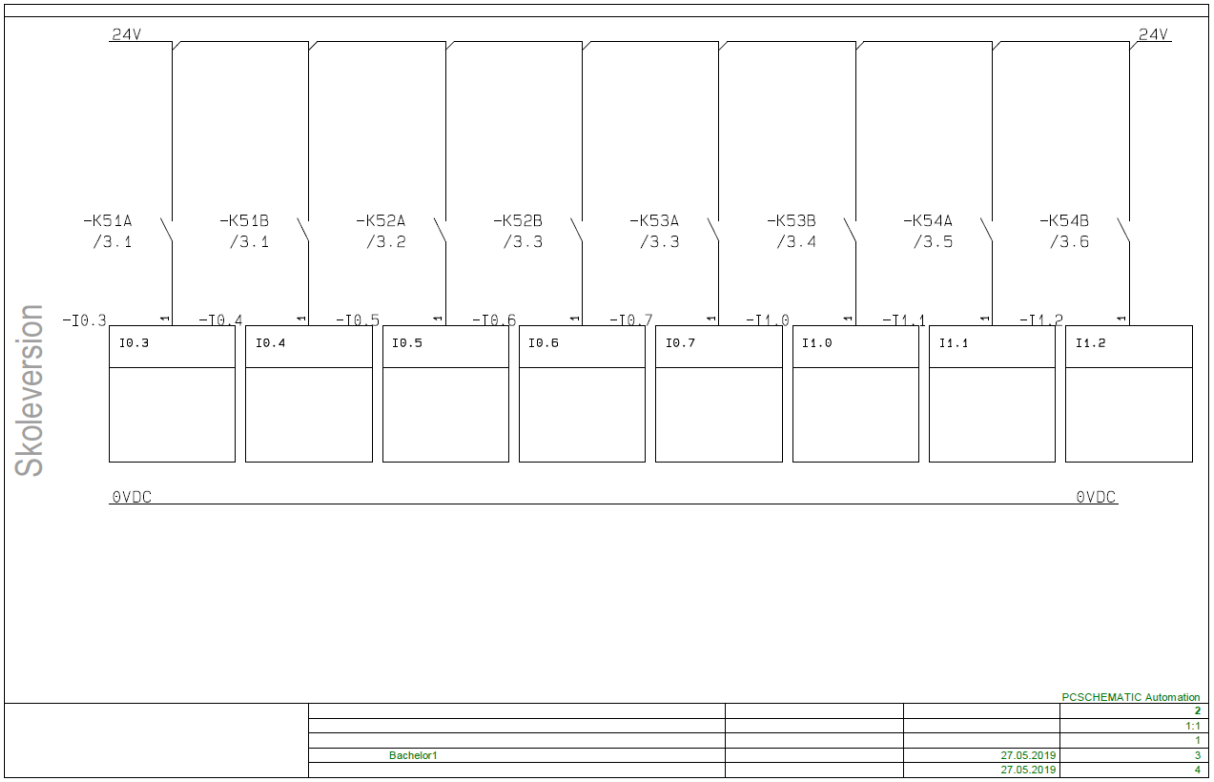
Variabler for PLS programmet.

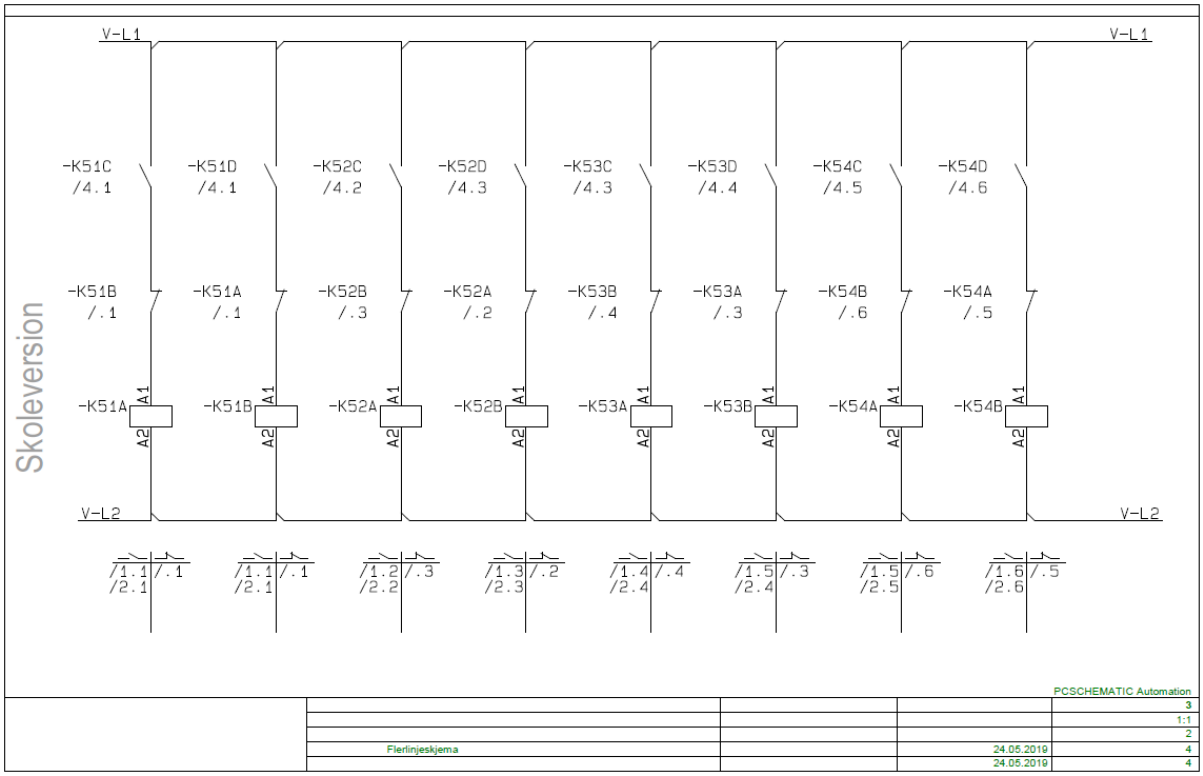
Default tag table

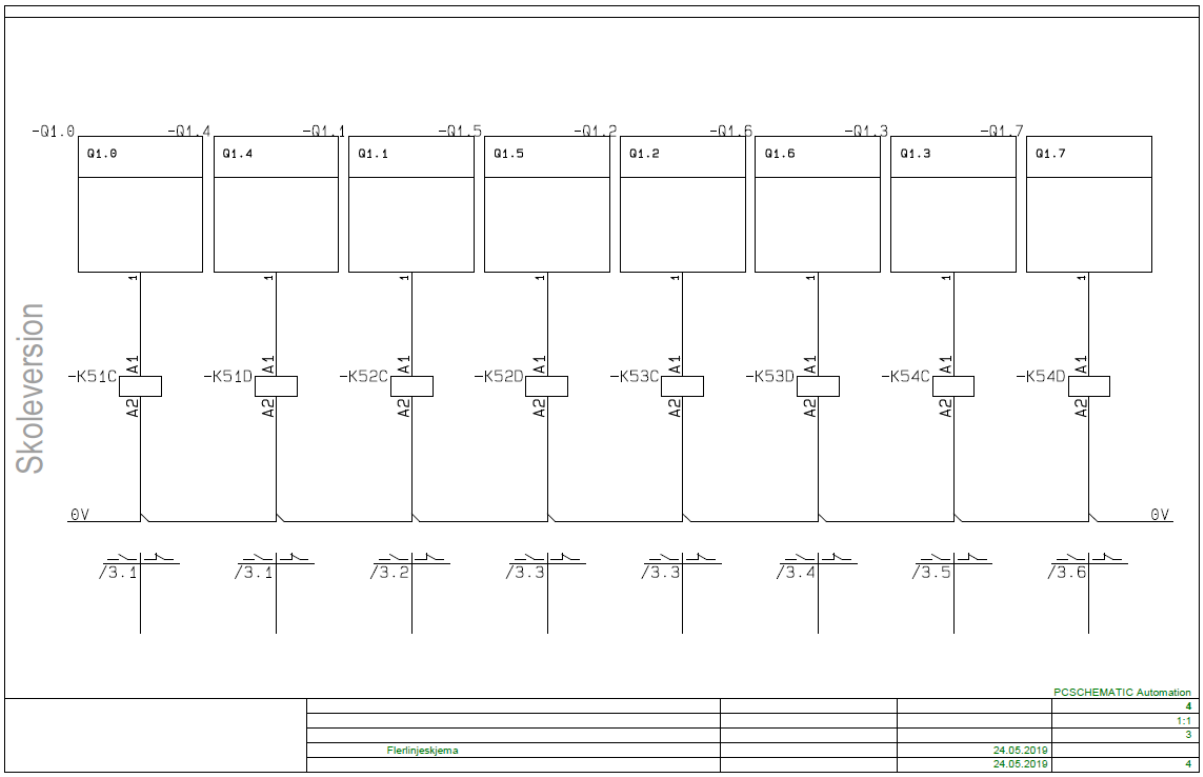
	Name	Data type	Address	Retain	Acces...	Writa...	Visibl...	Supervis...	Comment
1	HMI_styring	Bool	%I0.0		☑	☑	☑		Programmet kjores via HMI
2	HovedPLS_styring	Bool	%I0.1		☑	☑	☑		Programmet kjores via hoved PLS
3	Datafra_hovedPLS	Bool	%I0.2		☑	☑	☑		
4	Vifte1_venstre	Bool	%I0.3		☑	☑	☑		Inngangsvariabel, vifte 1 mot venstre
5	Vifte2_venstre	Bool	%I0.4		☑	☑	☑		Inngangsvariabel, vifte 2 mot venstre
6	Vifte3_venstre	Bool	%I0.5		☑	☑	☑		Inngangsvariabel, vifte 3 mot venstre
7	Vifte4_venstre	Bool	%I0.6		☑	☑	☑		Inngangsvariabel, vifte 4 mot venstre
8	Vifte1_hoyre	Bool	%I0.7		☑	☑	☑		Inngangsvariabel, vifte 1 mot hoyre
9	Vifte2_hoyre	Bool	%I1.0		☑	☑	☑		Inngangsvariabel, vifte 2 mot hoyre
10	Vifte3_hoyre	Bool	%I1.1		☑	☑	☑		Inngangsvariabel, vifte 3 mot hoyre
11	Vifte4_hoyre	Bool	%I1.2		☑	☑	☑		Inngangsvariabel, vifte 4 mot hoyre
12	Vifte1_venstre_running	Bool	%Q1.0		☑	☑	☑		Utgangsvariabel, vifte 1 kjorer mot venstre
13	Vifte2_venstre_running	Bool	%Q1.1		☑	☑	☑		Utgangsvariabel, vifte 2 kjorer mot venstre
14	Vifte3_venstre_running	Bool	%Q1.2		☑	☑	☑		Utgangsvariabel, vifte 3 kjorer mot venstre
15	Vifte4_venstre_running	Bool	%Q1.3		☑	☑	☑		Utgangsvariabel, vifte 4 kjorer mot venstre
16	Vifte1_hoyre_running	Bool	%Q1.4		☑	☑	☑		Utgangsvariabel, vifte 1 kjorer mot hoyre
17	Vifte2_hoyre_running	Bool	%Q1.5		☑	☑	☑		Utgangsvariabel, vifte 2 kjorer mot hoyre
18	Vifte3_hoyre_running	Bool	%Q1.6		☑	☑	☑		Utgangsvariabel, vifte 3 kjorer mot hoyre
19	Vifte4_hoyre_running	Bool	%Q1.7		☑	☑	☑		Utgangsvariabel, vifte 4 kjorer mot hoyre
20	Vifte1_knapp	Bool	%M2.0		☑	☑	☑		Knapp for vifte 1
21	Vifte2_knapp	Bool	%M2.1		☑	☑	☑		Knapp for vifte 2
22	Vifte3_knapp	Bool	%M2.2		☑	☑	☑		Knapp for vifte 3
23	Vifte4_knapp	Bool	%M2.3		☑	☑	☑		Knapp for vifte 4
24	Modus1	Bool	%M2.4		☑	☑	☑		Knapp for modus 1
25	Modus2	Bool	%M2.5		☑	☑	☑		Knapp for modus 2
26	Modus3	Bool	%M2.6		☑	☑	☑		Knapp for modus 3
27	Modus4	Bool	%M2.7		☑	☑	☑		Knapp for modus 4
28	Modus5	Bool	%M3.0		☑	☑	☑		Knapp for modus 5
29	Modus6	Bool	%M3.1		☑	☑	☑		Knapp for modus 6
30	Modus7	Bool	%M3.2		☑	☑	☑		Knapp for modus 7
31	Lysvifte1	Bool	%M3.3		☑	☑	☑		Statuslys
32	Lysvifte2	Bool	%M3.4		☑	☑	☑		Statuslys
33	Lysvifte3	Bool	%M3.5		☑	☑	☑		Statuslys
34	Lysvifte4	Bool	%M3.6		☑	☑	☑		Statuslys
35	Lysmodus1	Bool	%M3.7		☑	☑	☑		Statuslys
36	Lysmodus2	Bool	%M4.0		☑	☑	☑		Statuslys
37	Lysmodus3	Bool	%M4.1		☑	☑	☑		Statuslys
38	Lysmodus4	Bool	%M4.2		☑	☑	☑		Statuslys
39	Lysmodus5	Bool	%M4.3		☑	☑	☑		Statuslys
40	Lysmodus6	Bool	%M4.4		☑	☑	☑		Statuslys
41	Lysmodus7	Bool	%M4.5		☑	☑	☑		Statuslys

Flerlinjeskjema.






V-L1
V-L1
-K51C /4.1
-K51D /4.1
-K52C /4.2
-K52D /4.3
-K53C /4.3
-K53D /4.4
-K54C /4.5
-K54D /4.6
-K51B /.1
-K51A /.1
-K52B /.3
-K52A /.2
-K53B /.4
-K53A /.3
-K54B /.6
-K54A /.5
Skoleversion
-K51A
-K51B
-K52A
-K52B
-K53A
-K53B
-K54A
-K54B
A1
A2
V-L2
V-L2
/1.1 /.1 /2.1
/1.1 /.1 /2.1
/1.2 /.3 /2.2
/1.3 /.2 /2.3
/1.4 /.4 /2.4
/1.5 /.3 /2.4
/1.5 /.6 /2.5
/1.6 /.5 /2.6
PCSCHEMATIC Automation
3
1:1
2
Flerlinjeskjema
24.05.2019
4
24.05.2019
4


-Q1.0
-Q1.4
-Q1.1
-Q1.5
-Q1.2
-Q1.6
-Q1.3
-Q1.7
Q1.0
Q1.4
Q1.1
Q1.5
Q1.2
Q1.6
Q1.3
Q1.7
Skoleversion
-K51C
-K51D
-K52C
-K52D
-K53C
-K53D
-K54C
-K54D
A1
A2
0V
0V
/3.1
/3.1
/3.2
/3.3
/3.3
/3.4
/3.5
/3.6
PCSCHEMATIC Automation
4
1:1
3
Flerlinjeskjema
24.05.2019
24.05.2019
4

MTBF PLS komponenter

	A	B	C	D	E	F
1	***The following specified values of MTBF are statistic values which serve only for the estimation of maintenance and substitute needs. These values are no guaranteed product attribute.***					
2	**MLFB**	**Description**	**System**	**MTBF (year)**	**Temperature (°C)**	**Comment**
1248	6ES7505-0KA00-0AB0	DC 25W	SIMATIC S7-1500	28,00y		
1293	6ES7516-3FN01-0AB0	CPU 1516-3 PN/DP incl. Display	SIMATIC S7-1500	25,60y		
1306	6ES7521-1BL00-0AB0	DI 32x24VDC HF	SIMATIC S7-1500	90,53y		
1314	6ES7522-1BL00-0AB0	DQ 32x24VDC/0.5A ST	SIMATIC S7-1500	94,66y		
10	6AV2124-0GC01-0AX0	TP700 Comfort	SIMATIC HMI	7,70y		