Tampereen ammattikorkeakoulu

Kaukolämpölaitoksen ohjaus-järjestelmän saneeraus

Badr-Eddin Laouini

OPINNÄYTETYÖ
Marraskuu 2023

Konetekniikka
Koneautomaatio

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Konetekniikan tutkinto-ohjelma
Koneautomaatio

LAOUINI, BADR-EDDIN:
Kaukolämpölaitoksen ohjausjärjestelmän saneeraus

Opinnäytetyö 55 sivua
Marraskuu 2023

Opinnäytetyö toteutettiin AFRYlle, mutta projektin loppuasiakkaana toimi Oulun Seudun Sähkö. Oulun Seudun Sähkö keskittyy liiketoiminnassaan energiantuotantoon, sähkönsiirtoon ja kaukolämmön tuotantoon. Oulun Seudun Sähkö halusi päivittää Liminka POK -lämpölaitoksen ohjausjärjestelmän, koska vanhan järjestelmän komponentit vaativat päivitystä.

Työn yhtenä tavoitteena oli myös perehtyä Siemens Tia Portal -ohjelmointiympäristöön, koska uusi ohjausjärjestelmä toteutettiin käyttäen WinCC Advanced -ohjausjärjestelmää sekä Siemensin 1500 -logiikkaa.

Uutta järjestelmää kehitettiin vanhan järjestelmän pohjalta. Vanhan ohjausjärjestelmän ja olemassa olevan prosessin perusteellisen tarkastelun jälkeen siirryttiin perehtymään uuden järjestelmän ohjelmointiympäristöön. Uuden ohjausjärjestelmän oppimisen ohella projektin etenemistä jatkettiin tehokkaasti.

Käyttöliittymää ja logiikkaa kehitettiin samanaikaisesti, mutta toteutus tapahtui vaiheittain, keskittyen yhteen piiriin kerrallaan. Tämä sisälsi yhteyksien, mittauspiirien, moottoripiirien ja venttiilipiirien sekä sekvenssien ja kokonaisuuden yhdistämisen.

Asiasanat: PLC, ohjausjärjestelmät, logiikat, käyttöliittymät, kaukolämpölaitos

ABSTRACT

Tampere University of Applied Sciences
Degree Programme in Mechanical Engineering
Machine Automation

LAOUINI, BADR-EDDIN:
Renovation of a District Heating Plant's Control System

Bachelor's thesis 55 pages
November 2023

This thesis was implemented for AFRY, but the end customer of the project was Oulun Seudun Sähkö. Oulun Seudun Sähkö wanted to update the control system of the Liminka POK heating plant because the components of the old system required an update. Oulun Seudun Sähkö focuses on energy production, electricity transmission, and district heating production in its business.

One of the goals of the work was also to become familiar with the Siemens Tia Portal programming environment, as the new control system was implemented using the WinCC Advanced user interface and Siemens 1500 logic.

The new system was developed based on the old system. After a thorough examination of the old control system and the existing process, the focus shifted to getting acquainted with the programming environment of the new system.

The user interface and logic were developed simultaneously, but the implementation was carried out in phases, focusing on one circuit at a time. This included connections, measuring circuits, motor circuits, valve circuits, sequences, and combining them into the whole system.

The project successfully met its objectives. The designed automation system passed the Factory Automation Tests (FAT) on the first attempt, indicating effective and successful planning and implementation. The automation system was immediately put into use after the commissioning phase was completed.

Key words: PLC, control systems, logics, user interface, district heating plant

SISÄLLYS

1 JOHDANTO 7
1.1. Työn tausta 7
1.2. Työn tavoite 7
1.3. Työn rakenne 8
1.4. AFRY 9
1.5. Oulun seudun sähkö 9
2 AUTOMAATIO 11
2.1. Ohjelmoitavat logiikat 11
2.2. Logiikoiden ohjelmointikielten standardi 61131-3 12
2.3. Siemens -ohjelmalohkot 16
2.3.1 Organisaatioyksikkö 16
2.3.2 Toiminto FC 16
2.3.3 Toimintayksikkö FB 18
2.3.4 Tiedostoyksikkö 22
3 KAUKOLÄMPÖLAITOS 23
3.1. Kaukolämpölaitoksen toimintaperiaate 23
3.2. Liminka POK -lämpölaitos 23
3.2.1 Lämpölaitoksen prosessi 24
3.2.2 Kommunikointi Liminka BIO -lämpölaitoksen kanssa 25
4 OHJAUSJÄRJESTELMÄ 26
4.1. Vanhan ohjausjärjestelmän rakenne 26
4.2. Uuden ohjausjärjestelmän rakenne 27
5 PLC OHJELMAN SUUNNITTELU 29
5.1. Mittaukset 29
5.2. Ohjaukset 31
5.2.1 Pumppujen ohjaukset 31
5.2.2 Venttiilien ohjaukset 33
5.3. PLC-PLC kommunikointi 35
5.3.1 GET: Datan luku toisesta logiikasta 35
5.3.2 PUT: Datan lähettäminen toiseen logiikkaan 36
5.3.3 GET & PUT lohkojen käyttöehdot 36
6 KÄYTTÖLIITTYMÄN SUUNNITTELU 38
6.1. Käyttöliittymäprojektin luonti Tia Portal -ohjelmointiympäristössä 38
6.2. Yhteyden luominen logiikan ja käyttöliittymän välillä. 40
6.3. Navigoinnin luominen käyttöliittymään 41
6.4. Prosessikuvan piirtäminen 43

6.5. Muuttujien tuominen logiikasta käyttöliittymäympäristöön 44
6.6. Ponnahdusikkunoiden luominen (Faceplate) 46
7 POHDINTA JA YHTEENVETO 48
LÄHTEET 50
LIITTEET 52
Liite 1. Siemensin organisaatioyksiköt S7-1200/1500 sarjan logiikoille Tia-Portaalissa 52
Liite 2. Siemensin organisaatioyksiköt S7-300/400 sarjan logiikoille Tia-Portaalissa 54

1 JOHDANTO

1.1. Työn tausta

Tämä opinnäytetyö tehtiin AFRY:lle, mutta projektin loppuasiakkaana toimi Oulun Seudun Sähkö. Työssä käsitellään kaukolämpölaitoksen ohjausjärjestelmän saneerausta. Kaukolämpölaitoksen vanha ohjausjärjestelmä oli toteutettu Reginin EXOflex EP1011 -logiikalla. Käyttöliittymä oli toteutettu Reginin EXOscadalla, joka toimii PC:llä. PLC-ohjelma ja käyttöliittymä oli tehty EXOdesignerilla.

Työ oli osa isompaa projektia, jossa tehtiin kolmelle muulle kaukolämpölaitokselle automaatiosaneeraus. Kaikki laitokset tarvitsivat saneerauksen, koska komponentit alkoivat olla päivityksen tarpeessa. Liminka POK -lämpölaitokseen haluttiin myös lisätä mahdollisuus ohjata ja valvoa laitosta etänä Liminka BIO- lämpölaitoksesta. EXOdesigner -ohjelmisto ei ole kovinkaan monelle tuttu ohjelmointiympäristö, ja siksi ohjelmaan oli nykypäivänä vaikeaa löytää osaajia tekemään erilaisia päivityksiä.

Työssä tutustutaan lisäksi TIA-Portal ohjelmointiympäristöön, koska uusi ohjausjärjestelmä tehtiin kyseisessä ohjelmointiympäristössä.

1.2. Työn tavoite

Työn tavoitteena oli suunnitella Liminka POK -lämpölaitokselle uusi automaatiojärjestelmä, joka toimisi lähes samalla tavalla kuin vanha järjestelmä.

Työtä lähdettiin tekemään tutkimalla vanhaa PLC-ohjelmaa, tutustumalla vanhaan käyttöliittymään ja haastattelemalla laitoksen käyttäjiä. Tämän kaiken perusteella saatiin jonkinlainen ymmärrys siitä, minkälainen uuden automaatiojärjestelmän pitäisi olla. Lopputulokseksi haluttiin automaatiojärjestelmä, joka on tehokkaampi ja käytännöllisempi, mutta toimintaperiaatteeltaan melkein samanlainen kuin vanha automaatiojärjestelmä. Käytännöllisyyden ja tehokkuuden lisääminen onnistui siten, että uuden ohjausjärjestelmän avulla laitosta oli mahdollista

ohjata ja valvoa kokonaan Liminka BIO- lämpölaitoksesta, joka sijaitsee noin kahden kilometrin päässä Liminka POK -lämpölaitoksesta.

1.3. Työn rakenne

Työn toisessa luvussa tutustutaan yleisesti automaatioon liittyviin asioihin, sekä paneudutaan hieman TIA-Portal ohjelmointiympäristöön. Luvussa esitellään joitakin Siemensin ohjelmoitavia logiikoita ja ohjelmoitavissa logiikoissa käytettävää ohjelmointikielen standardia, sekä selitetään mitä tarkoittaa lyhenteet OB, FB, FC, DB ja miten niitä käytettiin ohjelmaa tehdessä.

Kolmannessa luvussa tutustutaan kaukolämpölaitoksiin yleisesti ja esitellään Liminka POK -lämpölaitosta. Luvussa selostetaan myös Liminka POK -lämpölaitoksen toiminta.

Neljännessä luvussa esitellään vanhan ja uuden ohjausjärjestelmän rakenteet.

Viidennessä luvussa esitellään kaikki ohjelmassa olevat mittaus-, ohjaus- ja kommunikointilohkot, sekä selitetään, miten niitä käytettiin. Luvussa käydään myös lyhyesti läpi, miten POK -laitoksen ja BIO -laitoksen välinen kommunikaatio toteutettiin.

Kuudennessa luvussa käydään läpi, miten käyttöliittymä suunniteltiin. Siinä esitellään myös lyhyesti, miten ”faceplattejä” hyödynnettiin käyttöliittymän teossa.

Seitsemännessä luvussa kootaan yhteen tulokset, ja pohditaan työn onnistumista. Samassa luvussa pyritään myös tuomaan esiin mahdolliset parannusehdotukset paremman tuloksen saamiseksi.

1.4. AFRY

Työn toimeksiantajana toimi AFRY, joka on Euroopan yksi johtavista suunnittelu- ja konsultointiyhtiöistä. Yritys syntyi vuonna 2019, kun Ruotsissa v. 1895 perustettu ÅF yhdistyi Suomessa v. 1958 perustettuun Pöyryyn. AFRY työllistää noin 19 000 työntekijää ympäri maailmaa ja noin 2800 työntekijää Suomessa. [1]

AFRY:ssä on kuusi divisioonaa:

- Prosessiteollisuus
- Rakennettu ympäristö
- Energia
- Liikkeenjohdon konsultointi
- AFRY X
- Teolliset ja digitaaliset ratkaisut

Yritys tarjoaa muutosta kohti kestävää yhteiskuntaa. AFRY tarjoaa ratkaisuja globaaleihin megatrendeihin, kuten teollisuuden digitalisaatioon, biotalouteen, energiamarkkinoiden murrokseen sekä kaupungistumiseen. [1]

1.5. Oulun seudun sähkö

Työn loppuasiakas on Oulun Seudun Sähkö, joka tuottaa jäsenilleen edullisia ja laadukkaita palveluita kuluttajaosuuskuntana. Oulun Seudun Sähkön liiketoiminta perustuu energiantuotantoon, sähkönsiirtoon, kaukolämmitykseen ja valokuituun. [2]

Kokonaan jäsentensä omistama Oulun Seudun Sähkö on perustettu 26.3.1921. Omistus on nykypäivänä jaettu 12 sähköosuuskunnan ja kolmen muun yhteisön kesken. [2]

Oulun Seudun Sähkön 12 sähköosuuskuntaa ovat seuraavat [3] :

- Alatemmeksen Sähköosuuskunta
- Kempeleen Energiaosuuskunta
- Limingan Sähköosuuskunta
- Sähköosuuskunta Lumi

- Muhoksen Sähköosuuskunta
- Oulunsalon Sähköosuuskunta
- Salonpään Sähköosuuskunta
- Temmeksen Sähköosuuskunta
- Tyrnävän kirkonkylän Sähköosuuskunta
- Utajärven Sähköosuuskunta
- Vaalan Sähköosuuskunta
- Ängeslevän Sähköosuuskunta

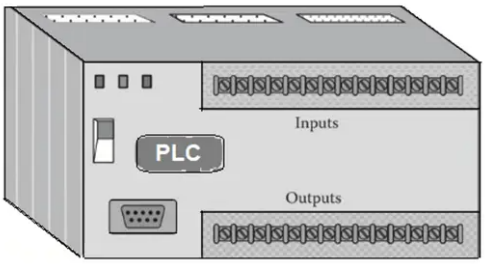
2 AUTOMAATIO

2.1. Ohjelmoitavat logiikat

Ohjelmoitavat logiikat, eli PLC:t ja teollisuus-PC:t, ovat yleisimpiä käytössä olevia automaation ohjauslaitteita [4, s.248]. Ohjelmoitavat logiikat vastaanottavat dataa tulojensa kautta, ja lähettävät ohjaussignaaleja lähtöjensä kautta. Niiden päätavoitteena on ohjata järjestelmän toimintoja ohjelmoidun sisäisen logiikan avulla [5]. Yritykset ympäri maailmaa käyttävät ohjelmoitavia logiikoita erilaisten prosessien automatisointiin.

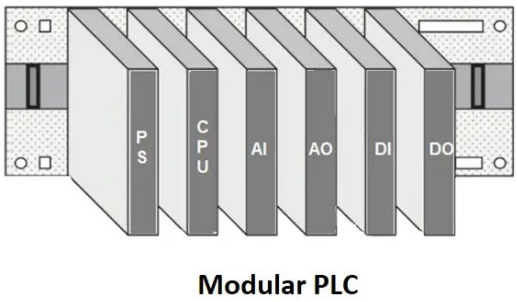
Logiikoita on kahdenlaisia: kompakteja ja modulaarisia [6]. Kompaktilogiikat ovat tarkoitettu pieniin koneohjaustehtäviin, koska laajentamismahdollisuudet ovat rajalliset. Kompaktilogiikoissa on sisäänrakennettu virtalähde eli PSU, keskusyksikkö eli CPU, sekä rajallinen määrä integroituja tuloja ja lähtöjä.

Modulaarinen logiikka on telinepohjainen järjestelmä, jossa PSU, CPU ja prosessin tai koneen ohjaukseen tarvittava määrä tulo- ja lähtökortteja (I/O-kortteja) löytyvät kaikki erillisinä komponentteina asennettuna yhteen tai useampaan telineeseen. Kuvassa 1 on esimerkki kompaktilogiikan mallista ja kuvassa 2 esimerkki modulaarisen logiikan mallista.



Compact PLC

Kuva 1. Esimerkkikuva kompaktilogiikan mallista.

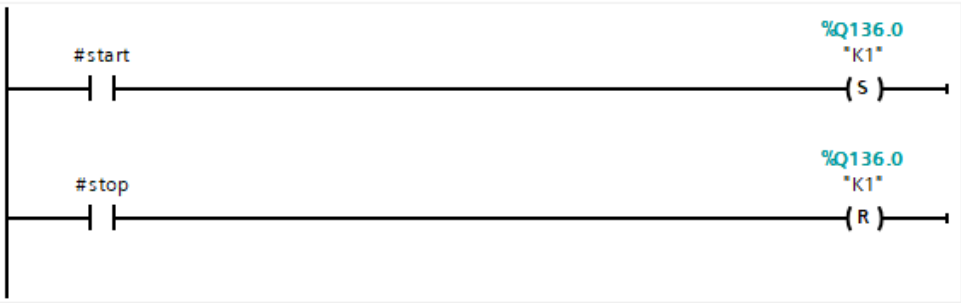


Kuva 2. Esimerkkikuva modulaarisesta logiikkamallista.

2.2. Logiikoiden ohjelmointikielten standardi 61131-3

IEC 61131 osa 3 määrittelee ohjelmoitavien logiikoiden yhtenäisen ohjelmointikielisarjan syntaksin ja semantiikan [7]. Pääsääntöisesti lähes kaikki suurimmat PLC-valmistajat noudattavat kyseistä standardia. Standardi sisältää viisi erilaista ohjelmointikieltä, joista kaksi on graafisia ja kaksi tekstikielipohjaisia. Viimeinen ohjelmointikieli ei itse asiassa ole ohjelmointikieli sen perinteisessä merkityksessä, vaan pikemminkin graafinen lähestymistapa ohjelmakoodin jäsentämiseen [8].

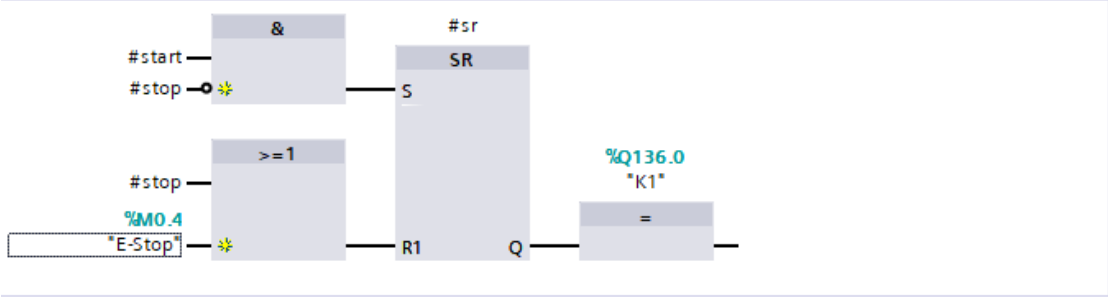
1. **LD** eli tikapuukaavio on yleisin PLC-ohjelmointikieli, jonka graafiset ominaisuudet on suunniteltu näyttämään ja käyttäytymään kuten sähkölaitteiden koskettimet [9]. LD-kieli on erittäin helposti kirjoitettava ja luettava ohjelmointikieli, minkä vuoksi sitä käytetäänkin eniten. Useimmiten PLC-ohjelmointia opettelevat aloittavat ohjelmoinnin LD-kielellä, ja siksi työmarkkinoilla LD-kieltä pidetäänkin aloittelijan kielenä. Kielen tehokkuus ei ole kielistä parhain. Kuvassa 3 on esimerkki LD-kielellä kirjoitetusta koodinpätkästä, jossa on lähtö K1, joka aktivoituu, kun "start" -muuttuja saa arvokseen yhden. K1-lähtö deaktivoidaan, kun "stop" -muuttuja saa arvokseen yhden.



Kuva 3. Esimerkkikuva LD-kielestä

2. **FBD** eli toimilohkokaavio on toinen suosittu graafinen ohjelmointikieli. Se koostuu erilaisista toimilohkoista, joilla on tulot ja lähdöt. Ohjelmoija voi itse rakentaa oman toimilohkon, jolloin hän pystyy määrittämään toimintoja. Halutessaan hän voi myös käyttää ohjelmointialustan valmiita operaatiotoimilohkoja, kuten AND, OR ja NOT, sekä toimilohkoja, kuten ajastimia ja laskureita. [10]

   FBD-kieli on suhteellisen yksinkertainen, vaikka sen käyttö vaatiikin jonkin verran koulutusta. Kieli on loistava valinta, kun tehdään turvallisuuteen liittyvää koodia, PID-silmukoita, aliohjelmia tai analogista tulosuodatusta [10]. Kuvassa 4 on esimerkki FBD-kielellä tehdystä koodinpätkästä.



Kuva 4. Esimerkkikuva FBD-kielestä

3. **ST** eli strukturoitu teksti on tekstipohjainen ohjelmointikieli [8, s.205]. Se on korkean tason kieli, jonka avulla voidaan suorittaa useita eri toimintoja ja komentoja yhdellä komentorivillä. Jos vertaamme sitä muihin korkean tason kieliin, ST muistuttaa eniten Pascalia tai C-kieltä. ST-kieli on kaikista viidestä kielestä kaikkein tehokkain. Se on kehitetty ohjelmoimaan monimutkaisia aritmeettisia funktioita, käsittelemään taulukoita sekä työskentelemään sanaobjektien ja tekstin kanssa.

ST-kieli voi tuntua aluksi hankalalta varsinkin vikatilanteiden selvittelyssä. Graafisiin kieliin verrattuna ST-kieltä on vaikeampaa seurata ja siksi koodin seuraaminen saattaa tuntua monimutkaiselta. Monesti yrityksillä onkin omat graafisiin kieliin erikoistuneet teknikot, jotka tutkivat vianselvittelyssä koodia. Kuvassa 5 on esimerkki ST-kielellä tehdystä koodinpätkästä.

```
1  IF (#set = TRUE) AND (#reset = FALSE) THEN
2
3      #LAHTO := TRUE;
4
5
6  ELSE
7
8      IF (#reset = TRUE) THEN
9
10
11         #LAHTO := FALSE;
12
13     END_IF;
14
15
16 END_IF;
```

Kuva 5. Esimerkkikuva ST-kielestä

4. **IL** on tekstipohjainen matalan tason ohjelmointikieli. Matalan tason ohjelmointikielellä tarkoitetaan sitä, että koodi on konekieltä tai assembler-koodia [8]. Käytännössä siis tällaisen koodin suorittaminen ei vaadi logiikalta paljon tehoa. IL-kieli tarjoaa täydellisen hallinnan, koska tietoja käsitellään itse prosessorissa. IL-kielen osaaminen mahdollistaa PLC:n arkkitehtuurin toiminnan ymmärtämisen [11].

IL kielen suosio on laskenut, koska kieli on kaikista viidestä kielestä hankalin. Normaalit SET/RESET -logiikkaohjaukset ja laskennat voidaan tehdä nopeasti ja tiiviisti, mutta monimutkaisemmat toiminnot vaativat erittäin hyvää ohjelmointikielen osaamista [4, s.259].

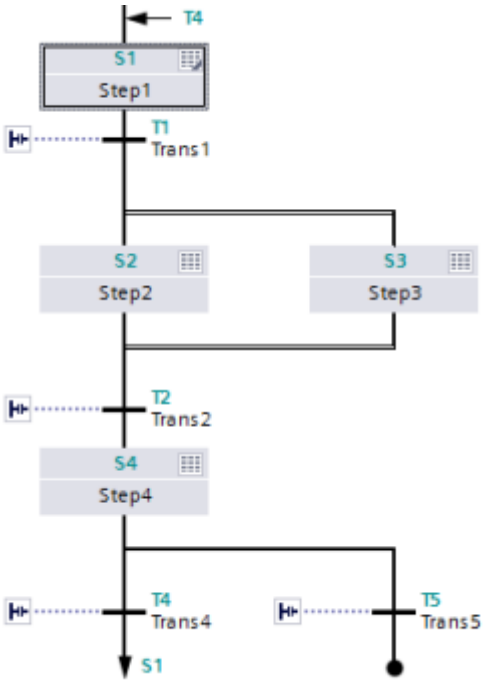
Nykypäivänä logiikoiden prosessorit ovat paljon tehokkaampia kuin vuosia takaperin. Tämän takia IL-kielen antama prosessorin vähäinen kuormitus-

hyöty ei ole nykypäivänä enää niin tarpeellinen. Ohjelmoijalle on yksinkertaisempaa käyttää muita kieliä tehdessään logiikkaohjelmaa. Kuvassa 6 on esimerkki IL-kielellä tehdystä koodinpätkästä.

```
1     A    #START
2     AN   #STOP
3     S    #SR
4     A(
5     O    #STOP
6     O    "E-Stop"                %M0.4
7     )
8     R    #SR
9     A    #SR
10    =    "K1"                    %Q136.0
```

Kuva 6. Esimerkkikuva IL-kielestä.

5. **SFC** eli sekvenssifunktiokaavio on graafinen työkalu, jota käytetään sekvenssiohjauksissa [8, s.210]. Se on askelista muodostuva ohjelma, jonka askeleet linkitetään toisiinsa siirtoehdolla. Askeleet sisältävät ohjauskomennot, jotka voidaan kirjoittaa eri ohjelmointikielillä (ST, IL, LD ja FBD) [4, s.266]. Kuvassa 7 on esimerkki SFC-kielellä tehdystä koodinpätkästä.



Kuva 7. Esimerkkikuva SFC-kielestä.

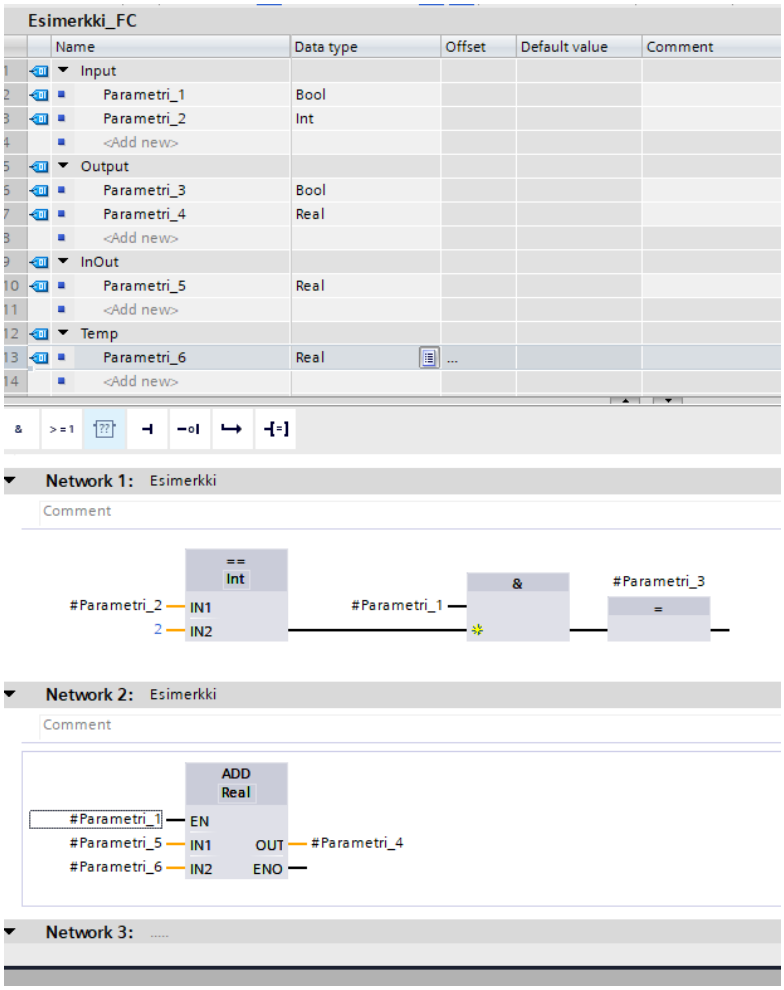
2.3. Siemens -ohjelmalohkot

2.3.1 Organisaatioyksikkö

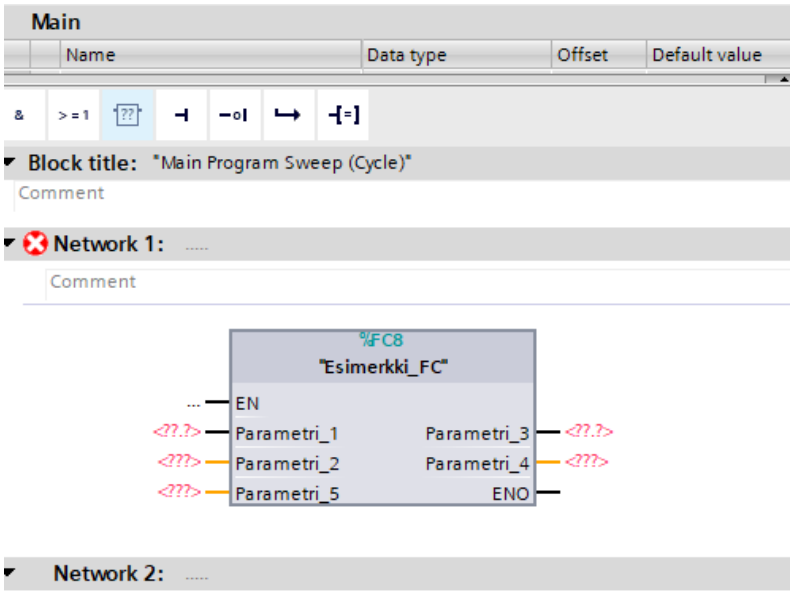
OB eli organisaatioyksikkö toimii käyttöjärjestelmän ja sovelluksen välisenä rajapintana. Organisaatioyksiköitä on erilaisia ja jokaisella on oma tehtävänsä. Kuitenkin kaikista tärkein organisaatioyksikkö on OB1, joka on pääohjelma. PLC:n koko ohjelman voi halutessaan kirjoittaa suoraan OB1:n. Käyttöjärjestelmä kutsuu OB1:n syklisesti [12]. Liitteestä 1 löytyvät kaikki Siemensin organisaatioyksiköt S7-1200/1500 sarjan logiikoille (taulukko 1) ja liitteestä 2 kaikki organisaatioyksiköt S7-300/400 sarjan logiikoille (taulukko 2).

2.3.2 Toiminto FC

Toiminto, eli FC, on ohjelmalohko, jolla ei ole omaa muistia. Sitä käytetään yleensä monimutkaisen toiminnon määrittämiseen. Ohjelmalohkoa voidaan kutsua toiseen FC:hen, OB1:seen tai FB:hen niin monta kertaa kuin halutaan. Sen sisälle voidaan määrittää toiminnolle erilaisia parametrejä, kuten sisääntulo (In), lähtö (Out), sisääntulo ja lähtö (In-Out), sekä temp. FC-ohjelmalohkossa määritellyt temp -muuttujat tallennetaan paikalliseen tietopinoon, joka katoaa ohjelmalohkon suorittamisen jälkeen. FC:tä kutsuessa In, Out ja In-Out -parametreja ei voi jättää tyhjiksi, koska FC:n muodolliset parametrit eivät tallennu mihinkään muistiin, joten alkuarvoa niille ei ole annettu. Kuva 8 näyttää, miltä FC:n sisältö voi näyttää. Kuvassa 9 näkyy miltä FC näyttää, kun sitä kutsutaan pääohjelmaan.[14]



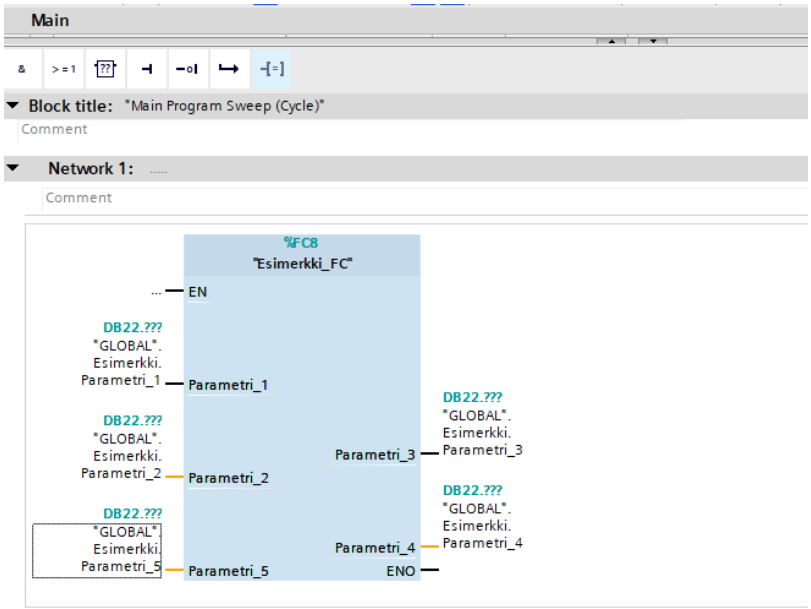
Kuva 8. Esimerkkikuva FC-lohkon sisällöstä.



Kuva 9. Esimerkkikuva FC-lohkosta, jota kutsutaan ilman parametrejä.

Kuvassa 9 parametrit on jätetty tyhjiksi, vaikka niin ei saisi tehdä, sillä tällöin ohjelmaa ei voi ladata logiikkaan. Kuva 10 on samasta tilanteesta kuin kuva 9, mutta

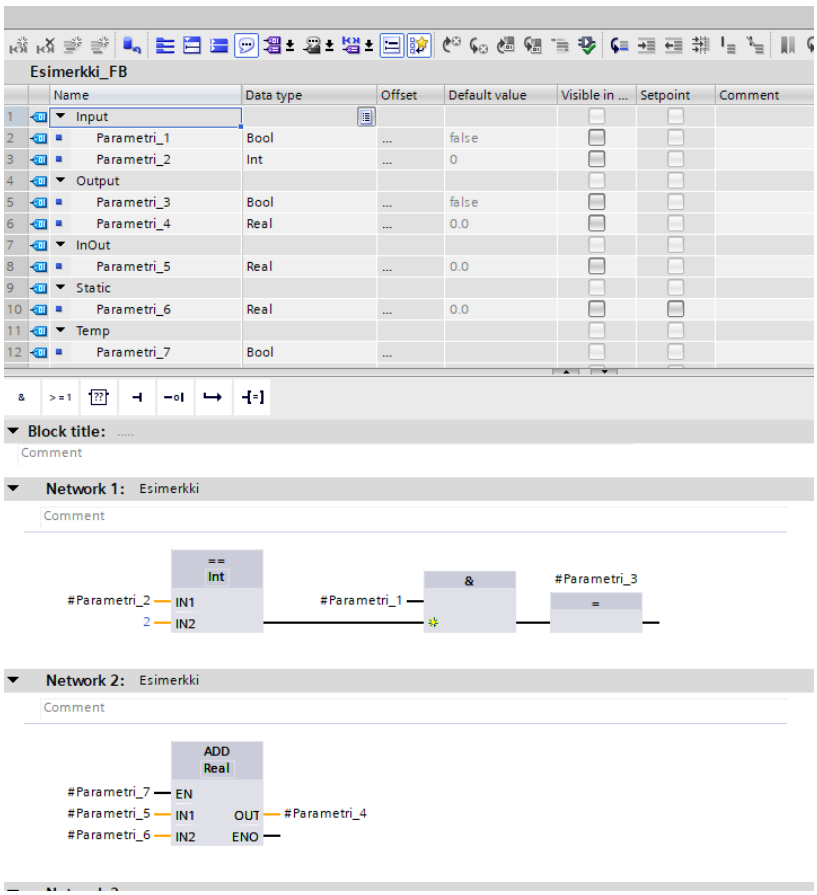
nyt parametreja ei ole jätetty tyhjäksi. Tällöin mitään punaista ei näy ja ohjelma voidaan ladata logiikkaan.



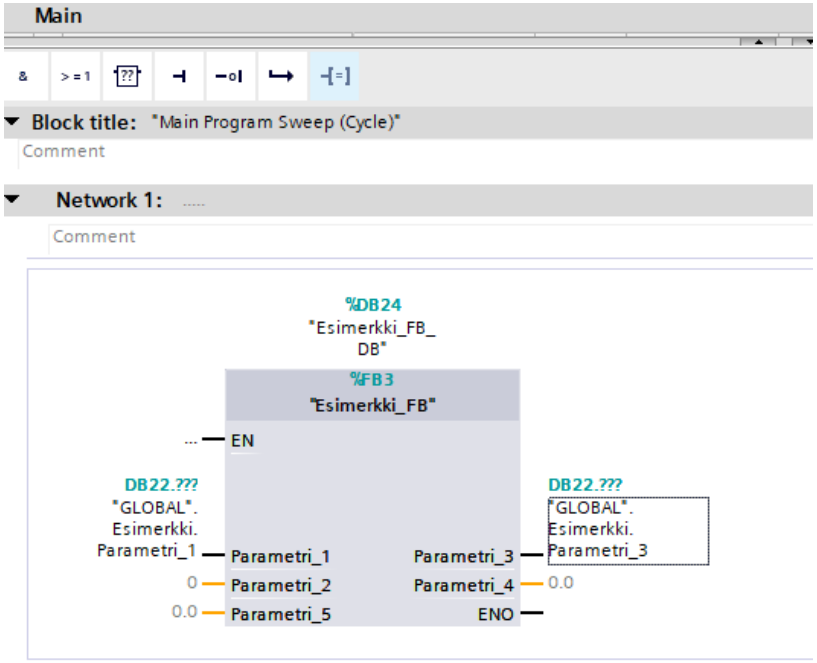
Kuva 10. Esimerkkikuva FC-lohkosta, jossa sitä kutsutaan parametreillä.

2.3.3 Toimintayksikkö FB

Toimintayksikkö, eli FB, on ohjelmalohko, jolla on oma muisti: DB (instanssitiedostoyksikkö. Ohjelmalohkoa voidaan kutsua toiseen FC:hen, OB1:seen tai FB:hen niin monta kertaa kuin halutaan. Ohjelmalohkoa kutsuttaessa se luo muistikseen tietokannan. Tätä tietokantaa kutsutaan instanssitiedostoyksiköksi. Sen sisälle voidaan määrittää toiminnolle erilaisia parametrejä, kuten In, Out, In-Out, temp sekä static. Static löytyy ainoastaan FB:stä ja se eroaa temp -muuttujista siten, että muuttujien tiedot tallennetaan instanssitiedostoyksiköihin eikä paikalliseen tietopinoon. Tällöin FB-ohjelman suorituksen jälkeen static -muuttujat ovat muistissa ja temp -muuttujat katoavat. FB:tä kutsuttaessa voidaan jättää mikä tahansa parametrin osa ilman osoitetta, koska tällöin ohjelma käyttää instanssitiedostoyksikön arvoja. Kuva 11 näyttää, miltä FB:n sisältö voi näyttää. Kuvassa 12 näkyy miltä FB näyttää, kun sitä kutsutaan pääohjelmaan.[14]



Kuva 11. Esimerkkikuva FB-lohkon sisällöstä.



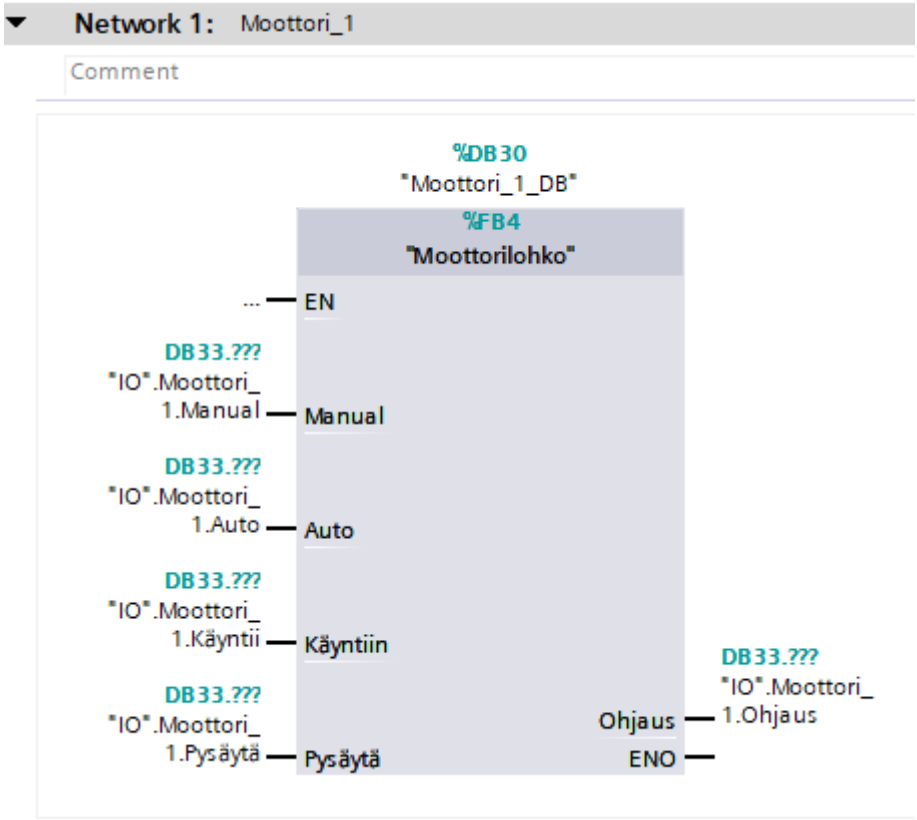
Kuva 12. Esimerkkikuva FB-lohkosta, kun sitä kutsutaan parametreillä ja ilman parametrejä.

FB-lohkoja voidaan kopioida, jolloin ne suorittavat ohjelmakoodin omilla muuttujillaan. FB:n käytännöllisyys voidaan parhaiten selittää esimerkkitilannetta käyttäen:

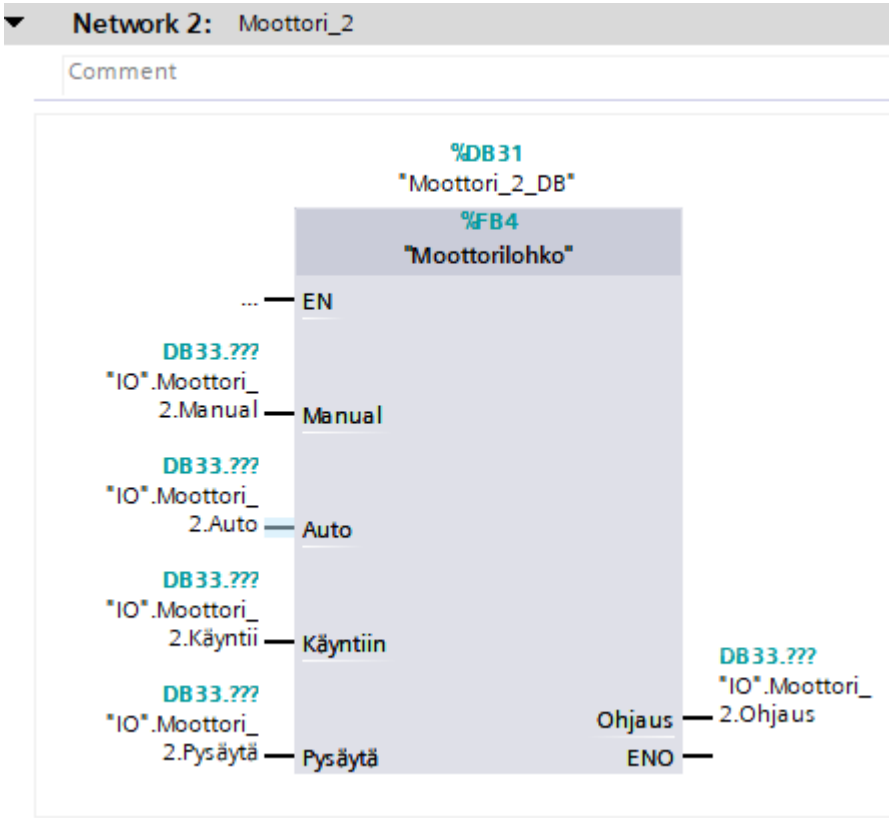
Käynnissä on prosessi, jossa pitää ohjata kolmea samanlaista moottoria eri tilanteissa:

1. Moottori_1
2. Moottori_2
3. Moottori_3

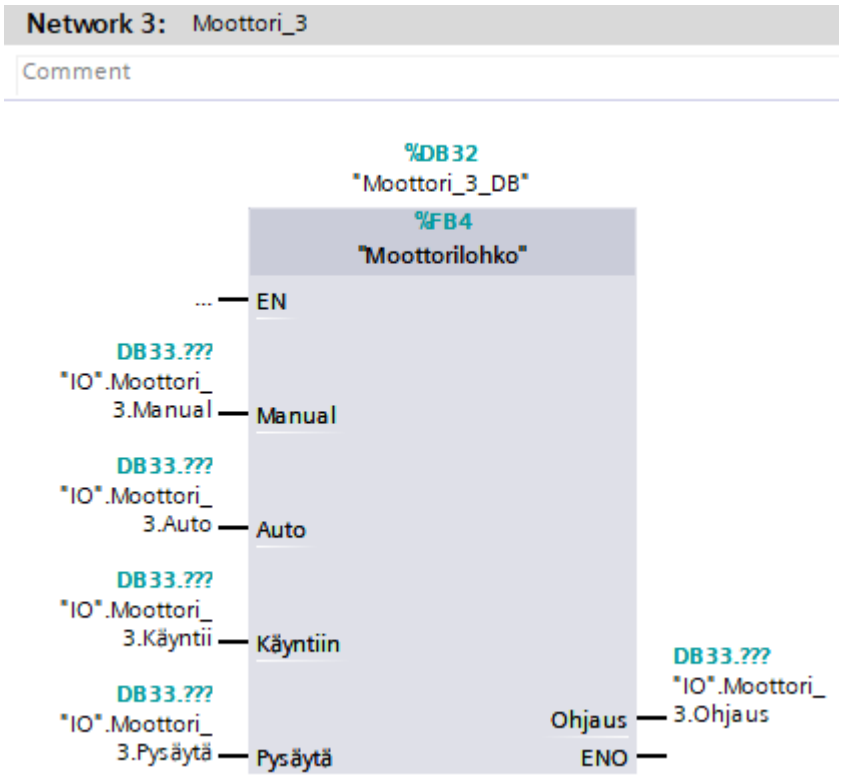
Ohjelmoija tekee yhden moottorilohkon FB:nä ja kutsuu tämän FB:n kolme kertaa esimerkiksi pääohjelmaan OB1:seen. Jokaisella kutsulla pitää tehdä oma instanssitiedostoyksikkö, jonne tallennetaan yhden moottorin tiedot. Muuttujiksi ohjelmoija laittaa halutun moottorin tarvittavat tiedot. Kuvissa 13-15 on esimerkkikuvia moottorilohkon käytöstä, jolla saadaan esimerkkitilanteen prosessi toimimaan.



Kuva 13. Moottorilohko moottori_1:lle.



Kuva 14. Moottorilohko moottori_2:lle.



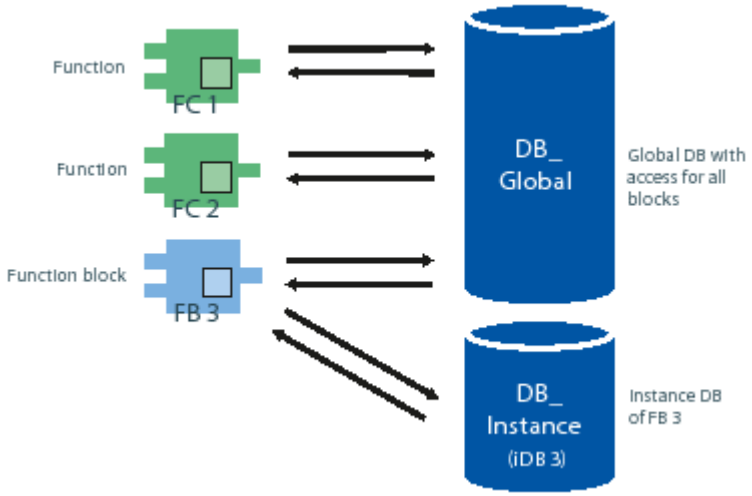
Kuva 15. Moottorilohko moottori_3:lle.

Kuvista 13, 14 ja 15 huomataan, että kaikissa on käytetty %FB4 moottorilohkoa ja jokaisella lohkolla on oma instanssitiedostoyksikkö %DB30-%DB32.

2.3.4 Tiedostoyksikkö

DB:t eli tiedostoyksiköt ovat käyttäjän luomia muistialueyksiköitä. Ne toimivat muuttuvan datan muistipaikkana, jota käyttäjäohjelma hyödyntää. Globaalit tiedostoyksiköt säilyttävät dataa, jota voidaan käyttää kaikissa muissa ohjelmalohkoissa. Tiedostoyksiköiden maksimikoko vaihtelee käytetyn CPU:n mukaan. Käyttäjä voi itse määrittää globaalien tiedostoyksiköiden rakenteen haluamallaan tavalla. Käyttäjä voi halutessaan hyödyntää PLC-tietotyyppejä, kuten käyttäjäkohtaisesti määritettyjä datatyyppejä (UDT), kun luodaan muuttujia tiedostoyksiköihin.

Jokainen funktioyksikkö, funktio tai organisaatioyksikkö voi lukea dataa globaalista tiedostoyksiköstä, tai kirjoittaa itse dataa globaaliin tiedostoyksikköön. Tämä data säilyy tiedostoyksikössä myös sen jälkeen, kun tiedostoyksiköstä poistutaan. Globaali tiedostoyksikkö ja instanssitiedostoyksikkö voidaan avata samanaikaisesti. Kuva 16 näyttää miten tiedostoyksiköitä voidaan käyttää. [15]



Kuva 16. Esimerkkikuva erilaisista tavoista käyttää tiedostoyksiköitä. [15]

3 KAUKOLÄMPÖLAITOS

3.1. Kaukolämpölaitoksen toimintaperiaate

Kaukolämpö on kuuman, virtaavan veden siirtämistä voimalaitokselta tai lämpökeskuksesta asiakkaiden luokse ja takaisin. Yleensä kaukolämpöä tuottavassa laitoksessa käytetään lämminvesikattilaa, joka lämmittää vettä ilman sen monimutkaista käsittelyä. Tämä kuuma vesi kulkee suljetussa kaukolämpöverkossa, joka pitää sen erillään ympäröivästä ympäristöstä. Kuuma vesi virtaa menoputkessa kohti asiakkaan lämmönvaihdinta, jonka kautta se lämmittää kiinteistön lämmitysjärjestelmän veden sekä tuottaa lämmintä käyttövettä. Lämmönvaihdin pitää kaukolämmön ja kiinteistön vesijärjestelmän erillään toisistaan, joten ne eivät koskaan sekoitu keskenään. [16]

Kaukolämmön veden lämpötila vaihtelee vuodenaikojen mukaan ja on korkeimmillaan talvella, jolloin sitä tarvitaan enemmän lämmitykseen. Keskikesällä sen sijaan lämpötila on alhaisempi, koska silloin lämpöä tarvitaan lähinnä lämpimän käyttöveden tuottamiseen. [16]

3.2. Liminka POK -lämpölaitos

Liminka POK -lämpölaitos on varalämpölaitos Limingassa, joka käyttää kevyttä polttoöljyä polttoaineena. Laitosta käytetään tilanteissa, joissa Liminka BIO -lämpölaitoksen tuottama lämpö ei riitä ylläpitämään tarvittavaa lämpötilaa.

Vaikka Liminka POK -lämpölaitos on huomattavasti pienempi kuin Liminka BIO -lämpölaitos, se pystyy tuottamaan tarvittavan määrän lämpöä erittäin nopeasti ja tehokkaasti, sillä polttoöljy on polttoaineena hyvin tehokas. Kuitenkaan öljyn jatkuva käyttö ei ole kustannustehokasta eikä ympäristöystävällistä energiantuotantomuotoa pitkällä aikavälillä.

3.2.1 Lämpölaitoksen prosessi

Liminka POK -lämpölaitoksen toimintaprosessi on yksinkertainen ja sitä voidaan ohjata joko manuaalisesti tai automaattisesti. Manuaaliohjauksella käyttäjä pystyy ohjaamaan kaikkia taulukossa 3 mainittuja laitteita.

TAULUKKO 3. Ohjattavissa olevat laitteet laitoksessa

Ohjattavissa olevat laitteet.		
Laite	Tyyppi	Selitys
POK1 -kattila	Digitaali- lähtö	Öljykattilan ohjaus
POK2 -kattila	Digitaali- lähtö	Öljykattilan ohjaus
POK3 -kattila	Digitaali- lähtö	Öljykattilan ohjaus
POK1 -venttiili	Digitaali- lähtö	Öljykattilan venttiilin ohjaus
POK2 -venttiili	Digitaali- lähtö	Öljykattilan venttiilin ohjaus
POK3 -venttiili	Digitaali- lähtö	Öljykattilan venttiilin ohjaus
P1 -pumppu	Digitaali- lähtö	Kaukolämpöpumppu
P2 -pumppu	Digitaali- lähtö	Kaukolämpöpumppu
P3-P4 -pumput	Digitaali- lähtö	Paineenpitopumppu
KL -venttiili	Digitaali- lähtö	Kaukolämpöventtiili
P1 - tamu	Digitaali- lähtö	Taajuusmuuntajan kontaktori

Automaattiohjauksella logiikka hallitsee laitteiden ohjausta. Se seuraa Liminka BIO -lämpölaitoksen lähtevän veden lämpötilaa: mikäli lämpötila laskee määritetyn tason alle, POK -lämpölaitos saa käynnistyskäskyn. Logiikka käyttää seuraavaa kaavaa käynnistyskäskyn ymmärtämiseen

$$(T_{SP} - T_{ACT}) \geq T_K$$

, jossa T_{SP} on ”set point” lämpötila eli asetuslämpötila, jota systeemi pyrkii saavuttamaan, T_{ACT} on todellinen lämpötila ja T_K on käyttäjän asettama käynnistyslämpötilaero. Eli logiikka ymmärtää käynnistää laitoksen, kun kaava toteutuu.

Kun Liminka POK -lämpölaitos saa käynnistyskäskyn, se käynnistää yhden öljykattilan kerrallaan. Jokaiselle kattilalle on annettu numero 1-3, joka määrittää niiden käynnistysjärjestyksen. Käyttäjä voi vaihtaa kattiloiden numeroita haluamallaan tavalla, mikä antaa joustavuutta laitoksen käytössä.

Kun öljykattila käynnistyy, se alkaa lämmittämään kattilassa olevaa vettä asetuslämpötilaan. Kun haluttu lämpötila on saavutettu, öljykattilan venttiili aukeaa, kaukolämpöventtiili aukeaa ja kaukolämpöpumppu käynnistyy. Tämän jälkeen kaukolämpöverkkoon syötetään lämmintä vettä, ja paluuvettä lämmitetään kuumassa kattilassa ennen sen paluuta kaukolämpöverkkoon. Tämä prosessi varmistaa sen, että lämmön siirtyminen tapahtuu tehokkaasti.

Logiikka käyttää seuraavaa kaavaa sammutuskäskyn ymmärtämiseen

$$(T_{SP} - T_{ACT}) \leq T_S$$

, jossa T_{SP} on ”set point” lämpötila eli asetuslämpötila, jota systeemi pyrkii saavuttamaan, T_{ACT} on todellinen lämpötila ja T_S on käyttäjän asettama sammutuslämpötilaero. Jos kaava toteutuu, ohjelma sammuttaa ja sulkee kaikki logiikalla ohjatut laitteet käyttäjän asettaman viiveajan jälkeen. Kuitenkin paineenpitopumppu pysyy koko ajan päällä.

3.2.2 Kommunikointi Liminka BIO -lämpölaitoksen kanssa

Liminka POK -lämpölaitos ja BIO -lämpölaitos kommunikoivat keskenään samassa profinet-verkossa. Tämä yhteys mahdollistaa molempien laitosten logiikoiden ja käyttöliittymien välisen datan lähettämisen ja vastaanottamisen.
Tämä kommunikaatio mahdollistaa sen, että POK -lämpölaitosta voidaan ohjata täysin BIO -lämpölaitoksesta, mikä olikin alun perin tavoitteena. Teoriassa myös BIO -lämpölaitosta voitaisiin ohjata täysin POK -lämpölaitoksesta, mutta tämä ei ollut haluttu ratkaisu.

4 OHJAUSJÄRJESTELMÄ

4.1. Vanhan ohjausjärjestelmän rakenne

Ohjausjärjestelmän vanha versio oli toteutettu Reginin EXOflex EP1011 -logiikalla, ja sen PLC-ohjelma ja käyttöliittymä oli tehty EXOdesigner-ohjelmistolla. Vanhan ohjausjärjestelmän toimintaperiaatetta selvitettiin käyttämällä saatavilla olevia sähkökuvia, PI-kaaviota sekä haastattelemalla käyttäjiä ja tutkimalla vanhaa ohjelmistoa. Kuvassa 17 on vanhan ohjausjärjestelmän sähkökeskus.

Kuva 17. Vanhan ohjausjärjestelmän sähkökeskus

Vanhan ohjausjärjestelmän ymmärtäminen oli erittäin tärkeää, sillä uusi ohjausjärjestelmä suunniteltiin perustumaan vanhaan järjestelmään. Esimerkiksi vanhan ja uuden järjestelmän I/O-määrät olivat melkein samat, joten ymmärtämällä vanhaa järjestelmää voitiin varmistaa, että uusi järjestelmä oli yhteensopiva.

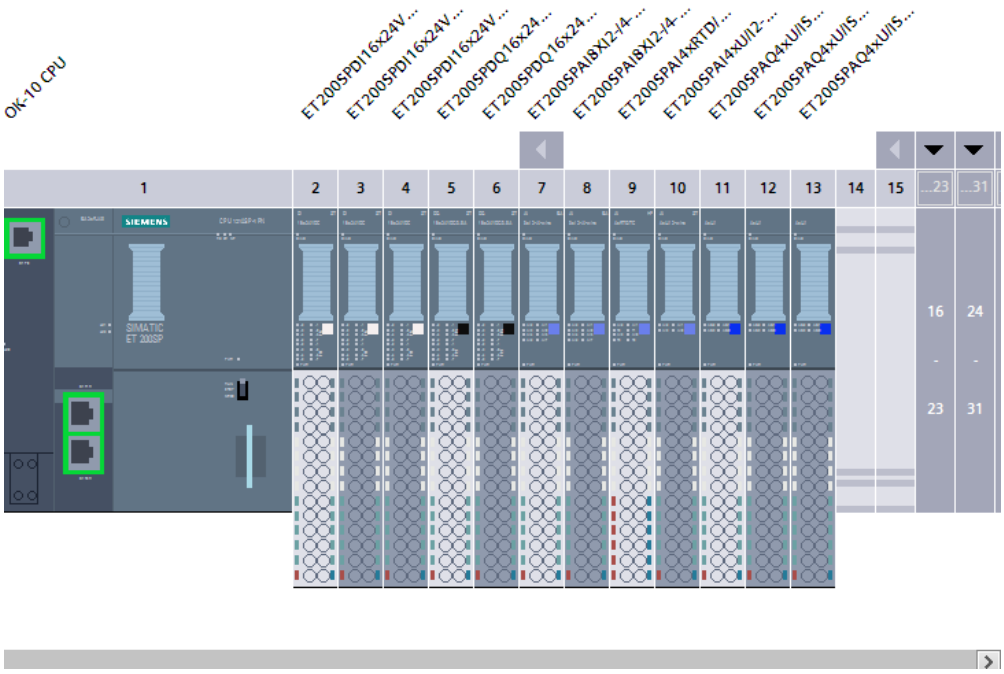
Vanhan ohjausjärjestelmän ohjelmisto oli tehty kokonaan IL ja ST -kielillä. Ohjelmassa ei ollut minkäänlaisia viittauksia, joten muuttujien jäljittäminen ja seuraaminen oli erittäin hankalaa. Kuvassa 18 nähdään vanhan ohjausjärjestelmän ohjelmistolla tehtyä koodia.

```
{ Variables

   R LOC_POR_Teho_Teho              ; Kaukolämpöteho kW
   R LOC_POR_Teho_Energia           ; Kaukolämpöenergia MWh
   L LOC_POR1_Hairio
   L LOC_POR2_Hairio
   L LOC_POR1_Kay_Do
   L LOC_POR2_Kay_Do
   L LOC_KL_PM1_Hairio_Di
   L LOC_POR1_Sekoitus_PM_Di
   L LOC_POR2_Sekoitus_PM_Di

   L LOC_OljySailio1_Valittu_Di
   L LOC_OljySailio2_Valittu_Di
   L LOC_OljySailio1_Vent_Kiinni_Di
   L LOC_OljySailio2_Vent_Kiinni_Di
   R LOC_POR1_VirtausVV_Asento     ;Por2 venttiilin tieto valvomoon
   R LOC_POR2_VirtausVV_Asento           ;Por1 venttiilin tieto valvomoon
   X LOC_Start_POR1_Select               ;0=Seis, 1=Käsi; 2=Auto
   X LOC_Start_POR2_Select               ;0=Seis, 1=Käsi; 2=Auto
   L LOC_StartOrder

   R LOC_RO1_Sailion_PV_LT_Ai ;** Öljylinja 1 lämm.veden lämpötila **
   R LOC_RO1_Sailion_Oljy_LT_Ai ;** Öljylinja 1 öljyn lämpötila **
   R LOC_RO2_Sailion_PV_LT_Ai ;** Öljylinja 2 lämmitysveden lämpötila ***
   R LOC_RO2_Sailion_Oljy_LT_Ai ;** Öljylinja 2 öljyn lämpötila **
   R LOC_KL_MV_LT_Ai ;** KL-menoveden lämpötila **
   R LOC_KL_PV_LT_Ai ;** KL-paluuveden lämpötila **
   R LOC_Ulko_LT_Ai ;** Ulkolämpötila **
   R LOC_POR_1_MV_LT_Ai ;** Kattilan POR 1 menolämpötila **
   R LOC_POR_2_MV_LT_Ai ;** Kattilan POR 2 menolämpötila **
   R LOC_KL_Virtaus_Ai ;** KL-virtaus **

   R LOC_KL_ShunttiVV_Ao
   R LOC_KL_PaluuVV_Ao
   R LOC_POR1_VirtausV_Ao
   R LOC_KL_PM_Ohjaus_Ao

   X LOC_KL_PM_PID_Select
   R LOC_KL_PM_PID_Input
```

Kuva 18. ExoDesignerilla tehty IL -koodinpätkä.

4.2. Uuden ohjausjärjestelmän rakenne

Uusi ohjausjärjestelmä toteutettiin Siemensin tuotteilla. Järjestelmä koostui logiikasta sekä kahdestatoista lisäkortista, jotka toivat lisää tilaa muun muassa tuloille ja lähdöille. Kuvassa 19 näkyy kuvakaappaus laitekokoonpanosta, joka on tehty Siemensin TIA-Portaalissa.



Kuva 19. Ohjausjärjestelmän laitekokoonpano

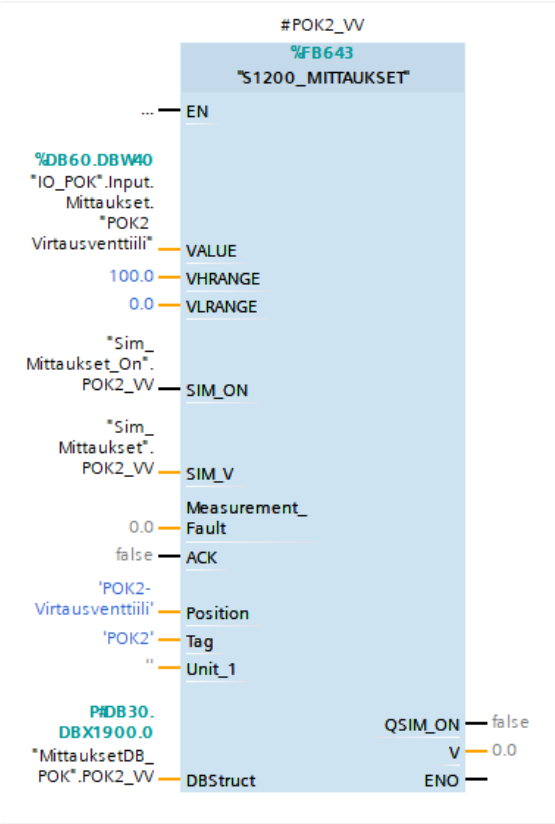
Lisäkorttien valinnassa otettiin huomioon tarvittava IO-määrä tarkastelemalla vanhan ohjausjärjestelmän IO-määrää. Lisäksi varauduttiin tulevaisuutta varten jättämällä kortteihin vapaata IO-kapasiteettia (yhteensä 35 kpl), jolloin mahdolliset uudet laitteet voidaan integroida järjestelmään tarvittaessa. Tämä suunnittelu mahdollistaa joustavuuden ja laajennettavuuden järjestelmän käytön myöhemmässä vaiheessa.

5 PLC OHJELMAN SUUNNITTELU

Ohjelman suunnittelussa käytettiin erilaisia TIA-peruspiirilohkoja, jotka muodostivat valmiin ohjelman rakenteen. Joitakin lohkoja luotiin itse, osa oli valmiita lohkoja Siemensilta, ja osa puolestaan otettiin malliprojektista. Malliprojektista otettuihin lohkoihin tehtiin tarvittavat muutokset, jotta ne soveltuivat tähän kyseiseen projektiin.

5.1. Mittaukset

Mittauspiirikokonaisuus muodostuu logiikkaohjelmalohkosta ja popup -ikkunasta. Piiriä käytetään järjestelmään liitetyille mittaussignaaleille. Mittauspiirin ominaisuuksia ovat mittaukselle aseteltavat hälytysrajat, hystereesi, hälytysviive, sekä mahdollisuus simuloida mittausarvoa. Mittauslohko muodostaa valvomolle rajapinnan. Kuvassa 20 nähdään osa ohjelmassa käytetystä mittauslohkosta. Lohko on toisesta projektista otettu lohko ja siihen on tehty pieniä lisäyksiä DBStructiin. Taulukossa 4 nähdään kaikki lohkossa olevat tulot, lähdöt sekä InOut:it eli ”tulolähdöt”.



Kuva 20. Mittauslohko

TAULUKKO 4. Mittaukset, IN, OUT, InOut

Nimi	Muoto	Kuvaus	Tyyppi
VALUE	INPUT	Analogian tulo	Int
VHRANGE	INPUT	Ylärajahälytyksen raja	Real
VLRANGE	INPUT	Alarajahälytyksen raja	Real
SIM_ON	INPUT	Simuloinnin aktivointi	Bool
SIM_V	INPUT	Simulointiarvo	Real
Measurment_Fault	INPUT	Ei käytössä	Real
ACK	INPUT	Hälytyksen kuittaus	Bool
Position	INPUT	Mittauksen nimi	String[40]
tag	INPUT	Mittauksen positio	String[10]
Unit_1	INPUT	Ei käytössä	String[10]
QSIM_ON	OUTPUT	Simulointi aktiivinen	Bool
V	OUTPUT	Mittausarvo	Real
DBStruct	InOut	Valvomon rajapinta	Mittaus

5.1.1.1. Toiminta

Tämä mittauslohko vastaanottaa raaka-arvon VALUE, joka on kokonaislukumuodossa. Mittauslohko suorittaa raaka-arvon skaalauksen ja näyttää skaalatun arvon V lähtöliitännässä.

Hälytysrajatulot:

- VHRANGE: Haluttu ylärajahälytyksen raja. Tämä raja voidaan kirjoittaa suoraan mittauslohkon tuloliitäntään kokonaislukuna. Suoraan kirjoitettua rajaa ei voi muokata käyttöliittymän kautta.
- VLRANGE: Haluttu alarajahälytyksen raja. Tämä raja voidaan myös kirjoittaa suoraan mittauslohkon tuloliitäntään kokonaislukuna. Samoin kuin ylärajahälytyksen rajan kohdalla, suoraan kirjoitettua rajaa ei voi muokata käyttöliittymän kautta.

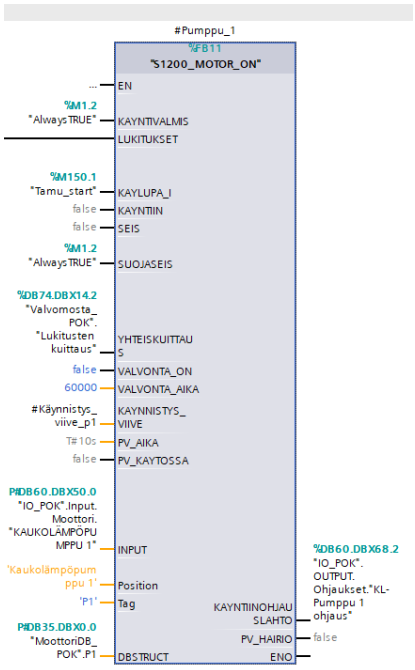
Simulointi:

- SIM_ON: Simulointitulona käytettävä signaali. Kun tämä tulo on kytketty päälle, mittauslohko siirtyy simulaatiotilaan.
- SIM_V: Simuloitu arvo, joka näkyy V-lähtöliitännässä. Tämä arvo otetaan huomioon vain, kun simulaatiotila on aktiivinen (SIM_ON päällä).

5.2. Ohjaukset

5.2.1 Pumppujen ohjaukset

Moottoripiiri muodostuu kolmesta keskeisestä osasta: logiikkaohjelmalohkosta, valvomokuvakkeesta ja popup -ikkunasta. Tämä piiri on suunniteltu käytettäväksi ON/OFF-moottoreiden ohjaamiseen järjestelmässä. Moottoripiirillä on useita ominaisuuksia, jotka mahdollistavat tiedon ja tilan seurannan, tilan valinnan (Manuaali/Automaatti), manuaalisen ohjaustilan moottorin käytön, lukitukset sekä hälytysten muodostamisen ja käsittelyn. Moottorilohko toimii rajapintana valvomolle, joka mahdollistaa kommunikoinnin muiden järjestelmäkomponenttien kanssa. Kuvassa 21 nähdään osa ohjelmassa käytetystä moottorilohkosta. Lohko on toisesta projektista otettu lohko, ja siihen on tehty pieniä lisäyksiä DBStructiin. Taulukossa 5 nähdään kaikki lohkossa olevat tulot, lähdöt sekä InOut:tit.



Kuva 21. Moottorilohko

TAULUKKO 5. Moottori, IN, OUT, InOut

Nimi	Muoto	Kuvaus	Tyyppi
KÄYNTIVALMIS	IN	0=Ei ole valmis 1=Valmis	Bool
LUKITUKSET	IN	0=Lukitsee lohkon	Bool
KÄYLUPA_!	IN	Käynnistyslupa	Bool
KÄYNTIIN	IN	1= Lohkosta startti	Bool
SEIS	IN	1= Lohkosta pysäytys	Bool
SUOJASEIS	IN	1= Ei ole häiriötä 0=Häiriö	Bool
YHTEISKUITTAUS	IN	Hälytyksen kuittaus	Bool
VALVONTA_ON	IN	Lohkon hälytyksien valvonta	Bool]
VALVONTA_AIKA	IN	Hälytyksien valvonta-aika	Dint
KÄYNNISTYS_VIIVE	IN	Käynnistysviive	Time
PV_AIKA	IN	Pyörinnän vartija-aika	Time
PV_KÄYTÖSSÄ	IN	Pyörinnän vartija	Bool
INPUT	IN	Moottorin tulot	Moottori
Position	IN	Moottorin nimi	String[40]
tag	IN	Moottorin positio	String[10]
DBSTRUCT	InOut	Valvomo rajapinta	Moottori1
KÄYNTIINOHJAUSLÄHTÖ	OUT	Moottorin ohjaus	Bool
PV_HAIRIO	OUT	Pyörintävahti häiriö	Bool

5.2.1.1. Toiminta

Tämä moottorilohko kontrolloi yhden moottorin lähtöä.

Moottorin käynnistyslupien ehdot:

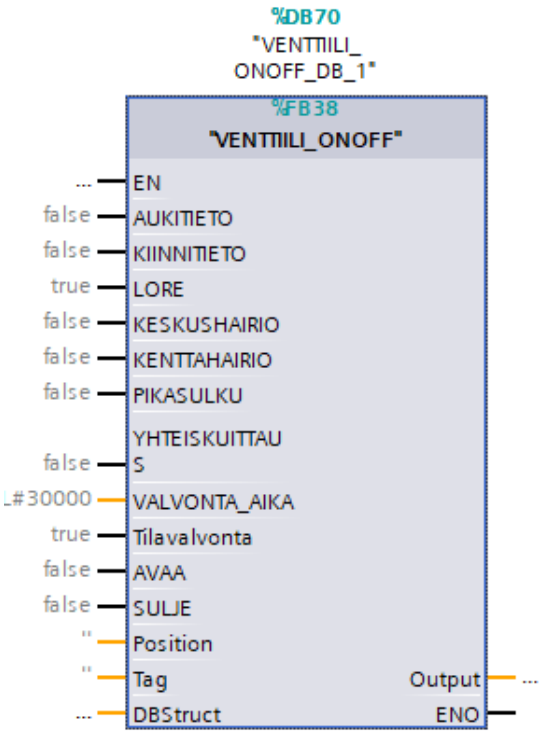
- Moottorin käynnistämiseksi KÄYNTIVALMIS -tulon on oltava aktiivinen (1).
- Moottorin käynnistämiseksi LUKITUKSET -tulon on oltava passiivinen (0).
- Moottorin käynnistämiseksi KÄYLUPA_! -tulon on oltava aktiivinen (1).

Moottorin ohjauskäskyt:

- KÄYNTIIN -tulo käynnistää moottorin, jos käynnistysluvat ovat kunnossa.
- SEIS -tulo pysäyttää moottorin.
- DBSTRUCT:in avulla voidaan käynnistää sekä pysäyttää moottori valvomosta.

5.2.2 Venttiilien ohjaukset

Venttiilipiiri muodostuu kolmesta keskeisestä osasta: logiikkaohjelmalohkosta, valvomokuvakkeesta ja ponnahdusikkunasta. Tämä piiri on suunniteltu käytettäväksi ON/OFF-venttiilien ohjaamiseen järjestelmässä. Venttiilipiirillä on useita ominaisuuksia, jotka mahdollistavat tiedon ja tilan seurannan, tilan valinnan (Manuaali/Automaatti), manuaalisen ohjaustilan venttiilin käytön, lukitukset sekä hälytysten muodostamisen ja käsittelyn. Venttiililohko toimii rajapintana valvomolle, joka mahdollistaa kommunikoinnin muiden järjestelmäkomponenttien kanssa. Kuvassa 22 nähdään osa ohjelmassa käytetystä venttiililohkosta. Lohko on toisesta projektista otettu lohko, jonka DBStructiin on tehty pieniä lisäyksiä. Taulukossa 6 nähdään kaikki lohkossa olevat tulot, lähdöt sekä InOut:tit.



Kuva 22. Venttiililohko

TAULUKKO 6. Venttiili, IN, OUT, InOut

Nimi	Muoto	Kuvaus	Tyyppi
AUKITIETO	IN	Venttiilin tila 1=Auki	Bool
KIINNITIETO	IN	Venttiilin tila 1=Kiinni	Bool
LORE	IN	Local/Remote	Bool
KESKUSHAIRIO	IN	Keskuksessa jokin häiriö	Bool
KENTTAHAIRIO	IN	Kentällä jokin häiriö	Bool
PIKASULKU	IN	1= Sulkee venttiilin	Bool
YHTEISKUITTAUS	IN	Hälytyksen kuittaus	Bool
VALVONTA_AIKA	IN	Hälytyksien valvonta-aika	Dint
TILAVALVONTA	IN	1= Lohko valvoo tilaa	Bool
AVAA	IN	1= Avaa venttiilin	Bool
SULJE	IN	1 = Sulkee venttiilin	Bool
Position	IN	Venttiilin nimi	String[40]
Tag	IN	Venttiilin positio	String[10]
DBSTRUCT	InOut	Valvomo rajapinta	Venttiili1
Output	OUT	Venttiilin ohjaus	Bool

5.2.2.1. Toiminta

Tämä venttiililohko kontrolloi yhden venttiilin lähtöä.

Venttiilin ohjauksen ehdot:

- Venttiilin ohjausta varten muuttujien KESKUSHAIRIO- sekä KENTTAHAIRIO-tulojen on oltava passiivisia (0).
- LORE-tulon ollessa aktiivinen (1), ohjaus on silloin paikallinen eli valvomosta tuleva ohjaus.
- LORE-tulon ollessa passiivinen (0), ohjaus on silloin ohjelmasekvenssien ohjattavana.

Venttiilin ohjauskäskyt:

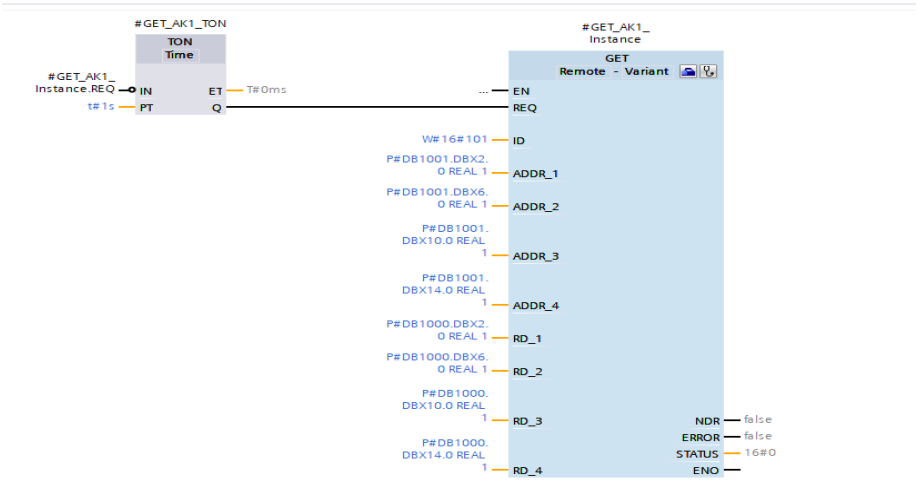
- AVAA-tulo avaa venttiilin, jos venttiilin AUKITIETO ei ole aktiivinen (1), eivätkä mitkään häiriöt ole aktiivisena.
- SULJE-tulo sulkee venttiilin, jos venttiilin KIINNITIETO ei ole aktiivinen (1), eivätkä mitkään häiriöt ole aktiivisena.

5.3. PLC-PLC kommunikointi

Liminka BIO- lämpölaitoksen ja Liminka POK- lämpölaitoksen logiikat pystyvät kommunikoimaan keskenään, koska ne ovat samassa verkossa. Kuitenkin logiikoille pitää kertoa mitä lähetetään ja mitä vastaanotetaan. Tämä datan siirto tehtiin GET- sekä PUT -lohkojen avulla.

5.3.1 GET: Datan luku toisesta logiikasta

Käskyllä "GET" voi lukea tietoja toisesta logiikasta. Käsky käynnistetään, kun ohjaustuloon REQ saapuu positiivinen reuna. Tämän jälkeen relevantit osoittimet alueille, joista dataa luetaan (ADDR_i), lähetetään kumppanilogiikalle. Kumppanilogiikka voi olla joko RUN- tai STOP -tilassa. Kumppanilogiikka palauttaa datan. Jos vastaus ylittää maksimikäyttäjädatan pituuden, se näkyy virhekoodina "2" STATUS-parametrissa. Vastaanotettu data kopioituu konfiguroiduille vastaanottoalueille (RD_i) seuraavalla kutsulla. Tämän toiminnon valmistumisesta ilmoitetaan tilaparametrilla NDR, joka saa arvon "1". Kuvassa 23 nähdään ohjelmassa käytettyä GET- lohkoa.[15]

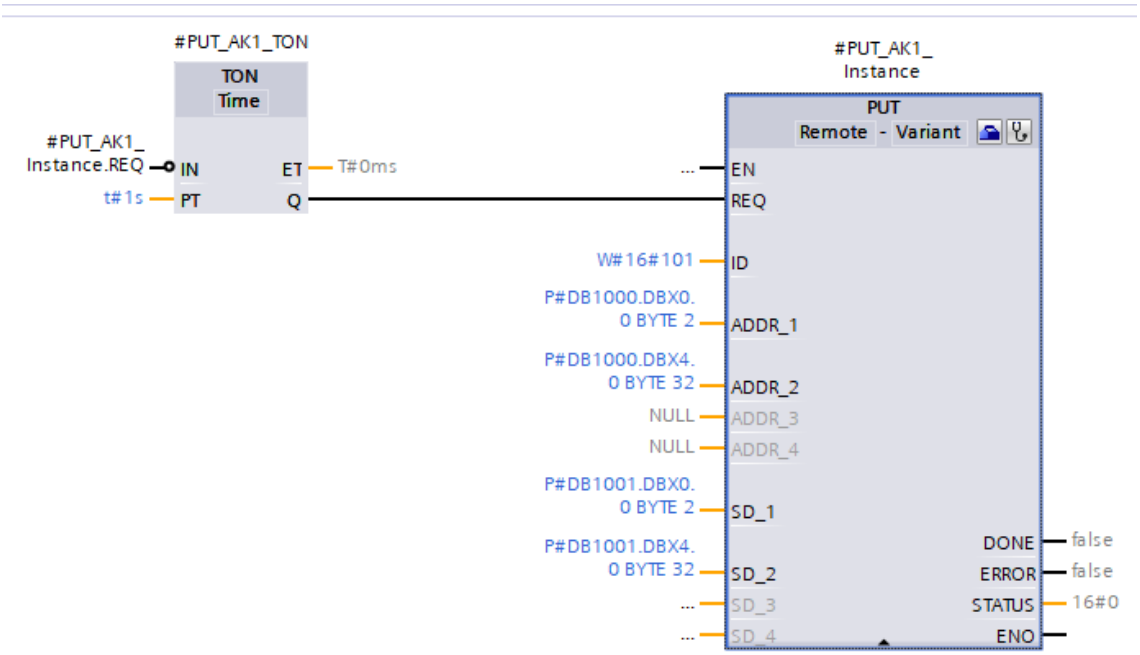


Kuva 23. GET-lohko

5.3.2 PUT: Datan lähettäminen toiseen logiikkaan

Käskyllä "PUT" voidaan kirjoittaa dataa etälogiikalle. Käsky käynnistyy positiivisen reunalta saapuvan ohjaussignaalin REQ avulla. Tämän jälkeen osoittimet kirjoitettaville alueille (ADDR_i) ja data (SD_i) lähetetään kumppanilogiikalle. Kumppanilogiikka voi olla joko KÄYNNISSÄ- tai STOP-tilassa.

Lähetettävä data kopioituu konfiguroiduista lähetysalueista (SD_i), ja kumppanilogiikka tallentaa vastaanotetun datan annettuihin osoitteisiin ja palauttaa suoritusvahvistuksen. Jos virheitä ei ilmene, tämä ilmoitetaan seuraavalla käskykutsulla tilaparametrillä DONE = "1". Kirjoitusprosessi voidaan aktivoida uudelleen vasta, kun viimeinen tehtävä on suoritettu. Kuvassa 24 nähdään ohjelmassa käytettyä PUT- lohkoa. [15]



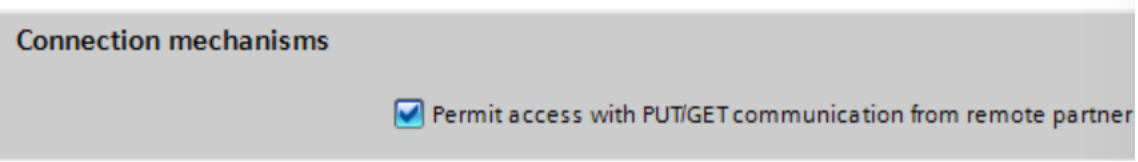
Kuva 24. PUT-lohko

5.3.3 GET & PUT lohkojen käyttöehdot

- Kaksi logiikkaa on käytettävissä ja ne on liitetty toisiinsa PROFINETin kautta. Yhteyttä ei ole vielä konfiguroitu.

- Alhainen suojaustaso kohdassa "<CPU> > Ominaisuudet > Suojaus" varmistaa, että molemmille logiikoille on sallittu lukemis- ja kirjoitusoikeudet.

- Käyttö PUT/GET-käskyllä on sallittu.



Kuva 25. PUT/GET käsky sallittu

6 KÄYTTÖLIITTYMÄN SUUNNITTELU

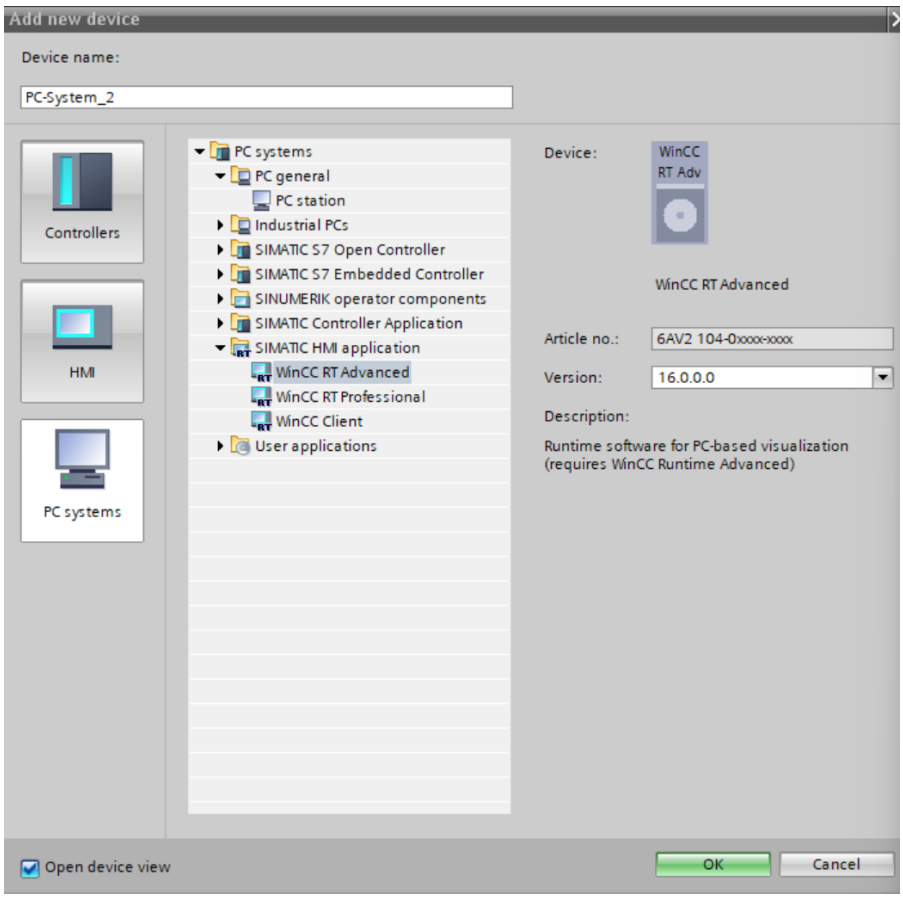
Limingan POK-lämpölaitoksen käyttöliittymän suunnittelu toteutettiin samanaikaisesti logiikkaohjelman kehityksen kanssa. Käyttöliittymä toteutettiin Siemensin WinCC Advanced -ohjelmistolla ja Tia Portal -ohjelmointiympäristössä. Kehitystyö eteni vaiheittain, eli yksi vaihe saatiin valmiiksi aina ennen seuraavaan siirtymistä. Tämä lähestymistapa mahdollisti sujuvan etenemisen ja projektin hallinnan.

Käyttöliittymän suunnittelun eri vaiheet olivat seuraavat:

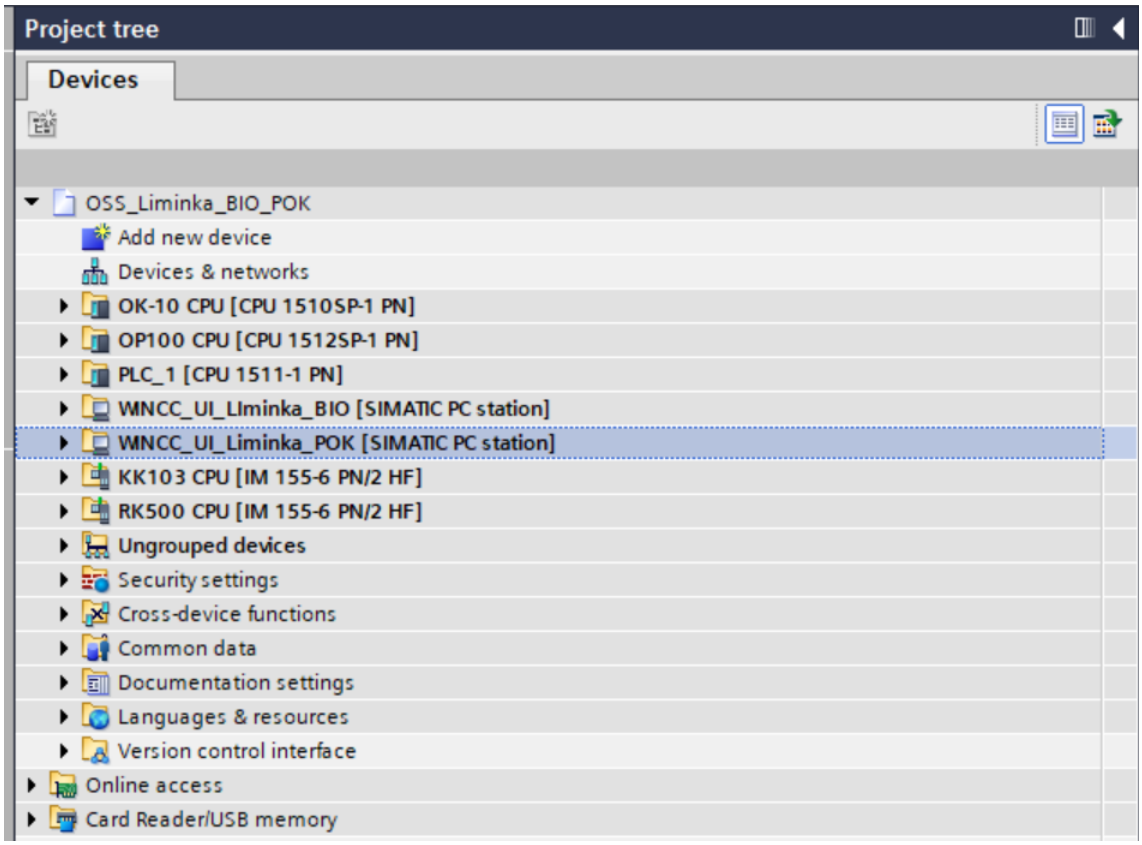
1. Käyttöliittymäprojektin luonti Tia Portal -ohjelmointiympäristössä.
2. Yhteyden luominen logiikan ja käyttöliittymän välillä.
3. Navigoinnin luominen käyttöliittymään.
4. Prosessikuvan piirtäminen.
5. Muuttujien tuominen logiikasta käyttöliittymäympäristöön.
6. Ponnahdusikkunoiden luominen. (Faceplate)

6.1. Käyttöliittymäprojektin luonti Tia Portal -ohjelmointiympäristössä

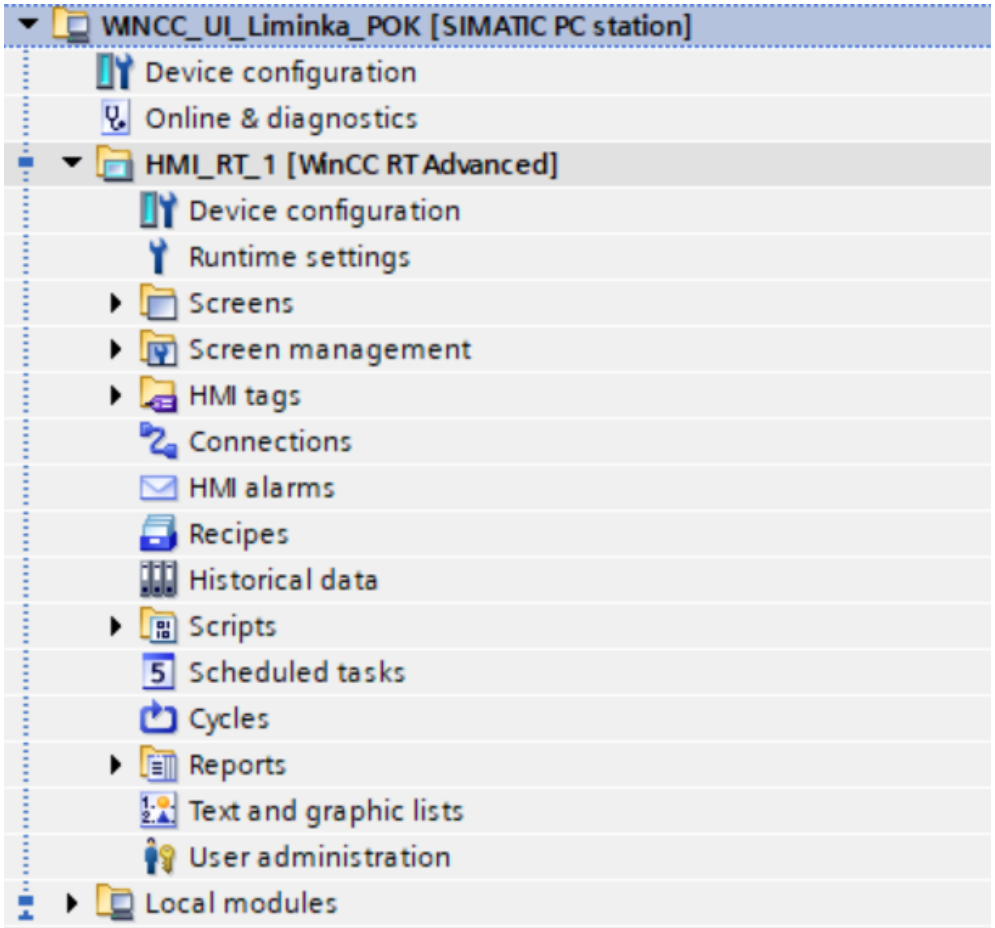
Tia Portal -ohjelmointiympäristössä valittiin oikeat komponentit "Add New Device" -luettelosta. Luettelo on nähtävissä kuvassa 26. Tämän jälkeen Tia Portal loi automaattisesti komponentin kansion "Project Tree" -luetteloon. Kansiosta löytyivät kaikki valittuun komponenttiin liittyvät tiedot. Kuvassa 27 on esitetty "Project Tree" -luettelo, ja kuvassa 28 näkyy Liminka POK -laitoksen käyttöliittymäkansio.



Kuva 26. Add new device -luettelo



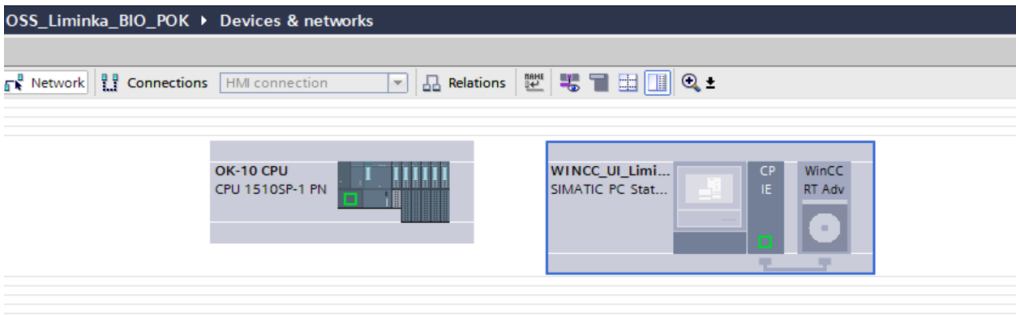
Kuva 27. Project tree -luettelo



Kuva 28. Liminka POK -laitoksen käyttöliittymäkansio

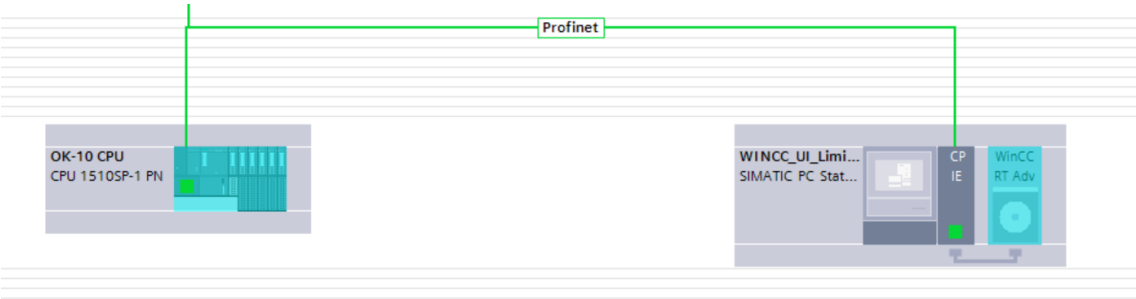
6.2. Yhteyden luominen logiikan ja käyttöliittymän välillä.

Logiikan ja sen lisäkorttien tuonti projektiin toteutettiin samalla menetelmällä kuin käyttöliittymän tuonti projektiin (ks. kohta 6.1). Tämän jälkeen kaikki laitteet näkyivät "Project tree" -luettelossa ja myös "Device & Networks" -näkymässä (Kuva 29).

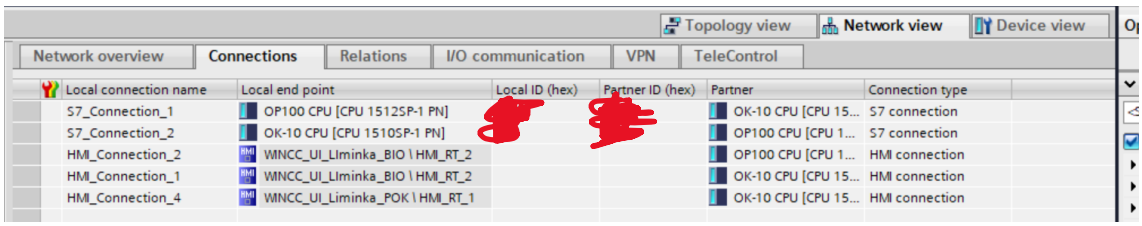


Kuva 29. Device & Networks näkymä.

Kuvassa 29 näkyy, että laitteet on tuotu projektiin, mutta niiden välille ei ole vielä luotu yhteyksiä. Yhteyksien luominen tapahtui yksinkertaisesti määrittelemällä laitteille omat IP-osoitteet ja sitten yhdistämällä ne vetämällä viivat kuvan 30 esittämällä tavalla. Tämän jälkeen oli vielä tehtävä "Network view" -välilehdellä "HMI -connection" kuvan 31 esittämällä tavalla.



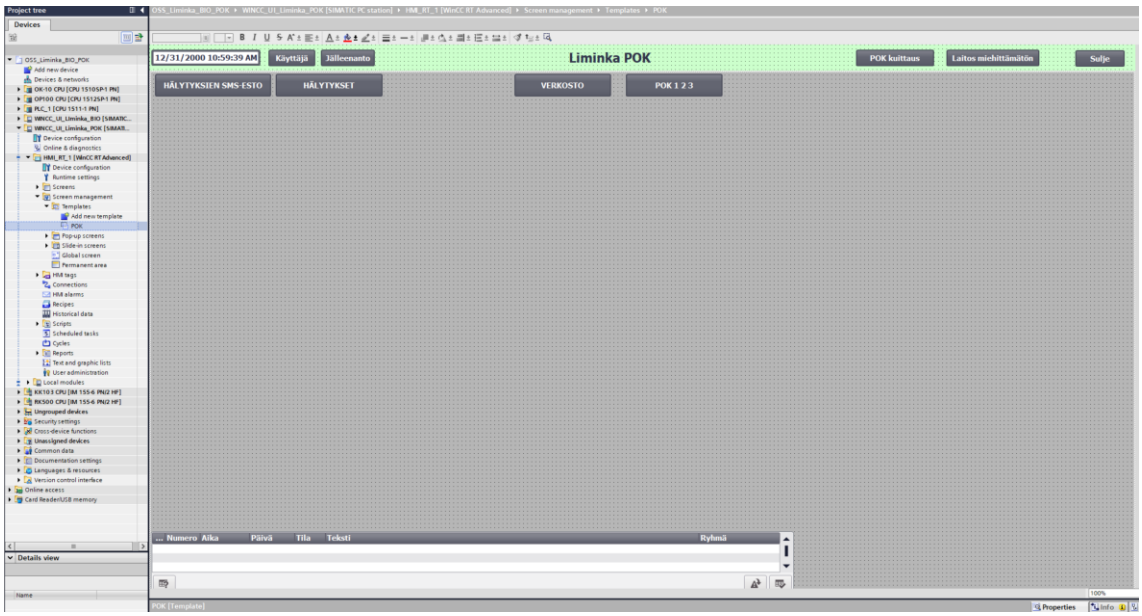
Kuva 30. Profinet yhteys



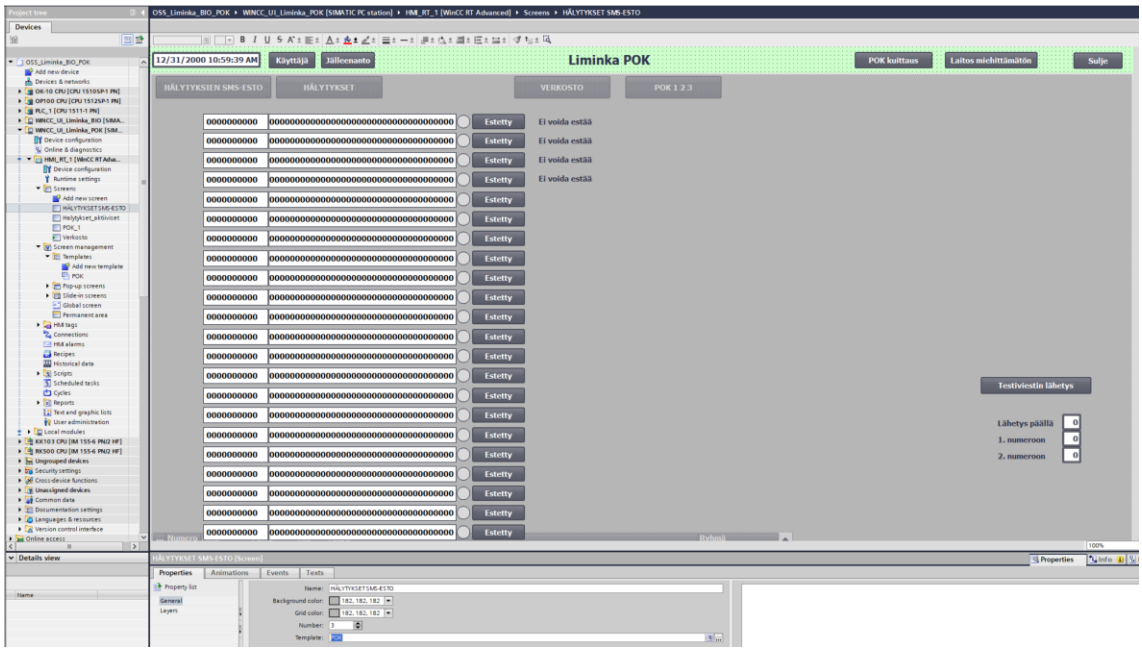
Kuva 31. HMI -Connection

6.3. Navigoinnin luominen käyttöliittymään

Käyttöliittymän navigointi toteutettiin hyödyntäen Tia Portalin näyttöpohjaisten mallien (screen template) toimintoa. Käytännössä luotiin yksi peruspohjanäkymä, johon sisällytettiin kaikki navigointiin liittyvät painikkeet, otsikko, päivämäärä, kellonaika sekä muut käyttäjälle tärkeät elementit, jotka olisi hyvä olla näkyvillä jokaisella sivulla. Tämä peruspohjanäkymä tuotiin jokaiselle projektin sivulle. Kyseinen peruspohjanäkymä on esitetty kuvassa 32, ja sille annettiin nimeksi "POK". Kuvassa 33 on esimerkki siitä, miten POK-peruspohja tuotiin hälytysten SMS-esto-sivulle.

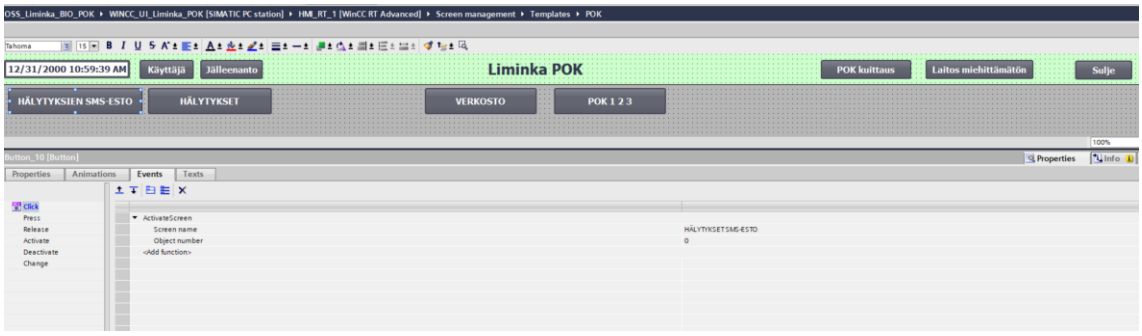


Kuva 32. Peruspohjanäkymä (Screen template)

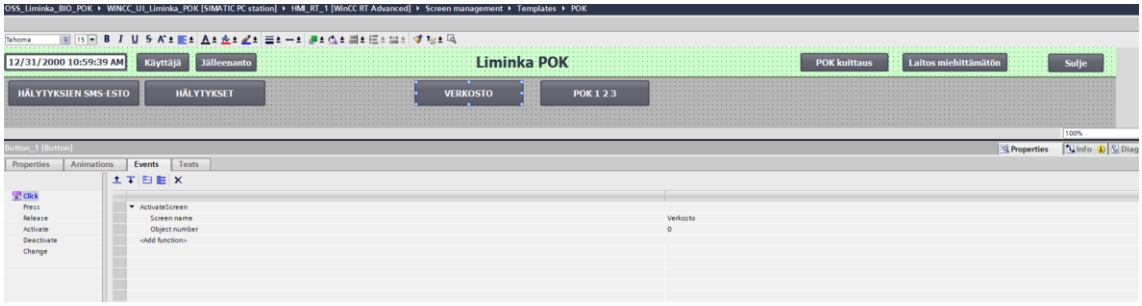


Kuva 33. POK-peruspohja tuonti hälytysten SMS-esto-sivulle

Itse navigointi toteutettiin painikkeiden "events" eli tapahtumien välilehdellä. Siellä luotiin "click" eli klikkaustapahtuma, jossa määriteltiin, että klikkaustapahtuman käynnistyessä tietyn niminen näkymä aktivoituu (esitetty kuvissa 34 ja 35).



Kuva 34. Navigoinnin luominen valitulle napille

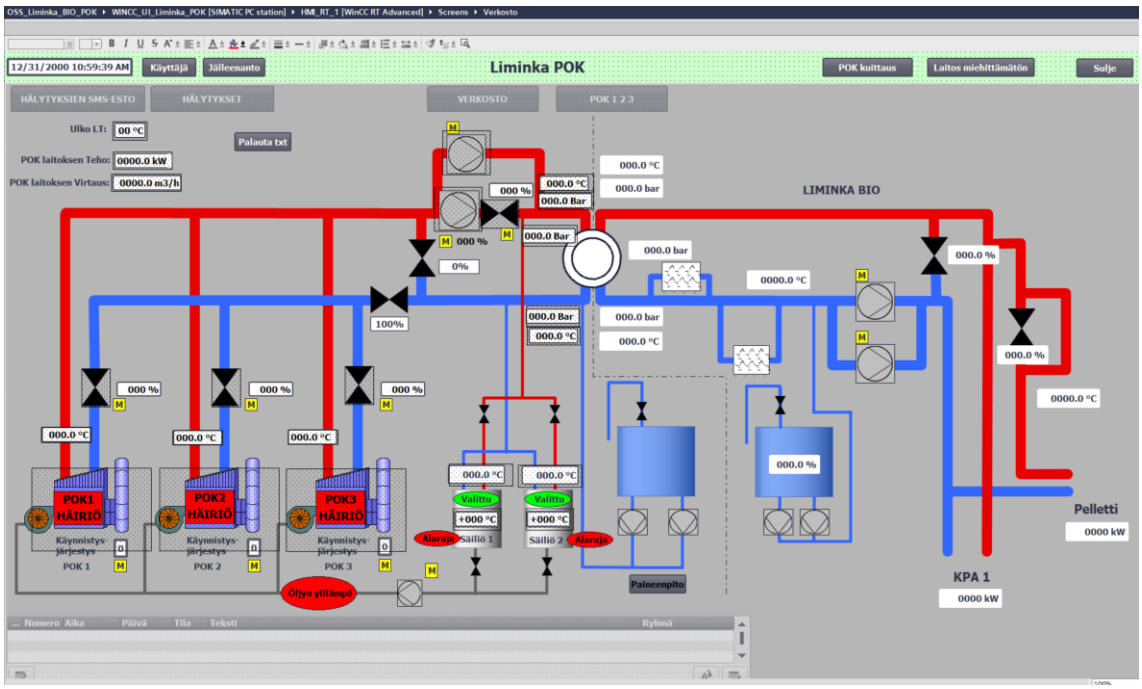


Kuva 35. Navigoinnin luominen valitulle napille

6.4. Prosessikuvan piirtäminen

Liminka POK -lämpölaitoksen prosessikuvan piirtämisessä hyödynnettiin PI-kaaviota. Prosessikuva luotiin käsityönä Tia Portal -ohjelmistossa, jotta se vastaisi mahdollisimman tarkasti PI-kaaviota. Kuitenkin piirustelussa painotettiin käytännöllisyyttä.

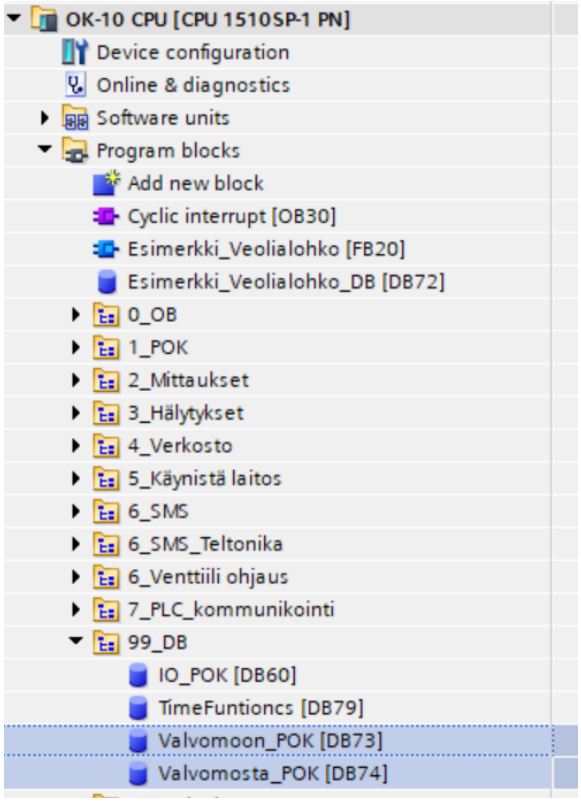
Koska Liminka BIO -lämpölaitoksen toiminta vaikuttaa merkittävästi Liminka POK -lämpölaitoksen automaattiseen toimintaan, tärkeimmät elementit BIO -lämpölaitoksen prosessikuvasta lisättiin myös POK -lämpölaitoksen prosessikuvaan. Kuvassa 36 näkyy Liminka POK -lämpölaitoksen prosessikuva.



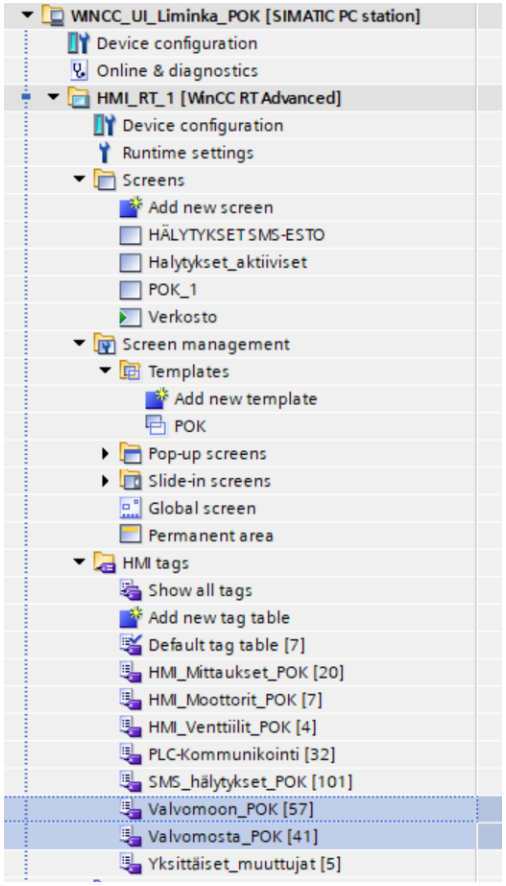
Kuva 36. Liminka POK -lämpölaitoksen prosessikuva

6.5. Muuttujien tuominen logiikasta käyttöliittymäympäristöön

Koska sekä logiikka että käyttöliittymä olivat samassa Tia Portal -projektissa ja niiden välille luotiin yhteys, muuttujien tuominen sujui helposti kopioimalla ja liittämällä. Kuvassa 37 näkyvät esimerkiksi valvomoon_POK ja valvomosta_POK DB:t, jotka ovat logiikan tiedostoyksiköitä. Kuvassa 38 taas näkyvät valvomoon_POK ja valvomosta_POK "HMI tags", jotka ovat käyttöliittymän muuttujia. Nämä muuttujat siirrettiin suoraan logiikasta käyttöliittymään.



Kuva 37. Valvomoon_POK ja valvomosta_POK tiedostoyksiköt logiikassa



Kuva 38. Valvomoon_POK ja valvomosta_POK (HMI tags) käyttöliittymämuuttujat

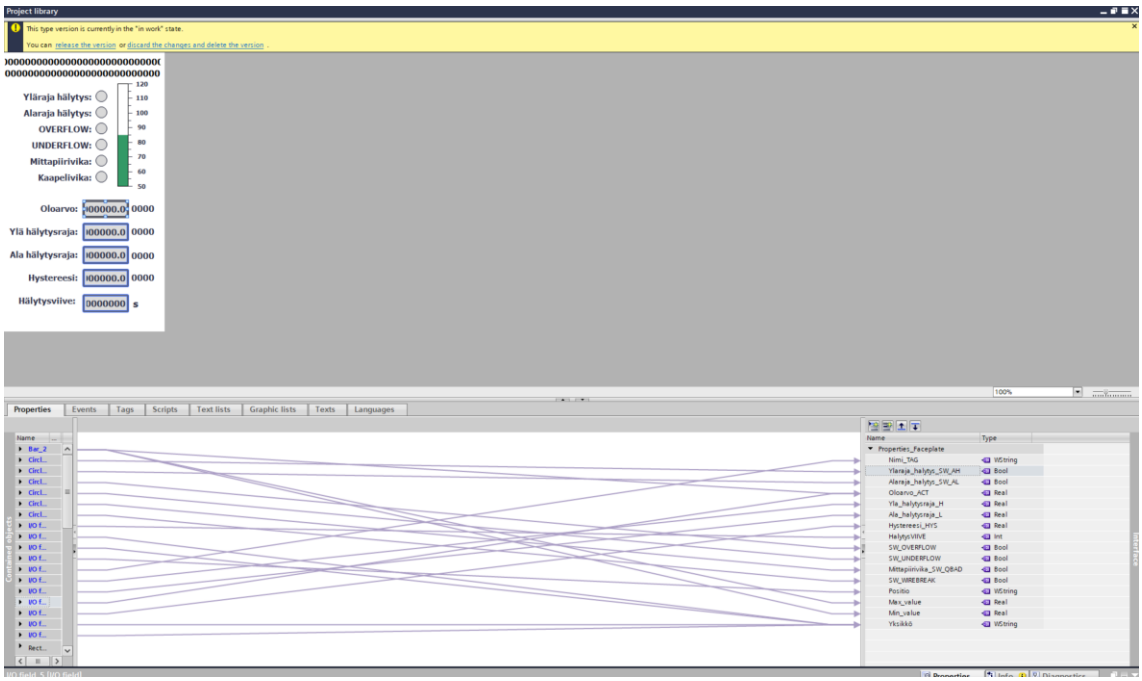
6.6. Ponnahdusikkunoiden luominen (Faceplate)

Ponnahdusikkunoita voitiin lajitella projektissa neljään eri kategoriaan. Taulukossa 7 nähdään kategoriat ja niiden luontimenetelmä.

TAULUKKO 7. Ponnahdusikkuna kategoriat

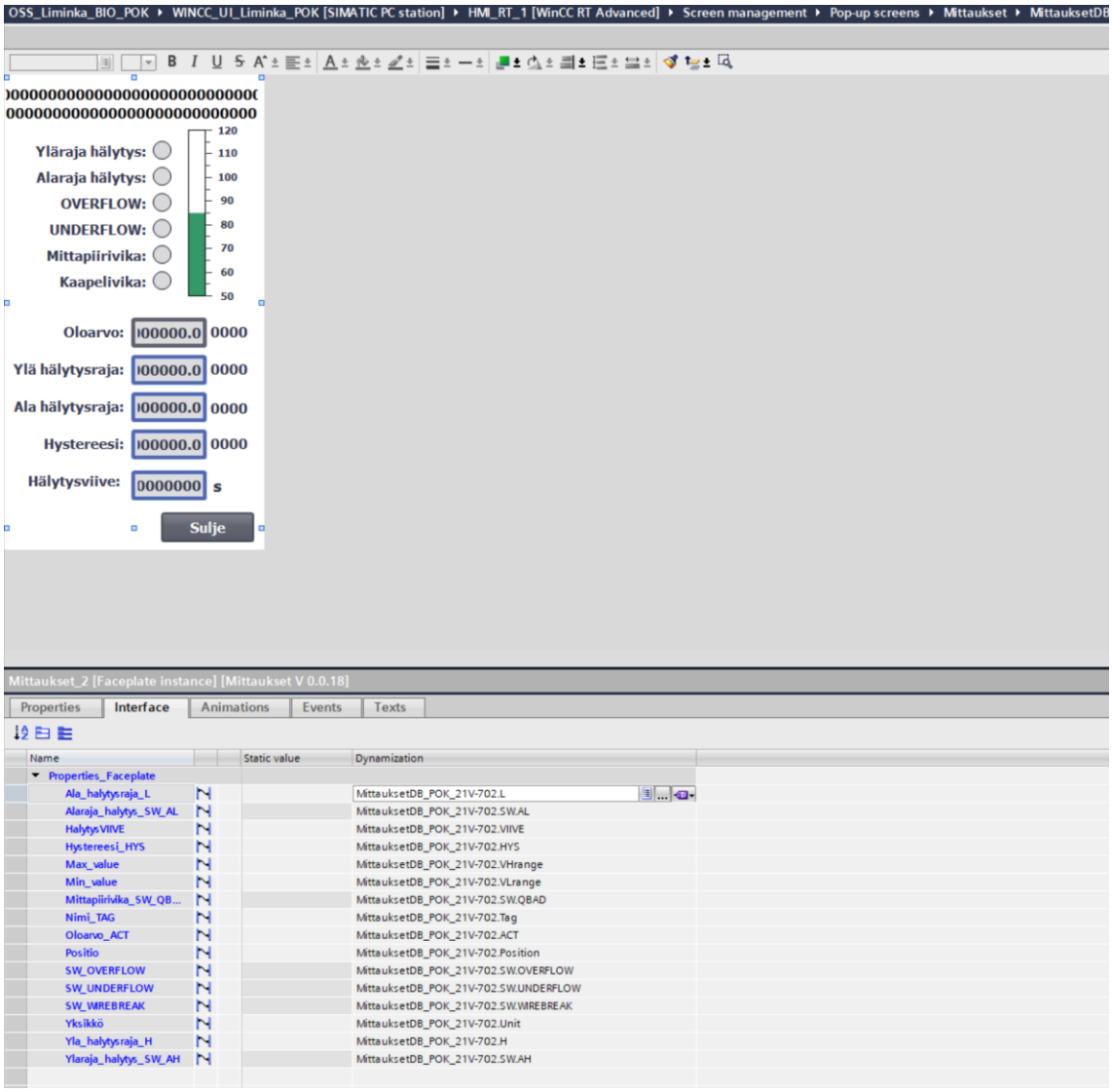
Kategoria	Tyyppi
Mittaus	Faceplate eli perusnäkymä
Moottori	Faceplate eli perusnäkymä
Venttiili	Faceplate eli perusnäkymä
Muut	Yksittäinen ikkuna

Faceplate TIA Portal -ohjelmointiympäristössä oli erinomainen työkalu, jonka avulla voitiin luoda yksi, tarvittaessa uudelleen käytettävä, perusnäkymä. Esimerkiksi näin: projektiin kuuluu 18 mittauspistettä, joille on luotava ponnahdusikkunat. Hyödyntämällä Faceplate-ominaisuutta voidaan luoda vain yksi faceplate eli perusnäkymä mittauksille, kuten kuvassa 39.



Kuva 39. Mittaus perusnäkymän luonti (Faceplate)

Tämän jälkeen voidaan liittää yksittäisten mittauspisteiden muuttujat tähän perusnäkymään. Paljon aikaa ja vaivaa säästyy, kun voi käyttää samaa perusnäkymää kaikkien mittauspisteiden ponnahdusikkunoina; kuten kuvassa 40 on havainnollistettu.



Kuva 40. Mittauspisteen liittäminen

Ponnahdusikkunat mittauksille, moottoreille ja venttiileille luotiin käyttämällä perusnäkymätoimintoa. Kuitenkin on tärkeää huomata, että kaikki ponnahdusikkunat eivät syntyneet samalla tavalla. Tämä johtuu siitä, että taulukon 7 ”Muut” -kategorian ponnahdusikkunoita käytettiin vain kerran. Näissä tapauksissa ei siis ollut tarvetta käyttää samaa perusnäkymää uudelleen.

7 POHDINTA JA YHTEENVETO

Tämän projektin tavoitteena oli suunnitella Liminka POK -lämpölaitokselle uusi automaatiojärjestelmä, joka toimisi lähes samalla tavalla kuin vanha järjestelmä. Työn yhtenä osana oli myös perehtyä TIA-Portal-ohjelmointiympäristöön.

Projekti saavutti asetetut tavoitteet ja tuotti halutut tulokset. Suunniteltu automaatiojärjestelmä läpäisi FAT (Factory Automation Test) -testit ensimmäisellä yrityksellä, mikä yleensä osoittaa, että suunnittelu on toteutettu onnistuneesti. Automaatiojärjestelmä otettiin käyttöön välittömästi, kun käyttöönottovaihe saatiin valmiiksi. Käyttöönottovaihe sujui nopeasti, ja siinä IO-testauksia suoritettiin yhdessä loppuasiakkaan kanssa. Lisäksi varmistettiin, että kaikki toimi suunnitellusti ja odotetulla tavalla.

Opinnäytetyötä tehdessäni tulin entistä tutummaksi Siemensin Tia Portal -ohjelmointiympäristön kanssa, ja se alkoi tuntua entistä mielekkäämmältä. Oli erityisen hienoa havaita, miten voimme ohjelmallisesti käynnistää ja sammuttaa laitteita. Koodissa oleva ohjauslogiikka vastasi laitteiden käyttäytymistä käytännössä erittäin tarkasti ja tavoitellulla tavalla.

Työ saavutti asetetut tavoitteet, ja automaatiojärjestelmä toimi juuri halutulla tavalla. Ohjelmointiprosessi olisi kuitenkin ollut mahdollista toteuttaa myös osittain eri tavalla. Esimerkiksi käytetyt vakiolohkot olivat vasta suunnitteluvaiheessa, ja niissä olisi ollut parantamisen varaa. Selkeämmät kommentit muuttujille ja lohkojen toiminnallisuuksien lisääminen olisivat voineet tehdä koodista vielä selkeämpää ja tehokkaampaa.

Myös Faceplatejen, eli perusnäkymien, luomisessa olisi voinut säästää aikaa käyttämällä datatyyppejä. Esimerkiksi kuvassa 40 huomataan, että "Properties_Faceplate" sisältää kuusitoista muuttujaa, jotka liittyvät yhteen mittaukseen. Tämän olisi voinut tehdä niin, että luodaan yksi datatyyppi, johon liitetään kaikki mittaukseen liittyvät muuttujat. Tällöin olisi tarvinnut tehdä vain yksi linkitys per mittaus, säästäen aikaa ja vaivaa.

Toisin sanoen: vaikka tavoitteet saavutettiinkin, ohjelmointiprosessissa olisi ollut mahdollisuus tehdä joitakin parannuksia (kuten selkeämmät kommentit ja datatyyppien hyödyntäminen), jotka olisivat tehneet järjestelmästä tehokkaamman.

LÄHTEET

[1] AFRY, ”Tietoa meistä | AFRY”. Viitattu: 1. huhtikuuta 2023. [Verkossa]. Saatavissa: https://afry.com/fi-fi/tietoa-meista

[2] Oulun Seudun Sähkö, ”Tietoa Oulun Seudun Sähköstä - Oulun seudun sähkö”. Viitattu: 1. huhtikuuta 2023. [Verkossa]. Saatavissa: https://www.oulunseudunsahko.fi/yritys/oulun-seudun-sahko/tietoa-oulun-seudun-sahkosta.html

[3] Oulun Seudun Sähkö, ”Sähköosuuskunnat - Oulun Seudun Sähkö - Oulun seudun sähkö”. Viitattu: 1. huhtikuuta 2023. [Verkossa]. Saatavissa: https://www.oulunseudunsahko.fi/yritys/oulun-seudun-sahko/sahkoosuuskunnat.html

[4] T. Keinänen ja M. Sumujärvi, *Automaatiotekniikka*, 1. p., vsk. 1. Sanoma Pro Oy, 2019.

[5] polycase, ”What Is a Programmable Logic Controller (PLC)?”, TechTalk Blog. Viitattu: 1. huhtikuuta 2023. [Verkossa]. Saatavissa: https://www.polycase.com/techtalk/electronics-tips/what-is-a-programmable-logic-controller.html

[6] R. J. M. Rao, ”Difference between Compact PLC and Modular PLC”, Inst Tools. Viitattu: 1. huhtikuuta 2023. [Verkossa]. Saatavissa: https://instrumentationtools.com/difference-between-compact-plc-and-modular-plc/

[7] wendelien, ”IEC 61131-3”. Viitattu: 1. huhtikuuta 2023. [Verkossa]. Saatavissa: https://plcopen.org/iec-61131-3

[8] D. H. Hanssen, *Programmable Logic Controllers: A Practical Approach to IEC 61131-3 Using CoDeSys*. New York, UNITED KINGDOM: John Wiley & Sons, Incorporated, 2015. Viitattu: 2. huhtikuuta 2023. [Verkossa]. Saatavissa: http://ebookcentral.proquest.com/lib/tampere/detail.action?docID=4040112

[9] Control Automation, ”Ladder Diagram (LD) Programming Contacts and Coils | Basics of Programmable Logic Controllers (PLCs) | Textbook”. Viitattu: 1. huhtikuuta 2023. [Verkossa]. Saatavissa: https://control.com/textbook/programmable-logic-controllers/ladder-diagram-ld-programming/

[10] J. Faylor, ”PLC Programming Languages: Go Beyond Ladder Logic”, Inductive Automation. Viitattu: 2. huhtikuuta 2023. [Verkossa]. Saatavissa: http://www.inductiveautomation.com/blog/plc-programming-languages-go-beyond-ladder-logic

[11] R. Ait Ali Yahia, ”An Introduction to Statement List (STL) Programming in Siemens TIA Portal”. Viitattu: 4. huhtikuuta 2023. [Verkossa]. Saatavissa: https://www.solisplc.com/tutorials/statement-list-stl-programming-in-siemens-tia-portal

[12] K. Vinod, ”Siemens PLC Programming OBs in Siemens TIA Portal – PLC Tutorial Point”. Viitattu: 5. huhtikuuta 2023. [Verkossa]. Saatavissa: https://www.plctutorialpoint.com/siemens-plc-programming-obs-in-siemens/?utm_content=cmp-true

[13] SIEMENS, ”Which organization blocks can you use in STEP 7 (TIA Portal)? - ID: 40654862 - Industry Support Siemens”. Viitattu: 5. huhtikuuta 2023. [Verkossa]. Saatavissa: https://support.industry.siemens.com/cs/document/40654862/which-organization-blocks-can-you-use-in-step-7-(tia-portal)-?dti=0&lc=en-FI

[14] D. Knapp, ”Difference between FC and FB in Siemens PLC”. Viitattu: 6. huhtikuuta 2023. [Verkossa]. Saatavissa: https://instrumentationblog.com/difference-between-fc-and-fb-in-siemens/
[15] SIEMENS, ”Tiaportal Help”. Siemens.
[16] ”Tietoa kaukolämmöstä”, Rauman Energia. Viitattu: 12. huhtikuuta 2023. [Verkossa]. Saatavissa: https://raumanenergia.fi/kaukolampo/tietoa-kaukolammosta

LIITTEET

Liite 1. Siemensin organisaatioyksiköt S7-1200/1500 sarjan logiikoille Tia-Portaalissa

Taulukko 1. Siemensin organisaatioyksiköt S7-1200/1500 sarjan logiikoille Tia-Portaalissa

Organisaatio-yksiköt	Logiikka	Tapahtumaluokka
OB1	S7-1200/1500	**Pääohjelma** (Päälohko käyttäjäohjelman sykliselle suoritukselle.)
OB100	S7-1200/1500	**Käynnistyslohko** (Kutsutaan kerran, kun CPU:n toimintatila muuttuu STOP-tilasta RUN- tilaan.)
OB20	S7-1200/1500	**Aikaviiveen keskeytyslohko** (Määritetyn ajan jälkeen keskeyttää syklisen ohjelmankäsittelyn.)
OB30	S7-1200/1500	**Syklinen keskeytys** (Keskeyttää syklisen ohjelmankäsittelyn määrätyin aikavälein.)
OB40	S7-1200/1500	**Laitteiston keskeytys** (Keskeyttää syklisen ohjelmankäsittelyn laitteistotapahtuman sattuessa.)
OB82	S7-1200/1500	**Diagnostiikkavirheen keskeytys** (Keskeyttää syklisen ohjelmankäsittelyn diagnostiikkavirheen sattuessa.)
OB80	S7-1200/1500	**Aikavirheen keskeytys** (Keskeyttää syklisen ohjelmankäsittelyn ohjelmasyklin enimmäisajan ylittyessä.)
OB83	S7-1500	**Moduulin kytkeminen tai irrottaminen** (Kutsutaan, kun määritetty ja ei-käytöstä poistettu moduuli tai alimoduuli vedetään tai kytketään.)

OB86	S7-1500	**Telineen tai aseman vika** (Kutsutaan, kun hajautetun periferian isäntäjärjestelmä, orja tai osa alimoduuleista epäonnistuu.)
OB121	S7-1500	**Ohjelmointivirhe** (Kutsutaan, jos ohjelmointivirhe tapahtuu käyttäjäohjelman käskyä käsiteltäessä.)
OB122	S7-1500	**IO-käyttövirhe** (Kutsutaan, jos käyttäjäohjelman käskyä käsiteltäessä tapahtuu virhe IO-datan käytössä.)
OB10-17	S7-1500	**Vuorokaudenaika** (Voidaan käynnistää säännöllisesti tai kerran tiettynä aikana.)
OB55	S7-1500	**Tila** (Kutsutaan, jos tilakeskeytys tapahtuu.)
OB56	S7-1500	**Päivitys** (Kutsutaan, jos päivityksen keskeytys tapahtuu.)
OB57	S7-1500	**Profiili** (Kutsutaan, jos tapahtuu valmistajakohtainen tai profiilikohtainen keskeytys.)

Liite 2. Siemensin organisaatioyksiköt S7-300/400 sarjan logiikoille Tia-Portaalissa

Taulukko 2: Siemensin organisaatioyksiköt S7-300/400 sarjan logiikoille Tia-Portaalissa

Organisaatio-yksiköt	Logiikka	Tapahtumaluokka
OB1	S7-300/400	**Pääohjelma** (Päälohko käyttäjäohjelman sykliselle suoritukselle.)
OB100-102	S7-300/400	**Käynnistyslohko** (Kutsutaan kerran, kun CPU:n toimintatila muuttuu STOP-tilasta RUN- tilaan.)
OB20-23	S7-300/400	**Aikaviiveen keskeytyslohko** (Määritetyn ajan jälkeen keskeyttää syklisen ohjelmankäsittelyn.)
OB30-38	S7-300/400	**Syklinen keskeytys** (Keskeyttää syklisen ohjelmankäsittelyn määrätyin aikavälein.)
OB40-47	S7-300/400	**Laitteiston keskeytys** (Keskeyttää syklisen ohjelmankäsittelyn laitteistotapahtuman sattuessa.)
OB82	S7-300/400	**Diagnostiikkavirheen keskeytys** (Keskeyttää syklisen ohjelmankäsittelyn diagnostiikkavirheen sattuessa.)
OB80	S7-300/400	**Aikavirheen keskeytys** (Keskeyttää syklisen ohjelmankäsittelyn ohjelmasyklin enimmäisajan ylittyessä.)
OB83	S7-300/400	**Moduulin kytkeminen tai irrottaminen** (Kutsutaan, kun määritetty ja ei-käytöstä poistettu moduuli tai alimoduuli vedetään tai kytketään.)

OB86	S7-300/400	**Telineen tai aseman vika** (Kutsutaan, kun hajautetun periferian isäntäjärjestelmä, orja tai osa alimoduuleista epäonnistuu.)
OB121	S7-300/400	**Ohjelmointivirhe** (Kutsutaan, jos ohjelmointivirhe tapahtuu käyttäjäohjelman käskyä käsiteltäessä.)
OB122	S7-300/400	**IO-käyttövirhe** (Kutsutaan, jos käyttäjäohjelman käskyä käsiteltäessä tapahtuu virhe IO-datan käytössä.)
OB10-17	S7-300/400	**Vuorokaudenaika** (Voidaan käynnistää säännöllisesti tai kerran tiettynä aikana.)
OB55	S7-300/400	**Tila** (Kutsutaan, jos tilakeskeytys tapahtuu.)
OB56	S7-300/400	**Päivitys** (Kutsutaan, jos päivityksen keskeytys tapahtuu.)
OB57	S7-300/400	**Profiili** (Kutsutaan, jos tapahtuu valmistajakohtainen tai profiilikohtainen keskeytys.)