jamk.fi

Sekvenssin visualisointi operaattorille Siemens TIA Portal -ympäristöön

Sami Saukko

Opinnäytetyö
Joulukuu 2017
Tekniikan ja liikenteen ala
Insinööri (AMK), automaatiotekniikan tutkinto-ohjelma

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Saukko, Sami	Opinnäytetyö, AMK	Joulukuu 2017
	Sivumäärä 43	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: x

Työn nimi

Sekvenssin visualisointi operaattorille Siemens TIA Portal -ympäristöön

Tutkinto-ohjelma

Automaatiotekniikan tutkinto-ohjelma

Työn ohjaaja(t)

Ari Kuisma, Veli-Matti Häkkinen

Toimeksiantaja(t)

JEEC Oy

Tiivistelmä

Logiikkaohjelmointi on työläs vaihe automaatioalan projekteissa. Ohjelmointia voidaan kuitenkin nopeuttaa valmiiksi luotujen logiikkaohjelmointipohjien avulla. Näitä ohjelmointipohjia voivat olla esimerkiksi erilaisten laitteiden, kuten moottorien ja venttiilien, ohjaukset. Ohjelmointipohjia on hyvä päivittää ja tehdä myös uusia koko ajan kehittyvällä alalla.

Työn tavoitteena oli luoda logiikkaohjelmointipohja, jota pystyttäisiin hyödyntämään sekvenssien visualisoinnissa ja ohjauksessa. Ohjelmointipohja helpottaa itse logiikkaohjelmointiosuutta sekä laitteiden ja prosessien käyttöönottoa. Pohjasta tulee olemaan hyötyä myös laitteistojen ja prosessien käyttäjille. Operaattori pystyy helposti näkemään, mitä prosessissa tapahtuu ja vikatilanteiden ilmetessä ongelma on helpompi paikantaa. Ohjelmointipohjat auttavat vähentämään myös mahdollisten virheiden määrää ohjelmakoodissa. Opinnäytetyö on jatkoa toimeksiantajan aikaisemmalle kehitystyölle.

Työn tuloksena syntyi kaksi logiikkaohjelmointipohjaa, joita voidaan hyödyntää sekvenssien kanssa. Näille pohjille tehtiin myös erilliset ohjeistukset, jotka on esitetty raportin liitteissä. Ensimmäistä pohjaa pystytään hyödyntämään sekvenssien ohjauksessa ja toista taas sekvenssien visualisoinnissa. Pohjien ulkoasua ja toimintoja pystytään muokkaamaan, jolloin pohjien toiminnot ja ulkoasu voidaan räätälöidä juuri kulloiseenkin projektiin sopivaksi. Tämä tarkoittaa myös sitä, että tulevaisuudessa näitä pohjia voidaan päivittää ja kehittää, jos ja kun uusia toimintoja halutaan ottaa käyttöön.

Avainsanat (asiasanat)

TIA Portal -ohjelmointityökalu, ohjelmoitava logiikka, ohjelmointi

Liitteenä kaksi kappaletta ohjeistuksia, 16 sivua. Liitteet 1 ja 2 ovat salassa pidettäviä, joten ne on poistettu julkisesta työstä. Salassapidon peruste Julkisuuslain 621/1999 24§, kohta 17, yrityksen liike- tai ammattisalaisuus. Salassapitoaika viisi (5) vuotta, salassapito päättyy 4.12.2022.

jamk.fi

Description

Author(s)	Type of publication	Date
Saukko, Sami	Bachelor's thesis	December 2017
		Language of publication: Finnish
	Number of pages: 43	Permission for web publication: x

Title of publication

Sequence visualization to operator in Siemens TIA Portal

Degree programme

Electrical and Automation Engineering

Supervisor(s)

Kuisma Ari, Häkkinen Veli-Matti

Assigned by

JEEC Oy

Abstract

PLC programming takes a great deal of time in automation industry projects. This can be made easier and faster with faceplates that control, for example, motors, valves, and other devices. It is very useful to upgrade these faceplates and make new faceplates in constantly growing automation industry. The thesis was assigned by JEEC Oy. JEEC offers engineering and consultant services in the fields of automation and electrification.

The goal of the thesis was to make a faceplate that can be used to control and visualize sequences and to make the programming part easier and faster. This will also help the process operator. The operator can easily see what is happening in the process and if an error occurs, it is easier to locate. Faceplates help to minimize possible flaws in the code.

The thesis resulted in two faceplates: one that can be used to control sequences and the other to visualize sequences. Both faceplates also contain manuals which are added at the end of this report. The layout and the functionality of the faceplates can be customized by the user to suit the particular project, which also means that in the future these faceplates can be updated, and more functions can be added to them.

Keywords/tags (subjects)

TIA-Portal software, programmable logic controller, programming

In appendixes there are two manuals, 16 pages. Appendixes 1 and 2 are confidential which have been removed from the public thesis. Grounds for secrecy: Act on the Openness of Government Activities 621/1999, Section 24, 17: business or professional secret. Period of secrecy is five years and it ends 4.12.2022.

Sisältö

Kuviot

Sanasto

Aliohjelma	Aliohjelma sisältää pieniä osia ohjelmakoodista. Aliohjelmia käytetään pääohjelmassa.
CPU	Central prosessing unit eli keskusyksikkö
Ethernet	Lähiverkkotekniikka
HMI	Human machine interface, rajapinta operaattorin ja prosessin välillä. Esim. näyttöpaneeli
HW-konfiguraatio	HW-konfiguraation avulla määritellään käytettävä laitteisto.
Lähtökortti	CPU:hun liitettävä kortti, jonka avulla voidaan ohjata laitteita.
Ohjelmalohko	Osio, joka sisältää ohjelmakoodia
Ohjelmoitava-logiikka	Pieni tietokone, jonka avulla voidaan ohjata laitteita ja prosesseja.
Pääohjelma	Ohjelma, jota CPU suorittaa kerta toisensa jälkeen.
Panosprosessi	Panosprosessissa tuotetaan määritellyistä raaka-aineista tietyn reseptin mukaisesti äärellinen määrä tuotteita.
Profibus	Process Field Bus, avoin kenttäväyläjärjestelmä
Profinet	Process Field Net, teollisuus-ethernet-standardi
Sekvenssi	Ohjaustapa, jolla ohjataan prosessia askelmaisesti.
Step 7	Siemensin logiikoiden ohjelmointiin käytettävä ohjelmisto.
Tag	Muuttuja Siemensin ohjelmistoissa
TIA Portal	Siemensin logiikoiden ohjelmointiin käytettävä ohjelmisto.
Tulokortti	CPU:hun liitettävä kortti, jonka avulla voidaan lukea tietoja.
WinCC	Valvomosuunnitteluohjelmisto

1 Johdanto

1.1 Opinnäytetyön tausta

Työelämässä on hyvä optimoida eri työvaiheet, jotta työskentely olisi nopeampaa ja virheiden määrät vähenisivät. Opinnäytetyössä tutustuttiin automaatioalan logiikkaohjelmointiin. Tarkemmin sanottuna sekvenssien ohjaukseen ja visualisointiin. Tätä työvaihetta on tarkasteltu ja pyritty tekemään työvaihe mahdollisimman helpoksi ohjelmoinnin kannalta.

Usein projekteissa tietyt osa-alueet pysyvät toiminnoiltaan samanlaisina. Tällaisille osa-alueille kannattaa siis tehdä ohjelmointipohjat valmiiksi, joita pystytään hyödyntämään ohjelmointiosuutta tehdessä. Esimerkkinä voidaan käyttää värityskirjaa. Kuvien ääriviivat kuvaavat tätä valmista pohjaa. Pohjan käyttäjän tulee lisätä värit eli projektissa käytettävät tiedot näiden ääriviivojen sisälle.

Toimeksiantaja JEEC Oy on kehittänyt logiikkaohjelmointiaan vuosien ajan. Yrityksessä on tehty uusia pohjia ja päivitetty myös vanhoja, jotta ohjelmointiosuus nopeutuisi ja virheiden määrää saataisiin vähennettyä.

Yrityksessä on jo jonkin aikaa keskusteltu sekvenssien visualisointi- ja ohjauspohjan tekemisestä. Nykyisellä menetelmällä laitteen käyttäjälle ongelmaksi on muodostunut vikatilanteiden analysointi. Operaattori ei välttämättä tiedä tarkalleen, miksi prosessi tai laitteisto on mennyt vikatilaan. Vikatilanteiden korjaaminen saattaa viedä todella paljon aikaa eikä operaattori välttämättä pysty edes tietämään, mistä ongelma johtuu. Yrityksellä oli siis tarve saada käyttöön sekvenssien visualisointi- ja ohjauspohja, joka korjaisi tämän ongelman.

1.2 Opinnäytetyön tehtävä ja tavoitteet

Tässä opinnäytetyössä tehtävänä oli luoda Siemensin TIA Portal -ympäristöön ohjelmointipohja, jota pystyttäisiin hyödyntämään sekvenssien visualisoinnissa ja ohjauksessa. Ohjelmointipohjan avulla oli tarkoitus nopeuttaa ja helpottaa logiikkaohjelmointia. Luotua pohjaa oli tarkoitus päästä hyödyntämään useiden tulevien projek-

tien parissa. Tämä säästää runsaasti aikaa ja vähentää mahdollisia virheitä ohjelmakoodissa. Tällaisten pohjien luominen onkin siksi todella kannattavaa ja työ tulee olemaan jatkoa toimeksiantajan aikaisemmalle kehitystyölle.

Tiivistettynä aiheeni oli tutkia, miten olisi kannattavinta luoda visualisointi- ja sekvenssinohjauspohja logiikan ja näytön välille. Tehtävänäni oli tutkia sekvenssien toimintaa ja selvittää, mitä olisi hyvä saada visualisoitua operaattorille, jotta sekvenssien operointi olisi mahdollisimman selkeää ja vikatilanteet saataisiin korjattua nopeasti. Tarvittavien ominaisuuksien selvitys tapahtui tutkimalla sekvenssejä ja TIA Portalin ominaisuuksia.

Toimeksiantajan tavoitteena oli saada toimiva pohja sekvenssien visualisoinnille ja ohjaukselle. Sekvenssin visualisointipohjan tavoitteena oli sekvenssin rakenteen visualisointi erilliselle näyttöpaneelille tai tietokoneelle. Näytöltä pitäisi pystyä näkemään, missä askeleessa sekvenssi on menossa ja mitkä ovat askeleiden siirtoehdot. Vikatilanteiden paikannuksen tuli olla myös mahdollista. Paikannuksen avulla voitaisiin selvittää, mikä askel oli aiheuttanut vikatilanteen. Sekvenssin operointipohjaan oli tavoitteena saada sekvenssin perusohjaukset, joita ovat esimerkiksi käynnistys, pysäytys ja hälytyksien kuittaus.

Nämä pohjat tulisivat helpottamaan laitteiston käyttöönottoa ja käyttöä. Tavoitteena oli saada pohjasta mahdollisimman selkeä ja helppokäyttöinen. Kun pohja pidetään mahdollisimman selkeänä, on sitä mahdollista muokata jatkossa, mikäli sille on tarvetta. Tekemäni visualisointipohjan avulla operaattori pystyy näkemään, miksi prosessi on jumiutunut tai mennyt vikatilaan. Tämä siis nopeuttaa vikatilanteiden korjaamista ja helpottaa huomattavasti laitteen tai prosessin operointia

Omana tavoitteenani oli päästä tutustumaan tarkemmin logiikkaohjelmointiin ja sekvenssien tekoon. Opinnäytetyötä tehdessä logiikkaohjelmointi tuli tutummaksi, josta on minulle paljon hyötyä tulevaisuudessa.

1.3 JEEC Oy

JEEC Oy on vuonna 2009 perustettu Jyväskylässä sijaitseva yritys. Yritys tarjoaa korkealaatuisia suunnittelu- ja konsultointipalveluja automaatio- ja sähkötoimialoilla. Automaatio on yrityksen ydinosaamisaluetta, sen lisäksi JEEC Oy tekee sähkö- ja instrumentointisuunnittelua. Automaation palveluihin kuuluvat perus-, sovellus- ja instrumentointisuunnittelu sekä näiden lisäksi asennusvalvonta, operointikoulutus, konsultointi sekä testaus ja käyttöönotto. Yritys työllistää tällä hetkellä 16 henkilöä. (JEEC Oy, 2017.)

Kehitystyö tehtiin toimeksiantajan tarjoamilla laitteistoilla ja ohjelmilla. Testaus tapahtui ensisijaisesti simuloimalla prosesseja. Tulevaisuudessa kun pohjia päästään hyödyntämään fyysisten laitteiden kanssa, päästään tekemään mahdolliset viimeiset korjaukset. Simuloimalla prosesseja päästiin kuitenkin jo todella pitkälle.

2 Opinnäytetyössä käytettävä laitteisto

2.1 Ohjelmoitava logiikka

Ohjelmoitava logiikka on pieni tietokone, jota käytetään laitteiden ja prosessien ohjauksessa (Keinänen, Kärkkäinen, Lähetkangas & Sumujärvi 2007, 212). Logiikat ovat rakenteeltaan tehty kestäviksi, jotta niitä voidaan käyttää haastavissa teollisuuden olosuhteissa, joissa niiden on tarkoitus kestää useiden vuosien ajan (Wright n.d).

Normaaliin tietokoneeseen verrattuna logiikka on hieman yksinkertaisempi. Logiikoiden valmistajia on suuri määrä ja jokaiselta löytyy useita eri malleja. Tässä työssä on kerrottu tarkemmin Siemensin ohjelmoitavista logiikoista, erityisesti 1500-sarjan logiikoista.

Logiikat koostuvat yleensä kolmesta pääosasta, jotka ovat virtalähde, CPU ja I/O-kortit. Virtalähteet voivat olla erillisiä tai integroituna CPU-yksikköön. Virtalähdettä valittaessa tulee tietää siihen kytkettävien laitteiden määrä, jotta osataan valita tarpeeksi tehokas virtalähde. Liian pientä virtalähdettä käytettäessä ei välttämättä pystytä syöttämään tarvittavaa käyttöjännitettä kaikille siihen liitetyille laitteille. Jokaiselle

laitteistokokonaisuudelle tulee siis valita juuri oikeanlainen virtalähde (How PLCs Work n.d). Alla olevassa kuviossa 1 on esitelty Siemensin virtalähde. Kannen alta löytyvät tarvittavat liittimet sähköjohtojen liitäntään.

Kuvio 1. Siemens PS 60W 120/230 V AC/DC -virtalähde (6ES7507-0RA00-0AB0 n.d.)

2.2 I/O-kortit

I/O-kortteja on pääasiassa kahta eri tyyppiä, digitaalisia ja analogisia. Lyhenteet I ja O tulevat sanoista input ja output, jotka taas suomeksi ovat tulo ja lähtö. Molemmissa korttityypeissä on siis erikseen tulo- ja lähtökortit. Kaikilla näillä korttityypeillä on useita eri malleja, joissa vaihtelee esimerkiksi kanavien määrä. Jokaiseen kanavaan voidaan lukea yhtä tietoa. Mitä enemmän kanavia on, sitä enemmän tietoa voidaan yhdellä kortilla lukea. Korttien ulkonäkö muuttuu aina mallia vaihdettaessa, mutta erityyppiset kortit, kuten digitaaliset ja analogiset kortit, näyttävät pääasiassa samanlaisilta (Katso kuvio 2).

Digitaalisilla tulokorteilla pystytään lukemaan päällä/pois-tietoja. Nämä voivat olla esimerkiksi kytkimen tietoja tai tieto siitä, onko jokin moottori päällä vai ei. Toiminta perustuu yleisemmin jännite- tai virtaviestiin. Esimerkkinä voidaan käyttää moottorin

tilatiedon lukua. Kun moottori on käynnissä, se antaa 24 voltin jännitteen digitaalitulokorttiin, jolloin kortissa oleva kanava huomaa, että siihen liitetty laite on kytketty päälle. Kun laite ei ole päällä, se ei anna tätä 24 voltin jännitettä kyseiseen kanavaan, jolloin kortin tulosta voidaan lukea, ettei laite ole päällä. Kuviossa 2 on digitaalinen tulokortti. Kortissa on 16 kanavaa, joihin liitetään laitteet kaapelien avulla. Liitännät löytyvät kortin edestä olevan kannen alta.

Kuvio 2. Digitaalinen tulokortti (Digital modules n.d.)

Digitaalisella lähtökortilla ohjataan siinä olevia kanavia päälle ja pois päältä. Kortin avulla voidaan esimerkiksi käynnistää jokin laite. Kun kanava on päällä, laite saa viestin, että sen tulee käynnistyä. Kun kanava on pois päältä, laite pysähtyy. Tässäkin kortissa käytetään samantyyppisiä jännite- ja virtaviestejä.

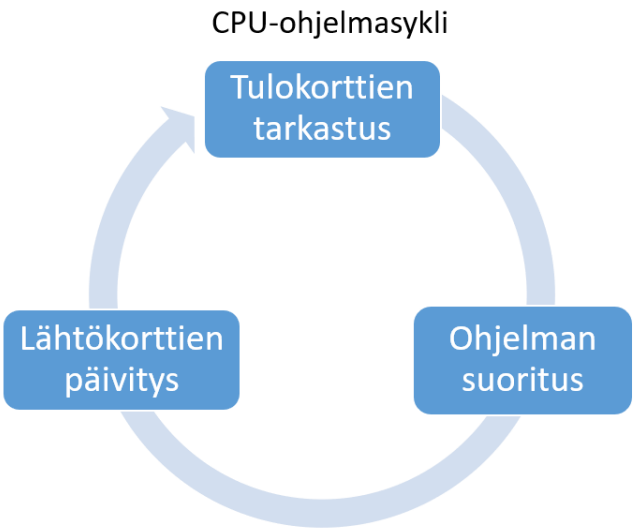
Analogisilla tulokortteilla voidaan lukea esimerkiksi lämpötilalähettimen lähettämää arvoa. Viestit tulevat joko jännite- tai milliampeeriviestinä. Lämpötilalähetin voi esimerkiksi mitata 0-100 asteen lämpötilaa, jolloin virtaviesti 4-20 mA voidaan skaalata tälle lämpötila-alueelle. Virtaviestin ollessa 20 mA on lämpötila mittalaitteella 100 astetta, kun taas vastaavasti 4 mA:ssa lämpötila on nolla astetta. Analogisia tulokort-

teja voidaan siis käyttää erilaisten mittauksien kanssa. Virtaviestin minimiarvo on tarkoituksella 4 mA, koska jos arvo tippuu 0 mA:iin, tiedetään, että laite on rikki tai johto katkennut.

Analogisten lähtökorttien avulla voidaan puolestaan lähettää jännite- tai virtaviestejä, joilla voidaan ohjata esimerkiksi säätöventtiilin asentoa. Toiminta on sama kuin analogisissa tulokorteissa mutta vain käänteinen. Kun lähetetään 20 mA:n arvo säätöventtiilille, se aukeaa kokonaan ja taas 4 mA:n arvolla se menee kiinni. Tässäkin minimiarvo pidetään 4 mA:ssa, jotta voidaan välttyä esimerkiksi mahdollisilta häiriöiltä.

2.3 Toiminta

CPU on koko laitteiston aivot. CPU:hun tehdään ohjelma erillisellä tietokoneella (How PLCs Work n.d). Esimerkiksi Siemensin logiikoille ohjelmia voi tehdä TIA Portalin avulla. Valmis ohjelma taas ladataan CPU:lle, minkä jälkeen CPU on valmis käytettäväksi. CPU suorittaa tätä samaa ohjelmaa kerta toisensa jälkeen läpi. Ohjelmaan voidaan tehdä erilaisia toimintoja, jotka riippuvat tulo- ja lähtökorttien tiloista. Kuviossa 3 voidaan nähdä CPU-ohjelmasyklin rakenne. Lyhyesti sanottuna CPU lukee tulokorttien tiedot, joiden avulla tehdään halutut toiminnot ohjelmassa. Tämän jälkeen päivitetään lähtökorttien tilat, minkä jälkeen ohjelma alkaa alusta (Katso kuvio 3).

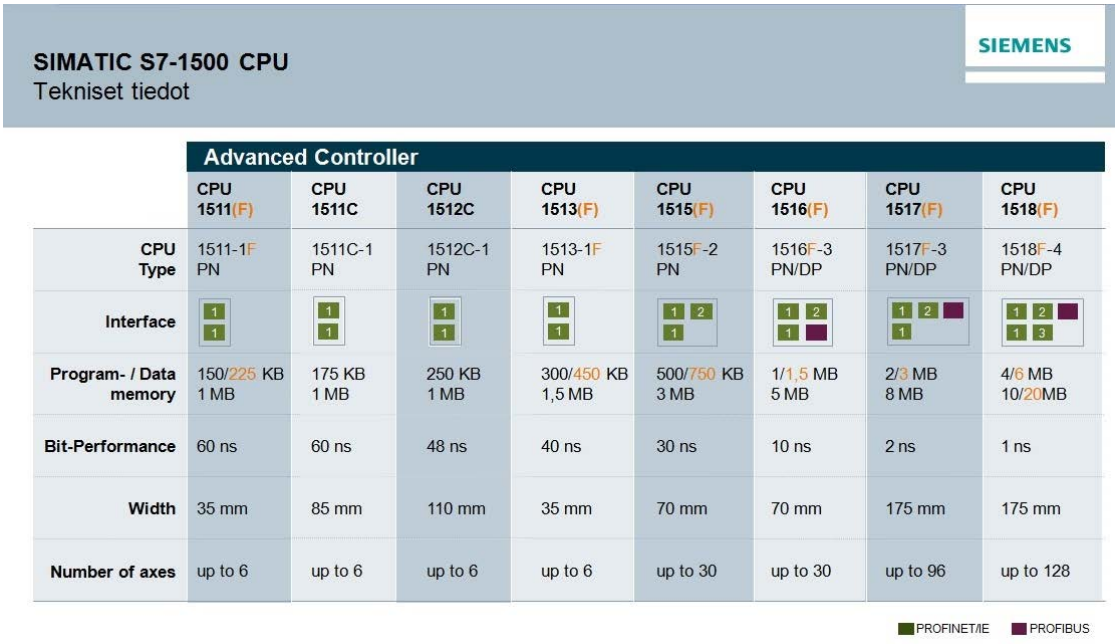


Kuvio 3. CPU-ohjelmasykli

Ohjelma koostuu erilaisista ohjelmalohkoista. Ohjelmassa on niin sanottu pääohjelma, jota CPU suorittaa kerta toisensa jälkeen (How PLCs Work n.d). Pääohjelmalohkoon voidaan lisätä aliohjelmia, jotka voivat sisältää esimerkiksi eri laitteiden ohjauksia. Näiden ohjelmalohkojen sisälle kirjoitetaan itse ohjelmakoodi, ja valmiita ohjelmalohkoja voidaan lisätä toisten ohjelmalohkojen sisälle. Jotta nämä ohjelmalohkot toimisivat, ne on lisättävä pääohjelman tai jonkin muun lohkon sisälle, joka on taas lisätty pääohjelmaan.

2.4 Siemens 1500 -ohjelmoitava logiikka

Tässä työssä tutustutaan tarkemmin Siemensin 1500-sarjan CPU:hun. Kyseinen sarja julkaistiin vuonna 2013, ja laitteet soveltuvat kaikista vaativimpien prosessien ohjaukseen (S7-1500 n.d). Sarjan CPU:sta löytyy enemmän ominaisuuksia, joita voitiin hyödyntää tässä työssä. Yksi erittäin tärkeä ominaisuus oli Graph-ohjelmointikielen käyttäminen. Tätä ominaisuutta ei löydy esimerkiksi 1200-sarjan CPU:sta. Kuviossa 4 on muutamia perustietoja 1500-sarjan laitteista ja kuinka ne eroavat toisistaan.



Kuvio 4. Simatic S7-1500 CPU:n tekniset tiedot

Jokaisessa laitteessa on vakiona Profinet-liitännät ja muutamasta mallista löytyy myös Profibus-liitännät. Profinet ja Profibus ovat teollisuudessa käytettäviä tiedonsiirtomenetelmiä. Teollisuudessa on päädytty käyttämään näitä omia menetelmiä, koska perinteisten menetelmien, esimerkiksi Ethernetin, nopeus ei riitä. Ethernetissa tietopaketin perille saanti saattaa kestää sekunteja, kun taas automaatiossa aikavaade on millisekunteja. Profinetin avulla voidaan liittää esimerkiksi näyttöpaneeli CPU:lle. Profibus on taas luotu hajautettujen kenttälaitteiden liittämiseen. Näiden tiedonsiirtomenetelmien avulla päästään todella nopeisiin vasteaikoihin. Myös langaton tiedonsiirto on mahdollista käytettäessä Profinettiä, koska se pohjautuu Ethernet-protokollaan. (Teollinen tiedonsiirto n.d.)

Tuoteperheessä on myös kompaktimalleja, joissa on integroituna tulo- ja lähtökortteja. Jokaisesta standardi-CPU-mallista löytyy myös tätä vastaava turva-CPU-malli, jossa on yhdistetty turva- ja vakiotoiminnot samaan logiikkaan. (S7-1500 n.d.) Mallien välillä myös laitteiden koko ja ulkonäkö muuttuvat. Kuviossa 5 voidaan nähdä 1515-2 PN -ohjelmoitava logiikka. Laitteessa on pieni näyttö ja muutama näppäin, joilla laitetta pystytään operoimaan. Laitteen näytön alta löytyvät liittimet, joiden avulla logiikka voidaan liittää esimerkiksi tietokoneeseen.

Kuvio 5. Siemens CPU 1515-2 PN (6ES7515-2AM01-0AB0 n.d.)

Ohjelmoitavia logiikoita voidaan ohjelmoida Siemensin TIA Portal -ohjelmiston avulla. Tästä ohjelmistosta ja sen toiminnoista kerrotaan kappaleessa 3 TIA Portal -ohjelmisto.

2.5 Näyttöpaneelit

Logiikoihin liitetään yleensä joko tietokone tai erillinen näyttöpaneeli. Niiden avulla laitteita ja prosesseja pystytään operoimaan. Näyttöjen avulla voidaan myös nähdä, mitä prosessissa tapahtuu. Näyttöpaneelit vievät vähemmän tilaa ja niiden avulla pystytään helposti operoimaan prosesseja. Näytöissä on myös paremmat suojaukset esimerkiksi pölyltä ja roiskeilta, jolloin ne sopivat paremmin likaisiin kohteisiin.

Siemensillä on muutamia eri näyttöpaneelisarjoja, jotka on jaoteltu toiminnallisuuksien mukaan. Näyttöjä on pääasiassa kahta eri tyyppiä: kosketusnäytöt tai painikepohjainen operointi. Kosketusnäytölliset näyttöpaneelit ovat pienempiä verrattuna painikepohjaisiin paneeleihin. (Kosketusnäytöt n.d.)

Tässä työssä tutustuttiin tarkemmin Siemensin Comfort-paneeleihin. Comfort-sarjan paneeleista löytyy useita eri malleja. Toiminnot näiden mallien välillä ovat samat, vain paneelien koko vaihtuu. Näyttöjä on aina neljän tuuman näytöstä 22 tuuman näyttöön. Comfort-paneeleista löytyy eniten eri toiminnallisuuksia verrattuna muihin sarjoihin. Tämä mahdollistaa sen, että kaikki parhaat ominaisuudet saadaan käyttöön.

3 TIA Portal -ohjelmisto

TIA Portal on ohjelmointityökalu. Lyhenne TIA tulee englannin kielen sanoista Totally Integrated Automation. Ohjelmaan on yhdistetty logiikoiden ohjelmointi, näyttöpaneelien ohjelmointi ja käyttöliittymän suunnittelu. Sen avulla voidaan ohjelmoida myös taajuusmuuttajia ja turvaratkaisuja. (TIA Portal -ohjelmointityökalu n.d.)

3.1 Käyttöliittymä

TIA Portalin avulla voidaan suunnitella käyttöliittymiä. Näiden käyttöliittymien avulla pystytään operoimaan prosesseja. Käyttöliittymien pitää olla ulkoasultaan selkeitä, jotta operaattorin on helppo hahmottaa mitä prosessissa tapahtuu (Myers, R. 2017). Hyvin suunnitellussa käyttöliittymässä saattaa olla todella paljon tietoa, mutta ne ovat helppolukuisia. Joissain käyttöliittymissä saattaa olla paljon vähemmän tietoa, mutta ne on suunniteltu sekavan näköisiksi, joten niitä on vaikeampi lukea. Selkeys tulee käytännössä siitä, miten tiedot sijoitetaan näytölle. (Valvomo : suunnittelun periaatteet ja käytännöt 2011, luku 7.2.)

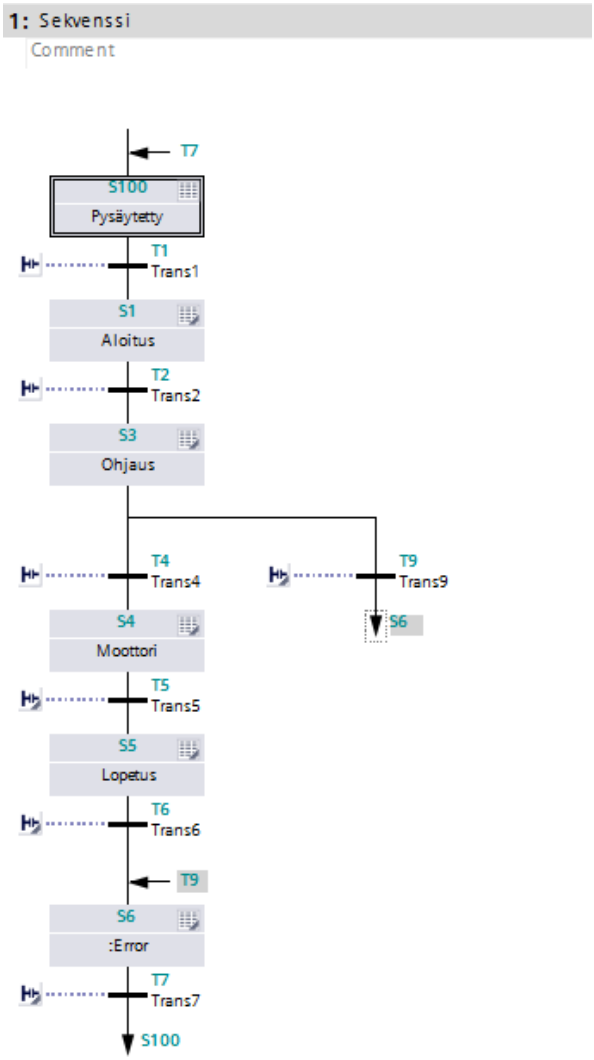
Yleensä henkilön katsoessa käyttöliittymää katse käy kuvaa läpi tietyssä järjestyksessä ja samalla aivot yrittävät löytää sieltä merkityksellisiä kokonaisuuksia. Tämä järjestys on sama kuin tekstin lukusuunta eli ylhäältä alas ja vasemmalta oikealle. Käyttöliittymä olisi siis hyvä jakaa osiin. Usein katse löytää ensimmäiseksi kuvioiden ylälaidat ja vasemmat sivut. Tätä tietoa olisi hyvä hyödyntää esimerkiksi tasaamalla nämä sivut. Tämä helpottaa käyttöliittymän lukemista. (Valvomo : suunnittelun periaatteet ja käytännöt 2011, luku 7.2.)

Käyttöliittymän suunnittelussa värit pidetään yleensä mahdollisimman hillittyinä. Tällöin esimerkiksi hälytyksissä voidaan hyödyntää kirkkaita värejä. Jos käyttöliittymässä yritetään korostaa liian monia asioita kirkkaiden värien avulla, voidaan törmätä ongelmaan, jossa liian monen asian korostaminen aiheuttaa sen, että mikään ei lopulta korostu. (Valvomo : suunnittelun periaatteet ja käytännöt 2011, luku 7.2.)

3.2 Sekvenssit

Sekvenssiohjaus on prosessien ja laitteiden ohjaustapa. Sekvenssissä ohjaustoimenpiteet etenevät askelmaisesti määriteltyjen siirtoehtojen mukaisesti. Sekvenssit sisältävät halutun määrän askelia, jotka sisältävät ohjaustoiminnot. Askeleesta toiseen siirrytään määriteltyjen siirtoehtojen täyttyessä. (Kippo & Tikka 2008, luku 4.5.)

TIA Portalilla voidaan tehdä näitä sekvenssejä. Sekvenssin toimintaa havainnollistetaan kuvan ja selostuksen avulla seuraavassa kappaleessa. Kuviossa 6 näkyy esimerkki sekvenssistä.



Kuvio 6. Esimerkki sekvenssistä

Sekvenssissä on tietty määrä askelia, joiden sisälle voidaan lisätä toimintoja. Jokaisen askeleen jälkeen tulee siirtoehto tai siirtoehdot. Siirtoehtoina tarkastellaan esimerkiksi haluttuja mittauksia, joiden perusteella osataan siirtyä eteenpäin oikeaan aikaan. Kuviossa 6 voisi esimerkiksi olla seuraavanlaisia toimintoja: Ensimmäinen askel, pysäytetty, ei sisältäisi mitään toimintoja. Kun sekvenssille on annettu käyntilupa, se olisi tässä askeleessa ja seuraavaan askeleeseen siirtoehtona voisi olla esimerkiksi start-näppäimen painaminen. Aloitus-askeleessa taas asetettaisiin laitteet alkuasentoihin ja tässä voisi olla siirtoehtona esimerkiksi näiden laitteiden asentojen tarkistus. Seuraavaan askeleeseen siirrytään vasta sitten, kun siirtoehdot ovat täyttyneet. As-

keleisiin pystytään myös lisäämään valvonta-aika. Tämän ajan avulla pystytään kontrolloimaan aikaa, kuinka kauan askeleella on aikaa suorittaa toiminnot ja siirtyä seuraavaan askeleeseen.

Jos siirtoehdot eivät täyty halutussa ajassa, siirtyy askel vikatilaan, josta pääsee eteenpäin vian kuittaamalla. Ohjaus-askeleessa voitaisiin asettaa esimerkiksi säätöventtiili auki. Tämän askeleen jälkeen onkin kaksi siirtoehtoa. Toisessa voitaisiin tarkastella, avautuuko venttiili haluttuun asentoon, minkä jälkeen siirryttäisiin seuraavaan askeleeseen. Toisessa siirtoehdossa voisi olla esimerkiksi venttiilin asennon vertailu. Jos venttiili ei avautuisi haluttuun asentoon, siirtyisi sekvenssi tässä tilanteessa askeleeseen kuusi, joka olisi vika-askel. Tämä askel voisi ilmoittaa käyttäjälle viasta, jotta se osattaisiin korjata. Moottori-askeleessa voitaisiin käynnistää moottori ja siirtoehtona voisi olla jonkin säiliön pinnankorkeuden mittaus. Kun säiliön pinta on halutulla korkeudella, siirryttäisiin lopetus-askeleeseen. Tässä voitaisiin taas pysäyttää pumppu, minkä jälkeen siirryttäisiin takaisin ensimmäiseen askeleeseen.

Sekvenssien toiminta on siis yksinkertaista, mutta niiden avulla pystytään ohjaamaan monimutkaisiakin prosesseja. Teollisuudessa sekvenssejä käytetään usein panosprosessien ohjauksessa. Näitä panosprosesseja voivat olla esimerkiksi maalin, paperin tai elintarvikkeen valmistus. Raaka-aineita syötetään panosprosessiin tietty määrä oikeassa suhteessa ja vaiheissa. Prosessissa käsiteltäville aineille tehdään erilaisia toimenpiteitä, esimerkiksi lämmitys, kuivaus ja sekoitus. (Kippo & Tikka 2008, luku 4.5) Panosprosessien ohjaus tapahtuu sekvenssien avulla, jolloin toimenpiteet tapahtuvat määrätyssä järjestyksessä.

Sekvenssejä voidaan tehdä myös useampia, jolloin prosessin tai laitteen ohjaus voidaan jakaa osiin. Prosessissa voi olla niin sanottu pääsekvenssi, joka ohjaa alasekvenssejä. Sekvenssejä käytetään myös jatkuvien prosessien ylösajoon. Sekvenssien avulla saadaan siis käynnistettyä prosessi, minkä jälkeen prosessi jätetään säätöjen ajettavaksi.

4 Ohjelmointipohjan luominen

4.1 Vanhat projektit

Aloitin työn tutustumalla Siemens TIA Portal -ohjelmistoon. Minulla ei ollut aikaisempaa kokemusta tämän ohjelmiston käytöstä. Opiskelujen aikana olin kuitenkin käyttänyt Step 7 ja WinCC -ohjelmia, joissa perustoiminnot ovat samat. Opinkin nopeasti ohjelman käytön ja aloin tutustumaan vanhoihin projekteihin, joissa oli käytetty sekvenssejä.

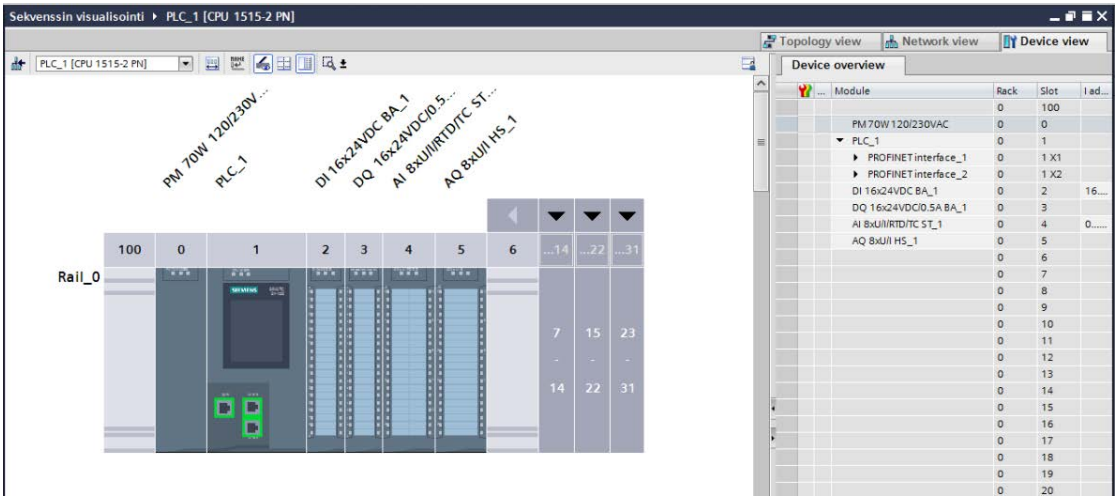
Vanhoja projekteja tutkimalla sain selville, millä tavalla tiedot liikkuvat näytön ja eri ohjelmalohkojen välillä. Tämä oli todella tärkeä tiedostaa, että pystyin etenemään työssä oikealla tavalla. Käytinkin vanhojen projektien tutkimiseen runsaasti aikaa, jotta sain selville, miten kaikki sekvensseihin liittyvä data liikkuu ohjelmassa.

4.2 HW-konfiguraatio

Kun olin perehtynyt tarkasti aikaisempiin projekteihin ja opetellut ohjelman käyttöä, olin valmis tekemään itse työtä. Aloitin tekemällä tyhjän projektin Siemens TIA Portal -ohjelmaan. Ensimmäisenä tehdään niin sanottu HW-konfiguraatio. Tämä tarkoittaa sitä, että valitaan käytettävät laitteet ja lisätään ne ohjelmaan. Tässä tapauksessa ei ollut väliä, minkälaiset laitteet projektiin valittiin, koska tätä projektia ei tultaisi käyttämään kokonaisuudessaan pohjana muille projekteille, vaan vain sen ohjelmakoodia.

Ensimmäisenä valitsin käytettävän CPU:n, joka oli malliltaan 1515-2 PN. Tämän valitsin siksi, koska 1500 sarjan CPU:lla voidaan käyttää ohjelmasta löytyvää Graph-ohjelmointikieltä. Tätä ohjelmointikieltä voidaan simuloida ohjelmasta löytyvien toimintojen avulla. Käytettävän 1500 sarjan CPU:t ovat hyvin yleisiä, joten juuri kyseinen malli sopii tehtävään projektiin hyvin. Seuraavaksi lisäsin virtalähteen. Virtalähteeksi valitsin 60-wattisen 230 voltin jännitettä käyttävän Siemensin virtalähteen. Näiden lisäksi tarvitsin vielä neljä erilaista korttimoduulia. Ensimmäiseksi lisäsin digitaalisen tulokortin ja tämän viereen digitaalisen lähtökortin.

Kolmas ja neljäs kortti olivat taas analogiset tulo- ja lähtökortit. Laitteiden tarkemmat mallit näkyvät kuviossa 7.



Kuvio 7. Projektissa käytetty laitteisto

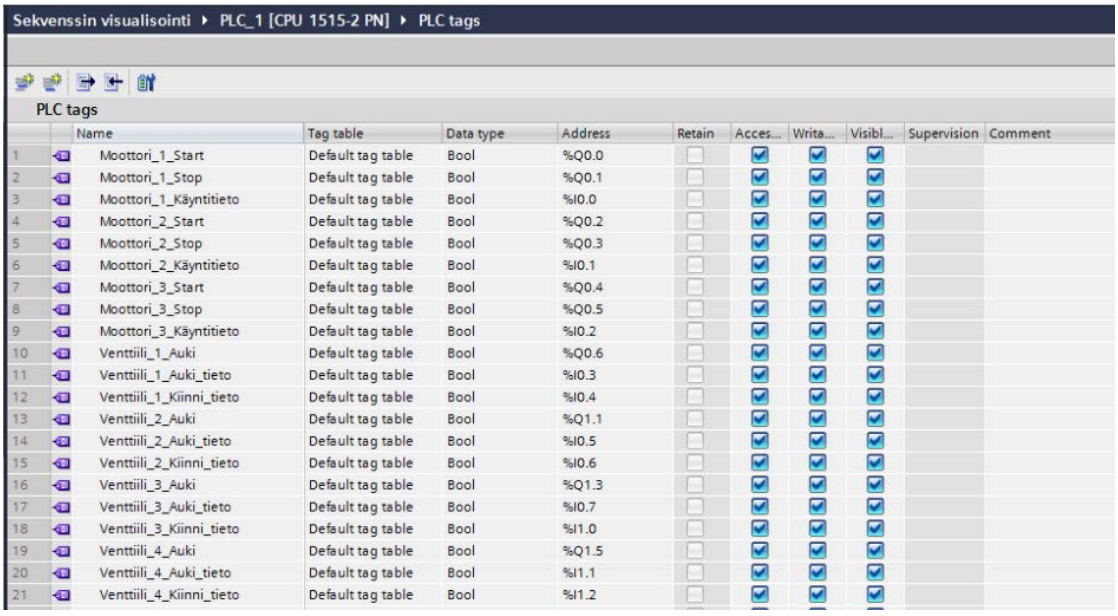
HW-konfiguraatiossa pystytään myös määrittelemään korttien käyttämät osoitteet ja muokkaamaan esimerkiksi analogiakorttien käyttämiä mitta-alueita. Tässä tapauksessa en lähtenyt muokkaamaan perusasetuksia, koska sillä ei olisi ollut merkitystä testailun kannalta.

Seuraavaksi lisäsin projektiin näyttöpaneelin. Paneeliksi valitsin 15-tuumaisen Comfort-sarjan TP1500-paneelin. Tämän valitsin siksi, että Comfort-sarjan paneeleissa pystytään hyödyntämään lähes kaikkia ohjelmasta löytyviä ominaisuuksia. Tämän jälkeen näyttöön tuli asettaa aloitusasetukset. Asetuksissa valittiin yhteysasetukset CPU:n ja paneelin välille. Halutessaan pystyi myös muokkaamaan näyttöpaneelin perusasetuksia, joita olivat esimerkiksi käyttöliittymän taustavärien vaihto.

Kun olin saanut kaikki laitteet lisättyä, tarkistin, että ohjelma ei anna varoituksia ja laitteet ovat varmasti yhteensopivia. Mitään vikailmoituksia ei tullut, ja pääsin siirtymään projektissa eteenpäin.

4.3 Ohjelma

Laitteiston lisäämisen jälkeen oli aika siirtyä itse ohjelman tekoon. Ennen ohjelmakoodin tekoa tein tagi-listan. Tähän listaan lisäsin kuvitteellisten moottoreiden ja venttiilien toiminnot ja näihin toimintoihin liitin I/O-korttien osoitteet. Esimerkiksi moottorille yksi lisäsin kolme eri tagia. Ensimmäisestä ”Moottori_1_Start”-tagista ohjattaisiin moottori päälle. Toinen tagi oli ”Moottori_1_Stop”, joka taas tulisi pysäyttämään moottorin. Viimeisenä lisäsin vielä ”Moottori_1_Käyntitieto”-tagin. Tämän avulla pystyttäisiin näkemään, onko moottori päällä vai ei. Venttiileille lisäsin myös kolme tagia. Ensimmäisestä ”Venttiili_1_Auki” ohjattaisiin venttiili auki. Jos tämä bitti ei olisi päällä, menisi venttiili automaattisesti kiinni, jolloin erillistä ohjausta tälle ei tarvittaisi. Toinen ja kolmas tagi olivat venttiilin asentotietoja. Näiden avulla pystyttäisiin näkemään, kummassa asennossa venttiili on. (Ks. kuvio 8.)



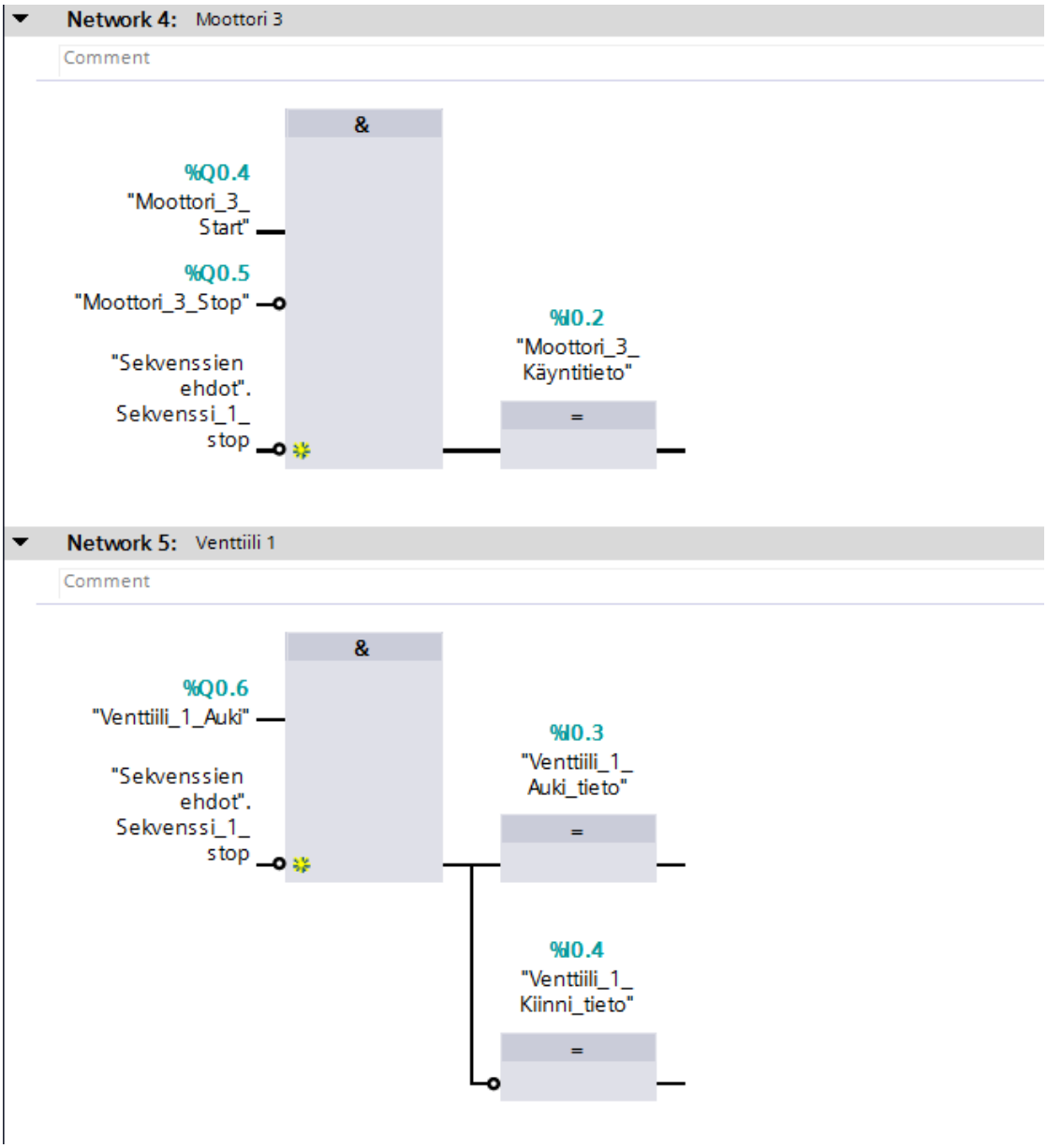
	Name	Tag table	Data type	Address	Retain	Acces...	Writa...	Visibl...	Supervision	Comment
1	Moottori_1_Start	Default tag table	Bool	%Q0.0		☑	☑	☑		
2	Moottori_1_Stop	Default tag table	Bool	%Q0.1		☑	☑	☑		
3	Moottori_1_Käyntitieto	Default tag table	Bool	%I0.0		☑	☑	☑		
4	Moottori_2_Start	Default tag table	Bool	%Q0.2		☑	☑	☑		
5	Moottori_2_Stop	Default tag table	Bool	%Q0.3		☑	☑	☑		
6	Moottori_2_Käyntitieto	Default tag table	Bool	%I0.1		☑	☑	☑		
7	Moottori_3_Start	Default tag table	Bool	%Q0.4		☑	☑	☑		
8	Moottori_3_Stop	Default tag table	Bool	%Q0.5		☑	☑	☑		
9	Moottori_3_Käyntitieto	Default tag table	Bool	%I0.2		☑	☑	☑		
10	Venttiili_1_Auki	Default tag table	Bool	%Q0.6		☑	☑	☑		
11	Venttiili_1_Auki_tieto	Default tag table	Bool	%I0.3		☑	☑	☑		
12	Venttiili_1_Kiinni_tieto	Default tag table	Bool	%I0.4		☑	☑	☑		
13	Venttiili_2_Auki	Default tag table	Bool	%Q1.1		☑	☑	☑		
14	Venttiili_2_Auki_tieto	Default tag table	Bool	%I0.5		☑	☑	☑		
15	Venttiili_2_Kiinni_tieto	Default tag table	Bool	%I0.6		☑	☑	☑		
16	Venttiili_3_Auki	Default tag table	Bool	%Q1.3		☑	☑	☑		
17	Venttiili_3_Auki_tieto	Default tag table	Bool	%I0.7		☑	☑	☑		
18	Venttiili_3_Kiinni_tieto	Default tag table	Bool	%I1.0		☑	☑	☑		
19	Venttiili_4_Auki	Default tag table	Bool	%Q1.5		☑	☑	☑		
20	Venttiili_4_Auki_tieto	Default tag table	Bool	%I1.1		☑	☑	☑		
21	Venttiili_4_Kiinni_tieto	Default tag table	Bool	%I1.2		☑	☑	☑		

Kuvio 8. PLC:n tagi-lista

Seuraavaksi loin FC-ohjelmalohkon. Tämä lohko tulisi toimimaan sekvenssilohkona, jonka sisälle kaikki sekvenssit lisättäisiin. Tämän jälkeen lisäsin FB-lohkon ja asetin ohjelmointikieleksi Graph, joka tarkoittaa, että tämä tulisi olemaan sekvenssilohko.

Tämän jälkeen lisäsin juuri luomani FB-lohkon FC-lohkon sisälle. Tässä vaiheessa ohjelma loi ”Data Blockin” FB-ohjelmalohkolle. Tämä sisältää sekvenssissä käytettäviä tietoja.

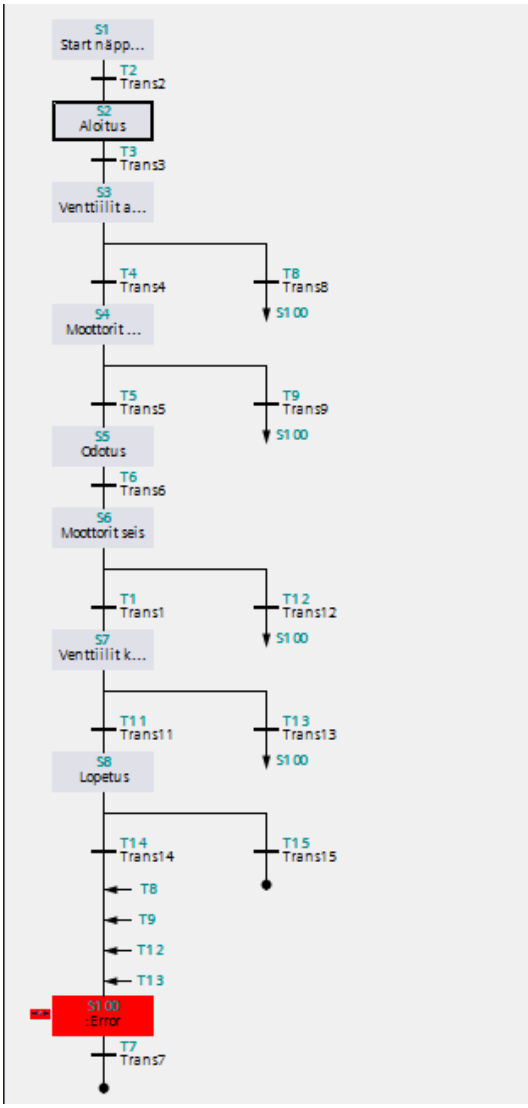
Sekvenssilohkon lisäämisen jälkeen tein tarvittavat toiminnot venttiileille ja moottoreille. Toiminnoiltaan nämä laitteet eivät ole kovin monimutkaisia, joten ohjelmakoodikin on todella yksinkertainen. Kuviossa 9 voidaan nähdä valmiiden ohjauksien rakenne.



Kuvio 9. Moottorin ja venttiilin ohjaus

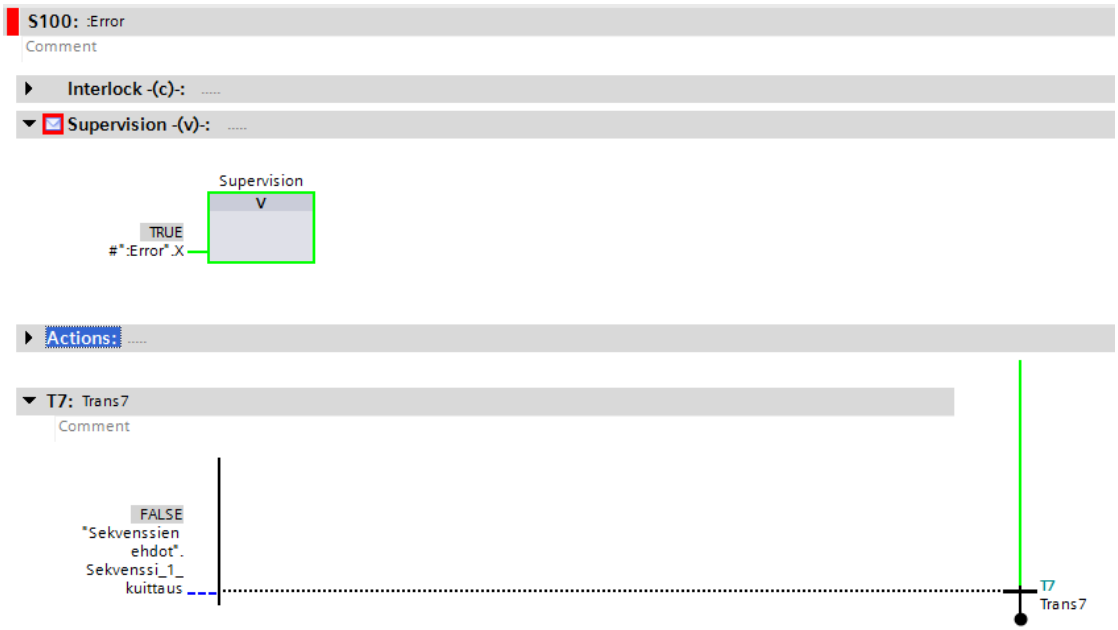
4.4 Sekvenssi

Sekvenssin teon aloitin lisäämällä siihen kaikki askeleet. Askeleita oli kokonaisuudessaan yhdeksän kappaletta. Sekvenssin toiminta oli seuraavanlainen: Kun käyttäjä painaa start-näppäintä, sekvenssi käynnistyy, minkä jälkeen se avaa kaikki neljä venttiiliä. Venttiilien avaamisen jälkeen käynnistetään kaikki kolme moottoria. Näiden jälkeen on odotusaskeleen vuoro. Tämän odotusajan käyttäjä pystyy määrittelemään näytöltä. Odotusajan jälkeen moottorit pysäytetään ja venttiilit suljetaan, minkä jälkeen sekvenssi loppuu ja pysäyttää itsensä. Mikäli jotain vikatilanteita tulisi, siirtyisi sekvenssi Error-askeleeseen, josta pääsisi kuittausnapin avulla pois. (Ks. kuvio 10.)



Kuvio 10. Sekvenssin rakenne, Error-askel aktiivisena

Error-askeleessa hyödynsin ohjelmasta löytyvää ominaisuutta ”Supervision”. Tämän supervision errorin avulla pystytään estämään sekvenssin eteneminen. Tästä error-tilanteesta pääsee pois vain kuittaamalla tämän supervision errorin. Tämä muuttaa myös aktiivisena olevan askeleen punaiseksi, mikä selkeyttää sekvenssin visualisointia. (Ks. kuvio 11.)



Kuvio 11. Error-askeleen sisältö

Testisekvenssissä on vain yksi yhteinen error-askel. Tämän supervision errorin voisi kuitenkin lisätä myös useampaan askeleeseen, jolloin sekvenssi jatkaisi suorittamista kuittauksen jälkeen.

4.5 Sekvenssin operointi ja visualisointi

Sekvenssin operointi tapahtuu sekvenssinohjelmalohkoon lisättävien muistipaikkojen tai osoitteiden avulla. Kun sekvenssiä operoidaan erilliseltä näytöltä tai tietokoneelta, nämä muistipaikat ja osoitteet on siirrettävä jokainen erikseen näytölle. Projekteissa saattaa olla useita sekvenssejä, jolloin siirrettävien muistipaikkojen ja osoitteiden määrä voisi olla todella suuri.

Näiden muistipaikkojen ja osoitteiden eli tagien määrää oli tarkoitus saada karsittua mahdollisimman pieneksi. Pienempi määrä tarkoittaa vähemmän tietojen linkitystä, joka taas vähentää työmäärää ja mahdollisia virheitä. Lähdin tutkimaan ohjelmasta löytyviä toimintoja näiden tietojen siirtämiseen. Löysinkin ohjelmasta toiminnon, jota pystyin hyödyntämään tietojen siirtämisessä. Liitteessä 1 olevassa ohjeistuksessa on kerrottu tarkemmin tästä toiminnosta ja siitä, kuinka se otetaan käyttöön. Ohjeistuksessa on kerrottu myös, miten tiedot lisätään esimerkiksi painonappeihin.

Kun olin saanut sekvenssien ohjauksen tehtyä oli aika siirtyä selvittämään, kuinka saisin sekvenssin visualisoinnin tehtyä. Ensimmäiseksi aloin tarkastelemaan ohjelmasta löytyviä ominaisuuksia ja sitä, miten näitä voisi hyödyntää. Löysinkin muutaman valmiin toiminnon, joita pystyin hyödyntämään työssäni. Näissä ei kuitenkaan ollut kaikkia haluttuja toimintoja, joten jouduin tekemään osan myös itse. Kerron sekvenssin visualisoinnista lisää liitteessä 2 olevassa ohjeistuksessa. Ohjeessa kerrotaan tarkemmin mitä toimintoja otetaan käyttöön ja miten tämä tapahtuu. Kerron myös miten itse tehtävien toimintojen tekeminen tapahtuu.

4.6 Ohjeistus

Kun olin saanut työn valmiiksi, kirjoitin vielä ohjeistuksen, kuinka näitä tekemiäni ohjelmointipohjia käytetään. Ohjeistuksen tekeminen oli helppoa, koska olin työn edetessä tehnyt selkeät muistiinpanot, mitä missäkin vaiheessa tulee tehdä. Lisäsin ohjeistukseen paljon kuvia selkeyttämään toimintojen käyttöönottoa.

Tein yhteensä kaksi ohjetta. Ensimmäisessä ohjeessa kerrotaan, kuinka sekvenssien ohjauspohja luodaan. Toisessa ohjeessa puolestaan kerrotaan, miten sekvenssien visualisointi tapahtuu. Nämä ohjeet menivät suoraan toimeksiantajan käyttöön, joten niitä pystytään hyödyntämään tulevien projektien parissa.

5 Pohdinta

Opinnäytetyön tavoitteena oli luoda toimeksiantajalle logiikkaohjelmointia helpottava sekvenssien visualisointi- ja ohjauspohja. Pohjan oli tarkoitus olla mahdollisimman helppokäyttöinen ja selkeä, jotta sitä pystyttäisiin jatkossa tarpeen vaatiessa muokkaamaan.

Työn tuloksena syntyi kaksi erillistä pohjaa sekvensseille. Toisen avulla pystytään operoimaan sekvenssejä ja toisen avulla puolestaan visualisoimaan sekvenssejä. Molemmille pohjille tehtiin erilliset ohjeistukset, jotka on myös liitetty tämän raportin liitteiksi 1 ja 2. Näiden valmiiden pohjien avulla ohjelmointi tulee olemaan nopeampaa. Pohjien avulla saadaan myös uusia ominaisuuksia käyttöön. Liitteenä olevaa sekvenssin operointiohjetta pystytään hyödyntämään myös muiden laitteiden ohjelmointipohjien teossa.

Tuloksena syntynyttä visualisointipohjaa päästiin hyödyntämään heti uuden projektin parissa. Raporttia tehdessä pohjaa ei oltu vielä käytetty itse prosessin kanssa, mutta simuloinnin kanssa pohja on toiminut hyvin. Enemmän tietoa pohjien toimivuudesta ja soveltuvuudesta saadaan vasta myöhemmin, kun niitä päästään käyttämään useampien projektien kanssa. Tässä yhteydessä selviää myös, kuinka paljon pohjat säästävät aikaa.

Työssä onnistuttiin tekemään kaksi ulkoasuiltaan selkeää ja helppokäyttöistä ohjelmointipohjaa. Molempia ohjelmointipohjia voidaan käyttää useamman sekvenssin visualisointiin ja operointiin, jolloin näytölle siirrettävien muistipaikkojen ja osoitteiden määrä saatiin minimoitua. Operointi- ja visualisointipohjien ohjeistuksista saatiin tiiviit ja helposti seurattavat eri työvaiheista otettujen kuvien avulla.

Pääosin työ onnistui hyvin, mutta ohjelmointipohjan testaaminen oikeilla laitteilla olisi ollut työn tulosten kannalta mielenkiintoista. Simuloimalla prosesseja saatiin kuitenkin melko selkeä kuva siitä, miten pohjat toimisivat oikeiden prosessien kanssa. Pidemmällä aikavälillä selviää myös, kuinka paljon pohjat nopeuttavat logiikkaohjelmointia.

Lähteet

6ES7507-0RA00-0AB0. N.d. Siemensin tuotesivut. Viitattu 14.10.2017. https://mall.industry.siemens.com/mall/en/WW/Catalog/Product/6ES7507-0RA00-0AB0

6ES7515-2AM01-0AB0. N.d. Siemensin tuotesivut. Viitattu 14.10.2017. https://mall.industry.siemens.com/mall/en/WW/Catalog/Product/6ES7515-2AM01-0AB0

Digital modules. N.d. Siemensin tuotesivut. Viitattu 14.10.2017. https://mall.industry.siemens.com/mall/en/WW/Catalog/Products/10204165?tree=CatalogTree

How PLCs Work. N.d. PLCdev:n kotisivut. Viitattu 12.10.2017. http://www.plcdev.com/how_plcs_work

JEEC Oy. Kotisivut. 2017. Viitattu 10.10.2017. http://jeec.fi/

Keinänen, T., Kärkkäinen, P., Lähetkangas, M & Sumujärvi, M. 2007. Automaatiojärjestelmien logiikat ja ohjaustekniikat. Helsinki: WSOY.

Kippo, A. & Tikka, A. 2008. Automaatiotekniikan perusteet. Helsinki: Edita Prima.

Kosketusnäytöt. N.d. Siemensin tuotesivut. Viitattu 13.10.2017. http://www.siemens.fi/fi/industry/teollisuuden_tuotteet_ja_ratkaisut/tuotesivut/automaatiotekniikka/kayttoliittymat/operointipaneelit/kosketusnaytot.php

Myers, M. 2017. Mixing Technology for a Better HMI. Viitattu 14.10.2017. http://www.machinedesign.com/mechanical/mixing-technology-better-hmi

S7-1500. N.d. Siemensin tuotesivut. Viitattu 13.10.2017. http://www.siemens.fi/fi/industry/teollisuuden_tuotteet_ja_ratkaisut/tuotesivut/automaatiotekniikka/ohjelmoitavat_logiikat_simatic/s7_1500.php

Teollinen tiedonsiirto. N.d. Siemensin tuotesivut. Viitattu 13.10.2017. http://www.siemens.fi/fi/industry/teollisuuden_tuotteet_ja_ratkaisut/tuotesivut/automaatiotekniikka/teollinen_tiedonsiirto_esim_profinet/profinet.htm

Valvomo : suunnittelun periaatteet ja käytännöt. 2011. 2.p. Helsinki: Suomen Automaatioseura ry.

Wright, H. N.d. Introduction to Programmable Logic Controllers (PLCs) and the Operational Function of Main System Modules. Maxim Intergrated Products, Inc. Viitattu 12.10.2017. https://www.maximintegrated.com/en/app-notes/index.mvp/id/4701