SAVONIA
ammattikorkeakoulu

OPINNÄYTETYÖ - AMMATTIKORKEAKOULUTUTKINTO
TEKNIIKAN JA LIIKENTEEN ALA

IMUVAUNUPÖYDÄN OHJAUSAUTOMAATION KEHITYSTYÖ

TEKIJÄ Miika Kalliomäki

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Tutkinto-ohjelma Sähkö- ja automaatiotekniikan tutkinto-ohjelma	
Työn tekijä Miika Kalliomäki	
Työn nimi Imuvaunupöydän ohjausautomaation kehitystyö	
Päiväys 12.5.2023	Sivumäärä/Liitteet 42
Toimeksiantaja/Yhteistyökumppani(t) AirWell Oy	

Tiivistelmä

Työn tavoitteena oli suunnitella ja toteuttaa uusi ohjausautomaatio AirWell Oy:n valmistamalle imuvaunupöydälle. Imuvaunupöytä on leikkauspöydän alla tai yhteydessä oleva rata, jossa imuvaunu liikkuu leikkauskoneen liikkeiden mukaisesti ja poistaa leikkaamisesta syntyvät savukaasut. Imuvaunupöydän vanha ohjausautomaatio oli suunniteltu ohjaamaan imuvaunupöytää, jossa työskentelee vain yksi imuvaunu kerrallaan. Tässä opinnäytetyössä tarkasteltiin, kuinka kaksi imuvaunua saadaan toimimaan samanaikaisesti yhdellä imuvaunupöydällä.

Opinnäytetyö oli kehitystyö. Työ aloitettiin tutustumalla aikaisempaan ohjausautomaatioon ja siinä käytettyihin Omronin ohjainlaitteisiin. Lisäksi työn teoriaosuudessa käytiin läpi ohjelmoitavia logiikoita yleisellä tasolla. Teoriaosuuteen käytetiin lähteinä aiheeseen liittyviä oppikirjoja sekä laitevalmistajien omia oppaita.

Työn tuloksena syntyi kahden imuvaunun imuvaunupöydälle toimiva ohjausautomaatio. Uusi ohjausautomaatio ja logiikkaohjelma saatiin siis kokonaisuudessaan suunniteltua ja otettua käyttöön, eli tavoitteisiin päästiin. Ohjausautomaatio jää yrityksen vakituiseen käyttöön, sillä sitä pystyy tarvittaessa käyttämään myös yhden imuvaunun pöydällä pienellä logiikkaohjelman muutoksella.

Avainsanat

Ohjelmoitavat logiikat, ohjainlaitteet, ohjausautomaatio, logiikkaohjelma

SAVONIA UNIVERSITY OF APPLIED SCIENCES THESIS
Abstract

Field of Study Technology, Communication and Transport	
Degree Programme Degree Programme in Electrical and Automation Engineering	
Author Miika Kalliomäki	
Title of Thesis Development of a fume extraction table's control system	
Date May 12, 2023	Pages/Appendices 42
Client Organisation /Partners AirWell Oy	

Abstract

The purpose of this thesis was to design and develop a new control system for a fume extraction table. Fume extraction table is a certain kind of track below a cutting table. Suction unit moves on the track following to the movements of cutting machine and extracts fumes coming from the cutting. Client Organisation for this thesis was AirWell Oy and this subject came up when the company were asked to deliver a fume extraction table with two suction units. The previous control system was designed to control only one suction unit at a time so it couldn't be used for this kind of extraction table.

Methods of getting this thesis started was getting to know the previous control system and used automation components. Programmable logic controllers and especially Omron automation systems were also reviewed in the theory part of the thesis. Sources for theory part were found on subject related textbooks and manuals made by manufacturers.

As a result of this thesis, a fully functional control system for a fume extraction table equipped with two suction units were designed and put into use. Objectives were achieved and the control system will remain in the company's regular use as it can also be used on the table of one suction unit with a small change in the logic program.

Keywords

programmable logic controllers, control systems, automation systems, logic program

SISÄLTÖ

1 JOHDANTO 7

1.1 Tilaajan esittely 7

1.2 Työn esittely 7

2 OHJELMOITAVAT LOGIIKAT 8

2.1 Logiikan rakenne ja toiminta 9

2.2 Logiikan ohjelmointi 10

3 OMRON OHJELMOITAVAT LOGIIKAT 13

3.1 Omron CP1L 13

3.2 Omron CX-One ohjelmisto 15

4 TYÖN TOTEUTUS 21

4.1 AW-Imuvaunupöytä 21

4.2 Työn lähtökohdat 21

4.3 Projektin suunnittelu 22

4.4 Komponenttien valinta 23

4.5 Ohjelmointi 26

4.6 Tiedonsiirto logiikoiden välillä 32

4.7 Testaaminen 33

4.8 Käyttöönotto 36

5 YHTEENVETO 40

KUVALUETTELO

Kuva 1. PLC rakenne (Automaatiotekniikan perusteet, 2022) 8
Kuva 2. Logiikkaohjelman kiertosykli (Ohjelmoitavat logiikat, 2004) 9
Kuva 3. Tikapuukaavio (Programmable logic controllers, 2009) 10
Kuva 4. Toimilohkokaavio (Opas toimilohko-ohjelmointiin, 2011) 11
Kuva 5. Sekvenssivuokaavio (Automaatiojärjestelmien logiikat ja ohjaustekniikat, 2007) 11
Kuva 6. Strukturoitu teksti (CX-One ja logiikkaohjelmointi, 2009) 12
Kuva 7. Omron ohjelmoitavat logiikat (Omron, 2023) 13
Kuva 8. Omron CP1L (Omron, 2023) 14
Kuva 9. CX-Programmer ohjelman alkunäyttö 15
Kuva 10. Logiikan lisääminen 16
Kuva 11. CX-Programmer ohjelmointinäkymä 17
Kuva 12. Uuden käskyn lisääminen 17
Kuva 13. Logiikan lisätoiminnot 18
Kuva 14. Symbolin lisääminen (CX-One ja logiikkaohjelmointi, 2009) 18
Kuva 15. Toimilohkon ohjelmointi 19
Kuva 16 CX-Simulator esimerkki (CX-Simulator Introduction guide, 2008) 19
Kuva 17 Vertailu esimerkki (CX-One ja logiikkaohjelmointi, 2009) 20
Kuva 18. Esimerkkikuva imuvaunusta (AirWell Oy, 2023) 21
Kuva 19. ME90S Sähkömoottori (Moves Oy, 2023) 23
Kuva 20. Vacon 20- taajuusmuuttaja (Vacon Oy, 2023) 23
Kuva 21. Duelco NST-3.2 turvarele (Duelco Oy, 2023) 24
Kuva 22. Lika Electronic I58S pulssianturi (Lika electronics, 2023) 25
Kuva 23. Omron S8VK virtalähde (Omron Oy, 2023) 25
Kuva 24. Instructions reference 26
Kuva 25. Symbolilista 1/3 27
Kuva 26. Symbolilista 2/3 27
Kuva 27. Symbolilista 3/3 27
Kuva 28. Eteen ja taakse liikkeet 28
Kuva 29. Ajonopeudet 28
Kuva 30. Imuvaunun eteenpäin liike automaattiajolla 29
Kuva 31. Pikalaskuri 29
Kuva 32 Pulssianturin nollaaminen 29

Kuva 33. Ajonesto....30
Kuva 34. Pulssiantureiden yhteenlasku....31
Kuva 35. Imuvaunujen törmäyssuoja....31
Kuva 36. Network Receive komento....31
Kuva 37. Tarvittavat parametrit network receive -komennolle....32
Kuva 38. Logiikan asetukset....33
Kuva 39. Ohjauskaappi testausvaiheessa....34
Kuva 40. Imuvaunun kauko-ohjain....35
Kuva 41. Moottori, vaihdelaatikko ja pulssianturi testipenkissä....35
Kuva 42. Imuvaunupöytä....37
Kuva 43. Imuvaunun moottori, vaihdelaatikko ja pulssianturi....37
Kuva 44. Leikkauskoneen pulssianturi....38
Kuva 45. Imuvaunun rajakytkin....38
Kuva 46. Ohjauskaappi....39

1 JOHDANTO

Tässä opinnäytetyössä käsitellään imuvaunupöydän ohjausautomaation suunnittelu- ja kehitystyötä. Työn teoriaosuudessa perehdytään ohjelmoitaviin logiikoihin ja ohjausjärjestelmiin. Työn toteutuksessa käytettiin Omronin ohjausjärjestelmiä, joten niitä käsitellään yksityiskohtaisemmin.

1.1 Tilaajan esittely

AirWell Oy on Sastamalan Kiikoisissa sijaitseva termiseen leikkausautomaatioon erikoistunut yhtiö, joka on perustettu vuonna 1993. Yritys tarjoaa monipuolisia ratkaisuita leikkaustarpeisiin automatisoitujen vesi-, laser-, plasma- ja polttoleikkaus laitteiden ja koneiden muodossa. AirWell valmistaa plasma- ja polttoleikkauskoneita omalla WellCut tuotemerkillään, sekä tarvittaessa yksilöllisiä koneita täysin asiakkaan toiveiden mukaan. Yrityksen toimintaan kuuluu vahvasti myös leikkauskoneiden peruskunnostus ja huolto.

Leikkausautomaation lisäksi AirWell tarjoaa useita erilaisia ratkaisumalleja myös leikkauksessa syntyvän savukaasun imuun ja suodatukseen. Yritys valmistaa lohkoimupöytiä ja imuvaunupöytiä, joista jälkimmäisen ohjausautomaation kehittämiseen tämän opinnäytetyön aihekin liittyy.

1.2 Työn esittely

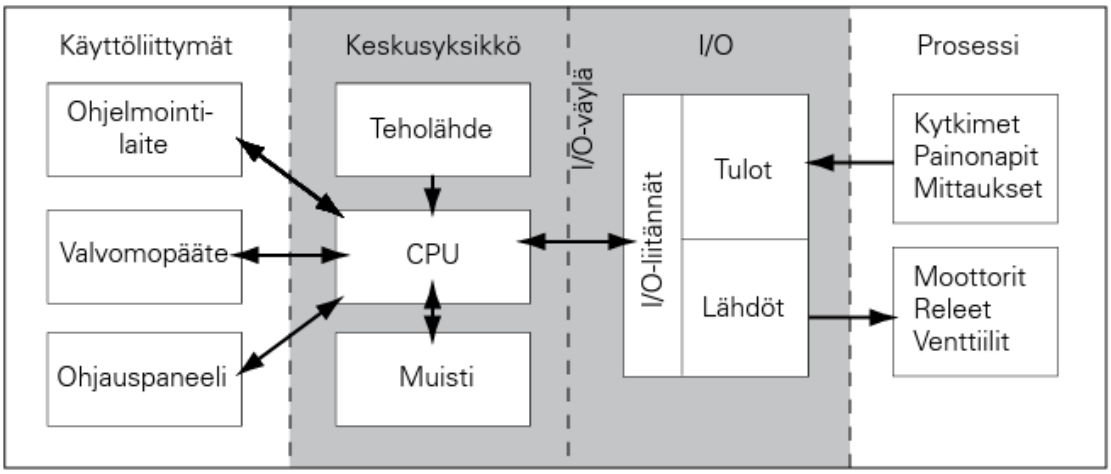
Tämän opinnäytetyön aiheena on AirWell Oy:n käyttämän imuvaunun ohjausautomaation kehitystyö. Tavanomaisessa tilanteessa AirWell valmistaa imuvaunupöydän, jossa on yksi imuvaunu. Imuvaunun on määrä liikkua leikkauspöydän alle sijoitetulla radalla automatisoidusti leikkauskoneen liikkeitä seuraten. Tällä tavalla imuvaunulla saadaan tehtyä tehokas kohdepoisto leikkaussavulle ja samalla kerättyä leikkauksesta syntyvät jätteet helposti tyhjennettävään roskavaunuun.

Opinnäytetyön aihe syntyi, kun AirWell sai pyynnön toimittaa imuvaunupöytä, joka asennettaisiin asiakkaan omaan, kooltaan erittäin suuren leikkauspöydän yhteyteen. Uudenlaisen ja haastavan tilanteesta teki se, että asiakkaan pöydällä työskentelee yhden sijaan samanaikaisesti kaksi leikkauskonetta, joten tarve on myös kahdelle imuvaunulle. Tässä tilanteessa nykyistä ohjausautomaatiota ei pystytä enää käyttämään. AirWellin nykyinen ohjausautomaatio on suunniteltu ohjaamaan vain yhtä imuvaunua yhdellä imuvaunupöydällä. Työn tavoitteena on siis kahden samalla pöydällä, ja samanaikaisesti toimivien alaimuvaunujen ohjausautomaation suunnittelu ja toteutus.

2 OHJELMOITAVAT LOGIIKAT

Ohjelmoitavat logiikat ovat saaneet nimensä englannin kielisestä vastineesta programmable logic controller, eli PLC. Ohjelmoitavat logiikat ovat yhdellä tai useammalla mikroprosessorilla varustettuja laitteita, joilla pystytään ohjaamaan erilaisten prosessien komponentteja. Ohjaaminen tapahtuu logiikoiden tulojen ja lähtöjen, tai väylien kautta niissä muistissa olevien ohjelmien ja asetettujen parametrien perusteella. Ohjauksien asettaminen ja muuttaminen tapahtuu sovellusohjelmaa muokkaamalla. (Omron, 2009)

Ohjelmoitavat logiikat ovat oleellinen osa automaatiojärjestelmiä. Yleisesti logiikan tuloihin kytketään ohjaukseen käytettäviä komponentteja kuten antureita ja kytkimiä, kun taas lähtöihin ohjattavia toimilaitteita kuten merkkilamppuja tai magneettiventtiileitä. PLC-laitteita kutsutaan myös vapaasti ohjelmoitaviksi logiikoiksi, sillä niiden ohjelmoinnissa on vapaus valita kirjoitusjärjestys ja ohjelmointikieli. Ohjelmointi voitaisiin teoriassa tehdä esimerkiksi LCD-näytön tai toimintopainikkeiden kautta, mutta nykyään ohjelmointiin käytetään lähes aina tietokoneella käytettävää ohjelmointisovellusta. Tehty sovellusohjelma tallennetaan paristovarmistettuun ohjelmamuistiin ja jää määrittämään logiikan tehtäviä prosessin aikana. (Automaatiojärjestelmien logiikat ja ohjaustekniikat, 2007)



Kuva 1. PLC rakenne (Automaatiotekniikan perusteet, 2022)

Halutessaan käyttäjä pystyy seuraamaan ohjelmoitavan logiikan toimintojen kulkua esimerkiksi ohjauspaneelin tai suuremmissa kokonaisuuksissa valvomopäätteen kautta. Toimintoihin pystytään tarvittaessa myös vaikuttamaan ohjauspaneelin, tai esimerkiksi toimintokytkimien ja näppäimistön avustuksella. Aina ei kuitenkaan ohjelmointilaitteen lisäksi erillisiä seurantalaitteita tarvita, ja jokainen automaatiokokonaisuus toteutetaankin aina kyseisen kokonaisuuden tarpeiden mukaan. (Automaatiotekniikan perusteet, 2022)

Ohjelmoitavien logiikoiden valmistajia on nykyään useita. Vaikka PLC:n perusperiaate pysyykin samana, on laitteiden ja käyttöliittymien välillä myös eroavaisuuksia. Tämän opinnäytetyön automaatioprojektissa käytetään Omronin valmistamia logiikoita, joten niihin perehdytään tarkemmin kappaleessa 3.

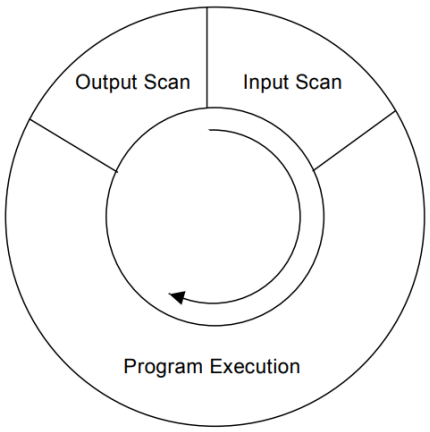
2.1 Logiikan rakenne ja toiminta

Logiikan sisäistä toimintaa ja viestiliikennettä oheislaitteisiin ohjaa mikroprosessori eli CPU, muistipaikat ja käyttöjärjestelmä. Suuremmissa logiikoissa laajojen ohjausten jouhevan käytön varmistamiseksi mikroprosessoreja saattaa olla useampiakin. Logiikat sisältävät ROM-muistia, johon esimerkiksi käyttöjärjestelmä on tallennettuna. Sovellusohjelmat ovat tallennettuna paristovarmennettuun RAM-muistiin tai Flash-ROM-muistiin, josta se siirretään automaattisesti RAM- muistiin käynnistettäessä (CX-One ja logiikkaohjelmointi 2009)

Logiikan liityntä prosessin kenttälaitteisiin tapahtuu tulojen ja lähtöjen kautta. Yksinkertaistetusti sanottuna, logiikkaohjelma säätää lähtöjä tulojen mukaan. I/O-liitäntäpaikat voivat olla kiinteästi integroituna logiikkaan, tai sijaita erillisinä tulo- ja lähtömoduuleina logiikan läheisyydessä. Kiinteiden I/O paikkojen logiikoita kutsutaan myös kompakteiksi logiikoiksi, ja erillisiä I/O moduuleita käyttäviä logiikoita modulaarisiksi logiikoiksi. Modulaariset logiikat ovat helpommin muunneltavissa mutta vaativat usein erillisen tehonlähteen (PSU). (CX-One ja logiikkaohjelmointi 2009)

Tulot ja lähdöt voidaan jakaa vielä kahteen eri tyyppiin, digitaalisiin ja analogisiin. Digitaaliset tulot ja lähdöt ovat binäärisiä, eli voivat olla muodoltaan vain 0 tai 1. Analogiset tulot ja lähdöt taas ovat portaattomia signaaleita, jotka vaihtelevat esimerkiksi lämpötilan tai paineen mukaan. (Automaatiolaitteet, 1996)

Lähtökohtaisesti logiikat toimivat syklisellä ohjelmankäsittelyperiaatteella. Tämä tarkoittaa sitä, että yhden syklin aikana tutkitaan CPU:n ja oheislaitteiden tilaa, suoritetaan RAM-muistiin tallennettu sovellusohjelma rivi kerrallaan ja päivitetään tulot ja lähdöt sekä mahdolliset muut väylät ohjelman mukaisesti. Logiikkaohjelmasta riippuen syklistä saatetaan kuitenkin poiketa, esimerkiksi keskeytyskäskyjen vuoksi. (Ohjelmoitavat logiikat, 2004)

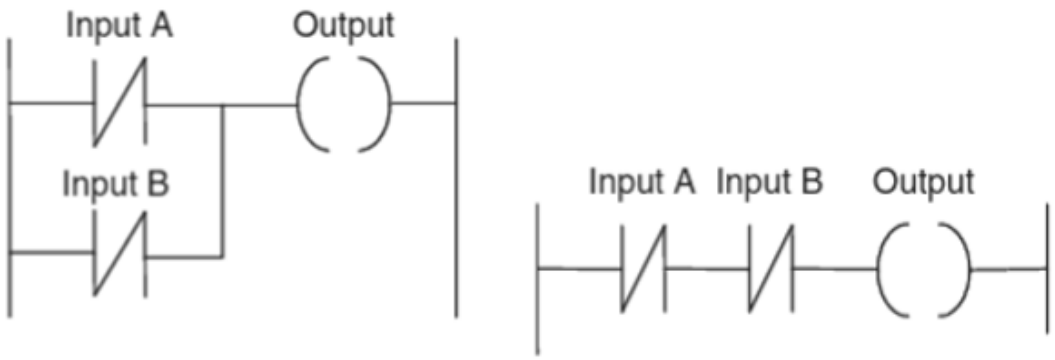


Kuva 2. Logiikkaohjelman kiertosykli (Ohjelmoitavat logiikat, 2004)

2.2 Logiikan ohjelmointi

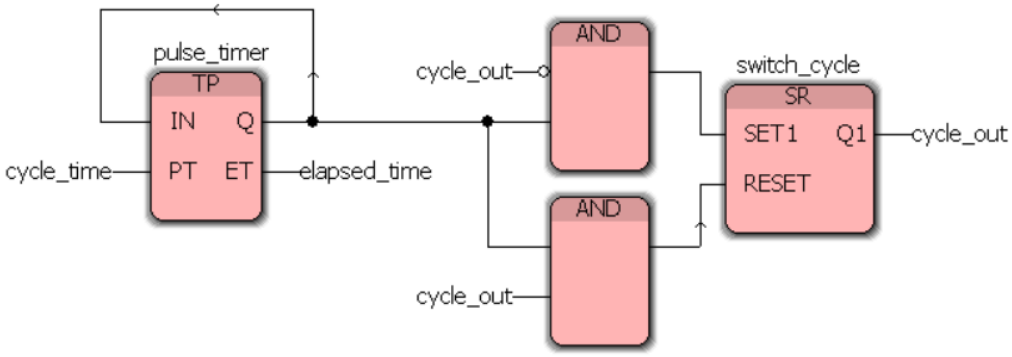
Aikaisemmin jokaisella ohjelmoitavan logiikan valmistajalla on ollut käytössään oma ohjelmointiohjelmansa ja menetelmänsä. Tämän myötä ohjelmoinnista tuli hyvin monimuotoista ja käytettävän logiikan valintaperusteena saattoi olla pelkästään ohjelmointitavan tunteminen ennestään. Tästä syystä International Electrotechnical Commission (IEC) on laatinut vuonna 2013 uusimman version PLC ohjelmointistandardista 61131–3, joka määrittää käytettäviksi ohjelmointikieliksi: Sequential Function Chart (SFC), Structured Text (ST), Function Block Diagram (FBD), Ladder Diagram (LD) ja Instruction on List (IL). Näistä niin sanottuja graafisia editoreita ovat SFC, FBD ja LD. Tekstieditoireiksi lasketaan ST ja IL ohjelmointikielet. (Automaatiojärjestelmien logiikat ja ohjaustekniikat, 2007) (IEC 61131-3 standardi, 2013)

Näistä perinteisin ja yksinkertaisin ohjelmointikieli lienee suomalaisittain sanottuna tikapuukaavio (LD), joka muistuttaa pitkälti perinteisten releohjauksien piirikaavioita. Kaavio sisältää kaksi pystysuoraa viivaa, jotka kuvaavat ikään kuin sähkönsyöttöä. Niiden välille rakennetaan ehdot, joiden täyttyessä ulostulo aktivoituu. Kaaviota luetaan loogisesti vasemmalta oikealle, ylhäältä alaspäin ja ”askelmat” erotetaan toisistaan vaakasuorilla viivoilla. Kyseisen ohjelmointikielen yksinkertaisuuden ansiosta, tikapuukaavion käyttö on edelleenkin hyvin suosittua. (Programmable logic controllers, 2009)



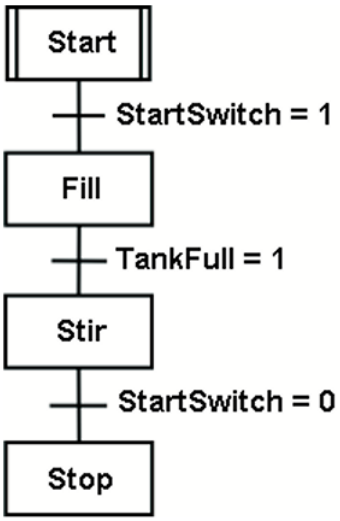
Kuva 3. Tikapuukaavio (Programmable logic controllers, 2009)

Toinen paljon käytössä oleva graafinen ohjelmointikieli on toimilohkokaavio (FBD). Toimilohkot eroavat perinteisistä funktioista siten, että toimilohkoilla on sisäinen tila ja muisti. Toimilohkon ulostulo riippuu siis sekä sisääntuloista että lohkon sisäisestä tilasta. Ero on siis pieni, mutta mahdollistaa paljon erilaisia sovelluksia ja tietyissä tilanteissa nopeuttaa ohjelmointia. (Opas toimilohko-ohjelmointiin, 2011)



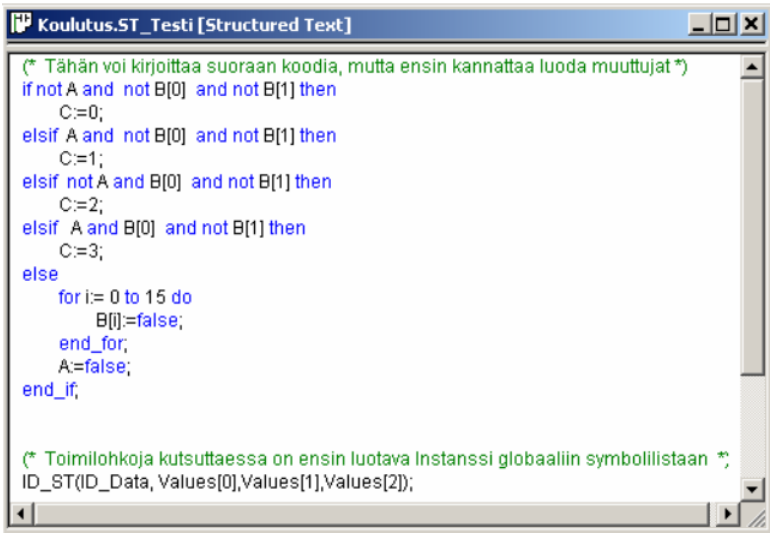
Kuva 4. Toimilohkokaavio (Opas toimilohko-ohjelmointiin, 2011)

Kolmas graafinen ohjelmointikieli, eli sekvenssivuokaavio (SFC) soveltuu parhaiten nimensä mukaisesti sekvenssien tekemiseen. Sekvenssivuokaaviota käytetään ylemmän tason ohjelmointiin, eli esimerkiksi sekvenssin rungon toteutukseen. Rungon lisäksi tarvittavat ehdot ja toimenpiteet toteutetaan yksinkertaisilla ehdoilla, tai ohjelmoidaan kokonaan toisella ohjelmointikielellä, koska SFC ei mahdollista monipuolisia ominaisuuksia näiden ohjelmoimiseksi. (Automaatiojärjestelmien logiikat ja ohjaustekniikat, 2007)



Kuva 5. Sekvenssivuokaavio (Automaatiojärjestelmien logiikat ja ohjaustekniikat, 2007)

Strukturoitu teksti ja komentolista ohjelmointi ovat lausemuotoisia ohjelmointikieliä. Ne eroavat graafisista kielistä täysin ja muistuttavatkin paljolti perinteisiä tekstimuotoisia ohjelmointikieliä. Asiaan perehtymättömälle lausemuotoiset ohjelmointikielet ovat selkeästi vaikeampiselkoisia, mutta monimutkaisempien ja laajempien kokonaisuuksien ohjelmointi saattaa olla merkittävästi joustavampaa ja nopeampaa näillä ohjelmointikielillä. Lausemuotoisissa ohjelmointikielissä hyvää on myös se, että ne ovat helpommin sovellettavissa eri valmistajien laitteisiin. (CX-One ja logiikkaohjelmointi, 2009)



Kuva 6. Strukturoitu teksti (CX-One ja logiikkaohjelmointi, 2009)

3 OMRON OHJELMOITAVAT LOGIIKAT

Omron on yksi lukuisista ohjelmoitavien logiikoiden valmistajista. Yritys valmistaa logiikoita kuudessa eri sarjassa, jotka jokainen sisältävät useita eri versioita mallisarjan logiikoista, hieman erilaisilla ominaisuuksilla. Tällä hetkellä valmistuksessa olevat sarjat ovat nimeltään CJ2, CP2E, CP1L, CP1E, CP1H ja CS1D. Eri sarjan logiikat eroavat toisistaan muun muassa I/O-pisteiden määrän, laajennusmahdollisuuksien, tuettujen ohjelmointikielien ja tiedonsiirtoväylien muodossa ja ovatkin suunniteltu kaikki eri käyttötarkoituksiin. Joukosta löytyy sekä yksittäisiä kompakteja logiikoita, että modulaarisia kokonaisuuksia. Kokonaisuudessaan vaihtoehtoja onkin siis useita kymmeniä jo yhden valmistajan valikoimassa. (Omron, 2023)

Kuva 7. Omron ohjelmoitavat logiikat (Omron, 2023)

Ohjelmoitavien logiikoiden lisäksi Omron valmistaa automaatiojärjestelmissä tarvittavia oheistuotteita. Valikoimasta löytyy logiikoiden laajennuksiin tarvittavia lisättäviä tulo- ja lähtöyksiköitä, erilaisia käyttöliittymiä kosketusnäyttöjen ja seurantalaitteiden muodossa, sekä etäyhteys ratkaisuja erilaisten verkkokytkimien muodossa. Lisäksi Omron tarjoaa myös valmiita teollisuuskäyttöön tarkoitettuja tietokoneita, sekä erillisiä koneautomaatio-ohjaimia. (Omron, 2023)

3.1 Omron CP1L

CP1L sarjan logiikoissa yhdistyy kompakti fyysinen koko ja suuren modulaarisen PLC:n monipuoliset ominaisuudet. Mallisarjan pienimmässä versiossa on käytettävissä ainoastaan 10 digitaalista I/O-pistettä, kun taas suurimmassa 60 I/O-pistettä. Logiikat ovatkin parhaimmillaan ohjaamassa yksittäisiä laitteita tai pieniä laitekokonaisuuksia. CP1L-logiikoita on tarjolla useilla eri piirikorteilla, riippuen muun muassa halutuista tiedonsiirtoväylistä, ulostulon tyypistä, tarvittavan muistin ja I/O-pisteiden

määrästä, sekä laajennusmahdollisuuksista. CP1L-sarjan logiikat voidaankin tarkemmin jaotella vielä CP1L-EL, -EM, -L ja -M versioihin. Koko CP1L sarja pohjautuu peruskonstruktioltaan aikaisempiin CP1H, CJ1 ja CS1 sarjoihin, joten aikaisemmin tehtyjä ohjelmia pystytään käyttämään suoraan ristiin näiden sarjojen logiikoiden välillä. (Omron, 2023)

Kuva 8. Omron CP1L (Omron, 2023)

Sarjan -EM ja -EL malleissa on sisäänrakennettu Ethernet-portti, joka takaa joustavan monitoroinnin ja ohjelmoimisen, tarvittaessa jopa etänä. CP1L-M ja -L mallit taas ovat lähtökohtaisesti varusteltu USB portilla. Lisäksi kaikkiin malleihin on valittavissa RS-232C tai RS-485 liitännät. (CP1L Datasheet, 2023) Logiikoiden laajennus on -EM ja -M malleilla mahdollista enintään kolmella CP1W laajennus yksiköllä, kun taas -EL ja -L malleilla vain yhdellä. Ohjelmamuistia on -EM ja -M malleissa tarjolla 10kS ja datamuistia 32kW, kun taas -EL ja -L malleissa ohjelmamuistia 5kS ja datamuistia 10kW. Logiikoihin pystytään lisävarusteena asentamaan 1–2 ylimääräistä sarjaporttia. (CX-One ja logiikka-ohjelmointi, 2009)

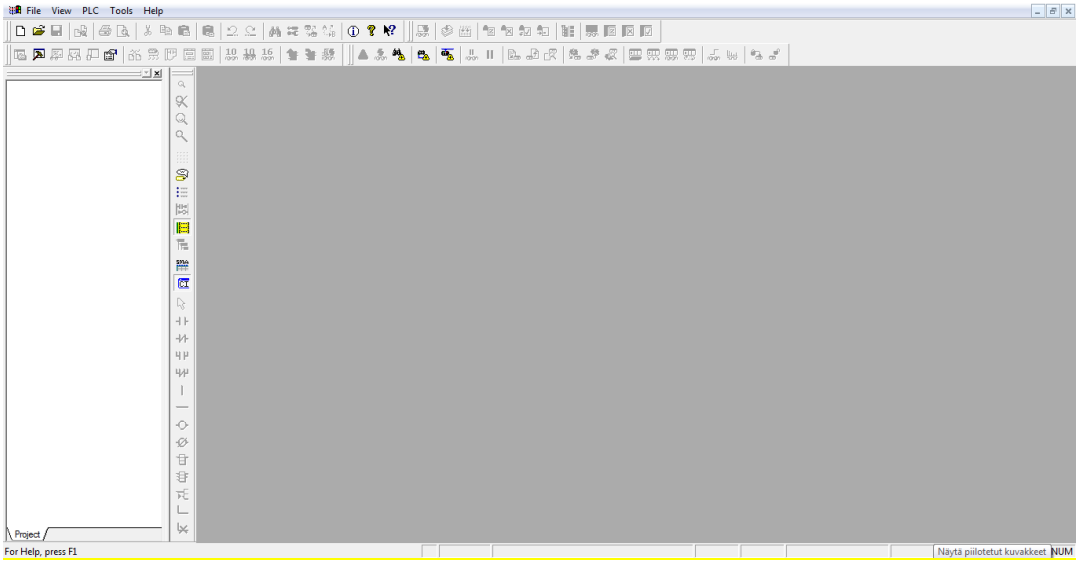
Laiteohjauksiin suunniteltuihin CP1L-mallisarjan logiikoihin on sisäänrakennettu toimintoja helpottamaan laitteiden hallintaa. Ominaisuuksina löytyy esimerkiksi edistyksellisen tehokas pulssiulostulo äärimmäisen tarkkaan paikoituksen ohjaamiseen, sekä valmiita ”high-speed” laskureita esimerkiksi pulssi- tai muiden korkea taajuuksisten antureiden lukemiseen. (CP1L Datasheet, 2023)

Tässä opinnäytetyössä ohjelmoitavalla logiikalla ohjataan yhtä pienehköä laitekokonaisuutta, jossa käytössä on tulopuolella muutamia antureita, rajakytkimiä ja painonappeja, sekä lähtöpuolella yksi moottorikäyttö ja muutamia merkkilamppuja. Juuri tällaiseen pienemmän kokonaisuuden ohjaamiseen CP1L-sarjan logiikat ovat suunniteltuja, ja sen takia valikoitui tässä projektissa käyttöömme.

3.2 Omron CX-One ohjelmisto

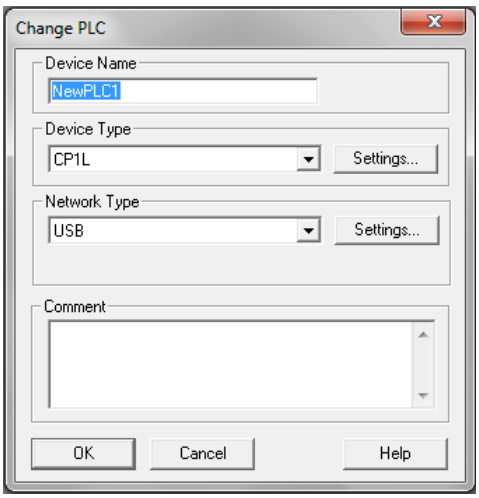
Omron on luonut ohjausjärjestelmiään varten CX-One ohjelmistopaketin, joka sisältää tarvittavat ohjelmat kaikkien Omronin valmistamien ohjauslaitteiden ohjelmointiin aina logiikoista käyttöliittymiin asti. CX-One tukee myös vanhempia jo valmistuksesta poistuneita komponentteja ja lisäksi sisältää myös tarvittavia työkaluja ja manuaaleja laitteiden hallintaan.

Omronin ohjelmoitavien logiikoiden ohjelmointia varten yleisimmin ja tässäkin työssä käytetty sovellus on nimeltään CX-Programmer. Ohjelmisto tukee kaikkia IEC 61131–3 standardin määrittämiä ohjelmointikieliä. CX-Programmerissa on valmiiksi hyvin laaja toimilohkokirjasto, joka sisältää esimerkiksi laskureita ja ajastimia. Toimilohkoja pystyy myös tarvittaessa luomaan itse lisää. (CX-One ja logiikkaohjelmointi, 2009)



Kuva 9. CX-Programmer ohjelman alkunäyttö

Kuva 9 havainnollistaa hyvin CX-Programmerin tyhjää alkunäkymää. Jotta Ohjelmoinnissa päästään alkuun, on ensimmäisenä lisättävä ohjelmaan logiikka ja määriteltävä sille asetukset. File-valikon kautta uutta projektia valittaessa avautuu ikkuna, jossa tämä päästään tekemään. Ohjelmalle kerrotaan laitteen nimi, laitteen tyyppi ja haluttu tiedonsiirtotapa. Settings-painikkeista avautuu tarvittaessa uusi valikko, josta pääsee tekemään tarkemman määrittelyn. Halutessaan pystyy myös lisäämään kommentin aloitetulle projektille.



Kuva 10. Logiikan lisääminen

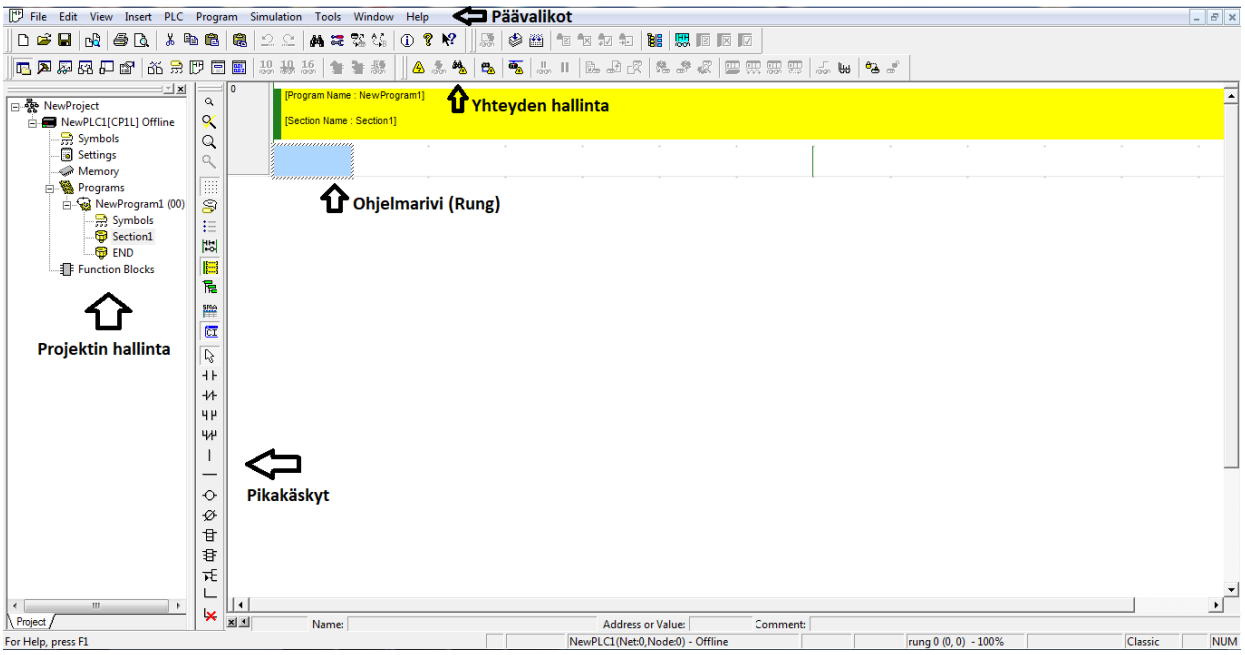
Asetuksien hyväksynnän jälkeen aukeaa kuvan 11 mukainen näkymä ja ohjelmointi voidaan aloittaa. Kuvaan on merkittynä pääpiirteittäin ohjelman toiminnan kannalta tärkeät osiot. Päävalikosta on pääsy esimerkiksi ohjelman sisäisiin työkaluihin ja manuaaleihin, sekä ohjelmiston perusasetuksiin.

Yhteyden hallintavalikko kattaa tarvittavat näppäimet logiikan yhteyden luontiin. Vaihtoehtoina on Work online, Direct online, Cp1L- Ethernet online, IP node online, sekä PLC:n vaihto monitorointi tilaan.

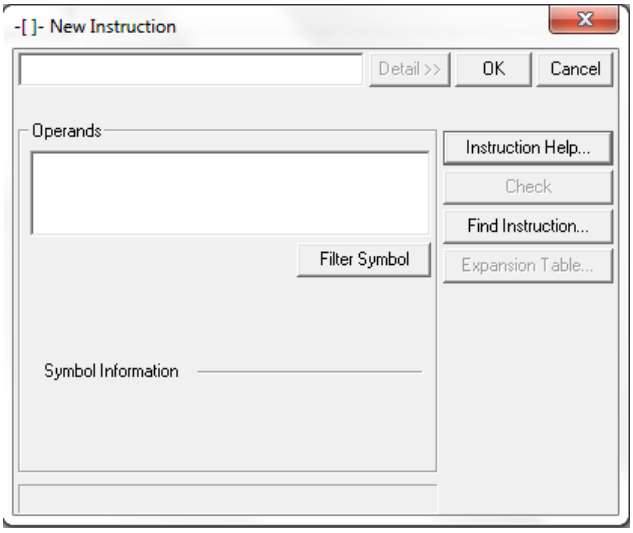
Ohjelmarivi on luonnollisesti paikka, johon ohjelmaa aloitetaan kirjoittamaan. Ohjelmointiin voidaan käyttää esimerkiksi pikakäskyjä tai tarvittaessa ”Function Blocks” painikkeen kautta hakea ja luoda omia toimilohkoja. Jos haluttua käskyä ei löydy pikavalikosta, voidaan se hakea myös ”Instruction Help” tai ”Find Instruction” painikkeiden kautta, jotka löytyvät uutta käskyä ohjelmaan lisätessä (kuva 12).

Ohjelmointirivi aukeaa perusoletuksena valmiina tikapuukaavio ohjelmointikielelle, mutta valitsemalla projektin hallinnasta Program -> Insert Program, pystyy ohjelmoinnin aloittamaan myös eri ohjelmointikielillä.

Projektin hallinta on niin sanottu projektivalikko, josta päästään käsiksi esimerkiksi eri projekteihin, logiikan asetuksiin, symbolitaulukkoon, tallennustilan hallintaan, logiikkaohjelmiin, ja edellä mainittuihin toimilohkoihin.

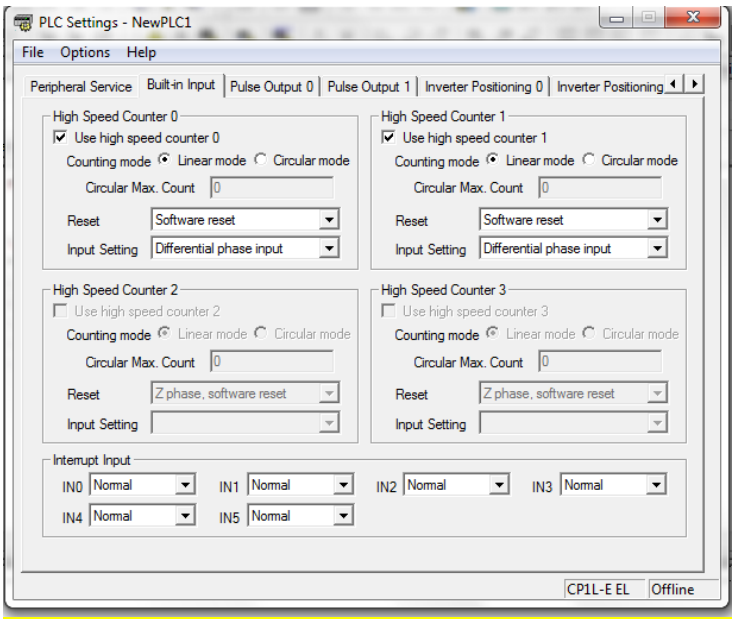


Kuva 11. CX-Programmer ohjelmointinäkymä



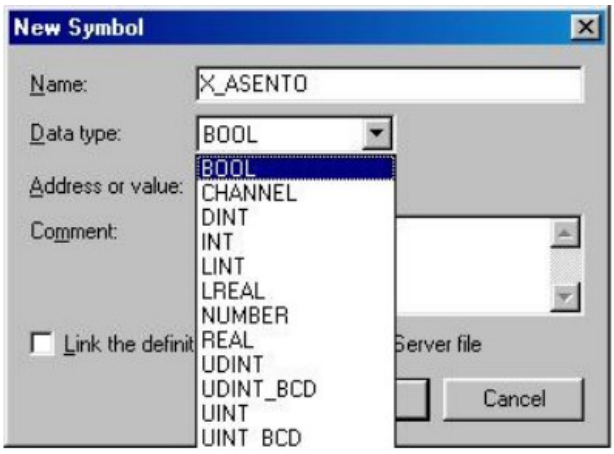
Kuva 12. Uuden käskyn lisääminen

Projektin hallinta valikosta PLC taulukon alapuolelta löytyy logiikan asetukset. Omronin ohjelmoitavissa logiikoissa on kätkössä useita toimintoja, jotka eivät ole oletusasetuksilla toiminnassa, vaan ne on otettava erikseen käyttöön. Esimerkiksi logiikkaan sisäänrakennetut pikalaskurit (High-speed counter) ja keskeytykset (interrupts) voidaan parametroida ja ottaa käyttöön tätä kautta. Saatavilla olevat asetukset ja toiminnot riippuvat täysin logiikan mallista. Asetukset eivät automaattisesti tallennu logiikalle, vaikka yhteys olisi muodostettuna, vaan ne on erikseen ohjelman tapaan ladattava logiikkaan.



Kuva 13. Logiikan lisätoiminnot

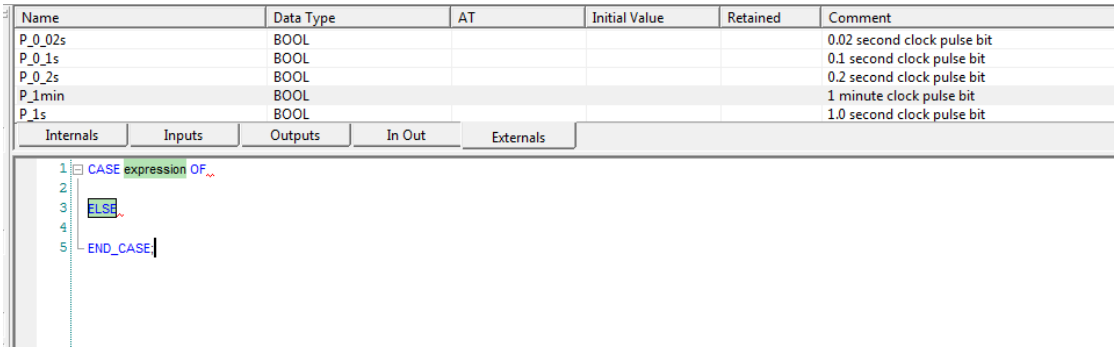
Yksi osa ohjelmointia on muuttujalistan tekeminen. CX-Programmerissa muuttujalilista voidaan tehdä etukäteen, ohjelmoinnin aikana tai jälkeen. Kaikille bitti- tai sanamuotoisille osoitteille voidaan antaa muuttuja sekä kommentti. Jokaiselle muuttujalle täytyy lisäksi määrittää sen osoitteelle tai käskylle hyväksyttävissä oleva tietotyyppi. Valittavissa olevia tietotyyppejä CX-Programmerissa on: Bool, Channel, Dint, Int, Lint, Lreal, Number, Real, Udint, Udint_bcd, Ulint, Ulint_bdc, Word, Dword, Lword ja string. (CX-One ja logiikkaohjelmointi, 2009)



Kuva 14. Symbolin lisääminen (CX-One ja logiikkaohjelmointi, 2009)

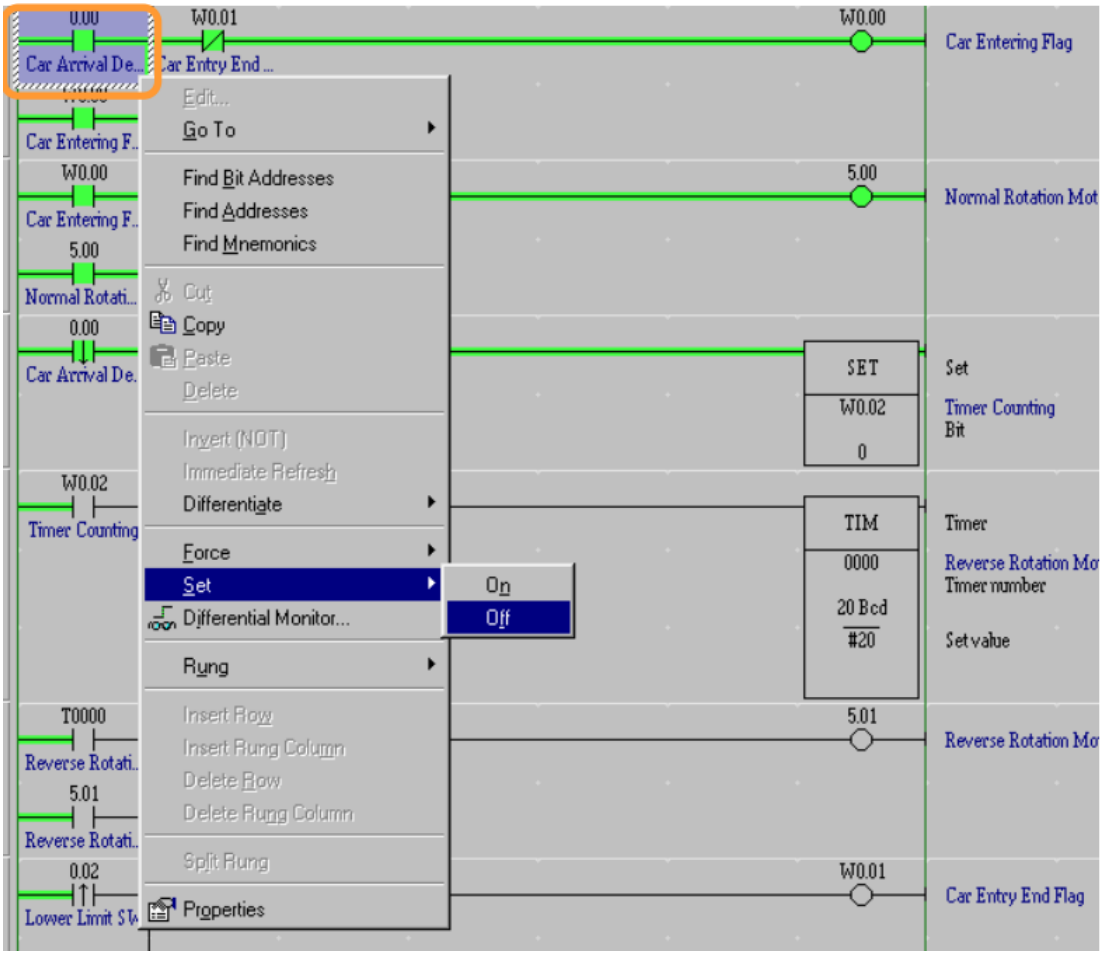
Jos valmiista toimilohkolistasta ei löydy tarvittavaa toimilohkoa, voidaan se luoda itse valitsemalla projektinhallinta valikosta Function Blocks -> Insert Function Block -> Ladder/Structured Text. Toimilohko voidaan siis luoda joko LD- tai ST-ohjelmointikielellä. Tikapuukaavio on helpompi käyttöinen, mutta monimutkaisempia ja erityisesti paljon laskentaa tekeviä toimilohkoja ohjelmoidessa on kannattavampaa hyödyntää strukturoitua tekstiä. (CX-One ja logiikkaohjelmointi, 2009)

Toimilohkoa ohjelmoidessa näkymä on kuvan 15 mukainen. Toimilohkolle asetetaan arvot vähintään lohkon sisäisille- ja ulkoisille-, sekä tulo- ja lähtömuuttujille.



Kuva 15. Toimilohkon ohjelmointi

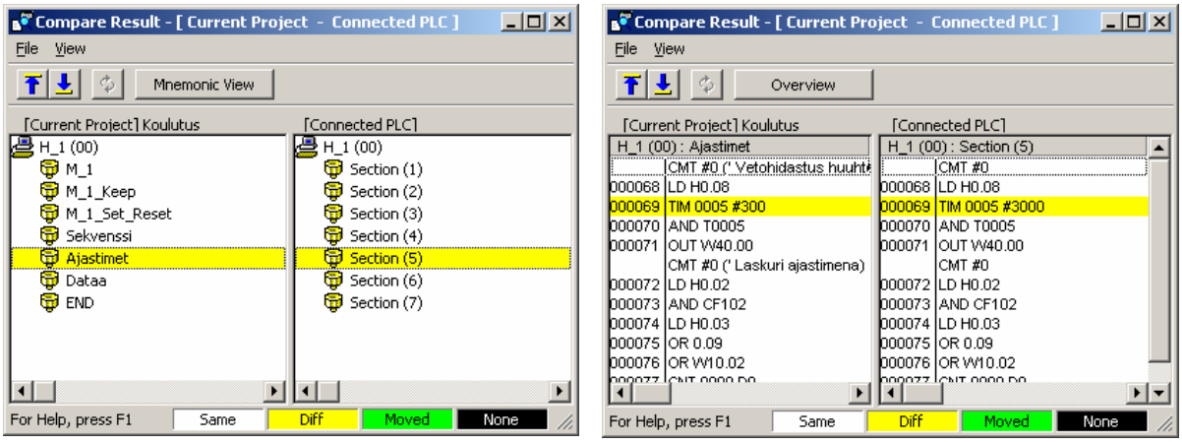
Ohjelmoinnin jälkeen olisi hyvä testata tehtyä ohjelmaa ennen kuin se otetaan käyttöön. Tehtyä ohjelmaa on mahdollista testata myös ilman yhteyttä logiikkaan, pelkällä tietokoneen näytöllä. CX-One ohjelmistopaketti sisältää CX-Simulator ohjelman, joka on tehty nimenomaan ohjelman simulointia varten. CX-Simulator voidaan avata erikseen ohjelmistopaketin kautta, tai CX-programmerin päävalikosta löytyvän simulation-painikkeen takaa.



Kuva 16 CX-Simulator esimerkki (CX-Simulator Introduction guide, 2008)

Kun ohjelma on saatu tehtyä ja todettu halutunlaiseksi, täytyy projekti tallentaa ja ladata käytettävään logiikkaan. Ohjelman lataamista varten on muodostettava yhteys ohjelmointivälineen ja logiikan välille. Jos yhteys otetaan suoraan logiikkaan, voidaan se tehdä esimerkiksi ”Work Online”-painikkeella. Näin ohjelma hakee automaattisesti logiikan määrittelyssä valitun tiedonsiirtotyypin mukaan käytössä olevia logiikoita ja löytäessä muodostaa yhteyden. Jos automaatiojärjestelmässä on kytkettynä useampia logiikoita ja yhteys otetaan esimerkiksi verkkokytkimen kautta, voidaan yhteyden muodostamiseen käyttää ”IP node online” komentoa. Tällöin on oltava tiedossa vähintään halutun logiikan IP-osoite.

Yhteyden muodostuttua logiikka vaihtuu online-tilaan ja tehty ohjelma voidaan ladata logiikalle ”Transfer to PLC”-painikkeen kautta. Logiikan ollessa online-tilassa ohjelmaa pystytään myös monitoroimaan reaaliajassa ja tarvittaessa tekemään pienempiä muutoksia ”work online” komennon avulla. Muutoksia tehtäessä logiikan tila on vaihdettava ajotilasta ohjelmointitilaan. Ohjelman lataus voidaan tehdä myös toisinpäin (Transfer from PLC), eli ladataan logiikalla jo oleva ohjelma CX-Programmeriin esimerkiksi monitorointia tai muokkaamista varten. CX-Programmer ohjelmassa on tarjolla myös valmis vertailutoiminto (Compare with PLC), jolla pystytään katsomaan koneelle tallennetun ja logiikalta jo löytyvän ohjelman eroavaisuuksia. (CX-One ja logiikkaohjelmointi, 2009)



Kuva 17 Vertailu esimerkki (CX-One ja logiikkaohjelmointi, 2009)

4 TYÖN TOTEUTUS

4.1 AW-Imuvaunupöytä

Imuvaunupöytä on leikkauskoneen yhteyteen tai alle tehty rakenne, jossa liikkuu kuvan 16 tyylinen imuvaunu. Imuvaunun tehtävänä on liikkua leikkauskoneen mukana ja täten poistaa leikkauksesta syntyvä savukaasu ja samalla kerätä tuleva leikkausjäte. Pöydän rakenteeseen on integroitu imukanavat, joita pitkin vaunuun järjestetään tarvittava imu. Imuvaunu avaa imukanavaa liikkuessaan, jolloin leikkaukseen saadaan aina tehokas kohdepoisto ja imuteho pysyy tasaisena koko leikkauksen ajan, riippumatta siitä, missä kohtaa pöytää leikkaus tapahtuu.

Kuva 18. Esimerkkikuva imuvaunusta (AirWell Oy, 2023)

4.2 Työn lähtökohdat

Projektin lähtökohtana oli tilanne, jossa aiemmin käytössä ollutta ohjausautomaatiota ei pystytty käyttämään asiakkaan tarpeiden mukaisen imuvaunupöydän ohjaamiseen. Asiakkaalla on käytössä yksi suuri leikkauspöytä, jossa työskentelee samanaikaisesti kaksi leikkauskonetta, mutta imuvaunupöydän vanha ohjausautomaatio oli suunniteltu ohjaamaan ainoastaan yhtä imuvaunua. Oli siis tarve suunnitella ja toteuttaa uusi ohjausautomaatio.

Vanha ohjausautomaatio ja imuvaunun toimintaperiaate yleisestikään ottaen eivät olleet minulle ennestään tuttuja, joten projekti aloitettiin ensimmäisenä tutustumalla imuvaunun nykyiseen ohjausautomaatioon. Yrityksen tiloissa ei sattunut olemaan juuri tällä hetkellä valmisteilla yhtäkään imuvaunupöytää, joten päädyttiin rakentamaan vanhan ohjausautomaation raakaversio testipenkkiin, jossa päästiin konkreettisesti tutustumaan vanhaan logiikkaohjelmaan ja simuloimaan ohjauksen toimintoja oikeilla komponenteilla.

Tutustuminen antoi melko hyvän käsityksen siitä, mitä tulevan imuvaunupöydän on määrä tehdä ja minkälaista automaatiota imuvaunun ohjaamiseen saatettaisiin tarvita. Tutustumisen pohjalta pystyttiin aloittamaan suunnittelu hahmottelemalla paperille minkälaisia muutoksia uusi automaatiojärjestelmä tulisi todennäköisesti tarvitsemaan ja pystyttäisiinkö osaa vanhasta ohjausjärjestelmästä vielä hyödyntämään.

4.3 Projektin suunnittelu

Onnistuneen automaatiosuunnittelun pohjana on useimmiten erilaisten turvallisuus- ja muiden vaatimusten täyttämisen lisäksi asiakkaan toiveet ja vaatimukset laitteiston toiminnalle. Tässä vaiheessa oli tiedossa, että imuvaunuille on ainakin saatava erikseen automaatti-, ja manuaaliajo mahdollisuus. Automaattiajolla imuvaunun on määrä seurata oman leikkauskoneensa liikkeiden mukana ja manuaaliajolla liikettä ohjattaisiin fyysisillä painikkeilla erillisellä kauko-ohjaimella. Kauko-ohjain sijoitetaan leikkauskoneen ohjauksen läheisyyteen. Manuaaliajoa käytetään esimerkiksi vaunun tyhjennyspaikalle ajamiseen ja vaunun paikoittamiseen.

Tiedossa oli myös se, että imuvaunujen tyhjennyspaikka tullaan haluamaan pöydän keskikohtaan ja leikkauskoneilla tullaan muutenkin ajamaan ajoittain pöydän keskikohdan toisella puolella, jolloin toinen imuvaunuista saattaa olla automaattiajolla samaan aikaan, kun toinen on manuaaliajolla. Tämä aiheuttaa mahdollisuuden imuvaunujen törmäämiselle, joten sen estämiseksi olisi ehdottomasti rakennettava törmäyssuojaus.

Myös hätäseis-piiri täytyy tehdä imuvaunupöydälle erikseen. Sitä ei integroida leikkauskoneen piiriin, sillä imuvaunupöytä toimii tässä tapauksessa omana irrallisena laitteenaan. Imuvaunu kulkee leikkauspöydän alla, jonne ei ole pääsyä muulloin, kun imuvaunua tyhjennettäessä, joten normaali ajotilanteessa välitöntä vaaraa ihmiselle ei ole. Tässä päädyttiinkin ratkaisuun, jossa yksi hätäseis-painike tulee kaukosäätimen yhteyteen ja imuvaunun tyhjennyspaikalle avattavaan luukkuun asetetaan avautuva kosketin, joka varmistaa laitteiston toiminnan pysähtymisen imuvaunun tyhjentämisen ajaksi.

Vaikka imuvaunut tulevat toimimaan samalla pöydällä, halutaan ne pitää leikkauskoneiden mukaisesti omien ohjauksiensa takana. Molempia leikkauskoneita voidaan ajaa itsenäisesti, joten myös imuvaunut omina järjestelminään pitämällä, vältytään tilanteelta, jossa toisen vaunun ohjaus vikaantuisi ja leikkaaminen jouduttaisiin lopettamaan molemmilla koneilla. Erilliset järjestelmät tarkoittavat tässä tapauksessa sitä, että molemmille imuvaunuille tarvitaan myös omat ohjauskaappinsa. Toisaalta tämä helpottaa johdotuksessa, sillä molempien vaunujen moottorit ovat joka tapauksessa sijoiteltava pöydän päihin, joka tarkoittaa tässä tapauksessa useamman kymmenen metrin välimatkaa. Nyt myös ohjauskaapit sijoittelemalla pöydän molempiin päihin, vältytään pisimmiltä kaapelivedoilta, joka on tärkeää erityisesti taajuusmuuttajakäyttöisten moottoreiden kohdalla.

4.4 Komponenttien valinta

Seuraava osa suunnittelua oli käytettävien komponenttien valinta. Projektissa pyrittiin hyödyntämään yrityksen omasta varastosta jo löytyviä komponentteja, mutta osa hankittiin tätä projektia varten.

Imuvaunun liike on aikaisemmin toteutettu niin, että oikosulkumoottorilla pyöritetään vetoakselia, jonka hammaspyörät liikuttavat imuvaunua ketjun välityksellä eri suuntiin. Liikkeen toteutusta on turha lähteä tätä projektia varten keksimään uudestaan, joten tämä hyväksi todettu konstruktio tullaan pitämään edelleen samanlaisena. Moottoriksi valittiin Moves ME90S-sarjan 1,1kW tehoinen oikosulkumoottori, joka kytketään 1:30 välityksellä olevaan saman valmistajan vaihdelaatikkoon.

Kuva 19. ME90S Sähkömoottori (Moves Oy, 2023)

Imuvaunua liikuttavan moottorin ohjaaminen on aikaisemmissa versioissa tapahtunut Vacon 20- sarjan taajuusmuuttajilla, eikä tässä tilanteessa ollut syytä lähteä sitä vaihtamaan. Taajuusmuuttajan ominaisuudet olivat siis ennalta tiedossa, joten valintaa tehdessä oli otettava huomioon vain se, että taajuusmuuttaja on valitun moottorin tehojen mukainen.

Kuva 20. Vacon 20- taajuusmuuttaja (Vacon Oy, 2023)

Tässä projektissa imuvaunulaitteiston ohjausta ei lähdetty integroimaan asiakkaan leikkauskoneiden hätäseis-piirin yhteyteen, joten imuvaunulle oli tehtävä oma hätäseis-piiri. Piiriin kuuluu ainoastaan yksi hätäseis- painike ja yksi avautuva kosketin. Näin pienessä järjestelmässä ei ole järkevää käyttää ohjelmoitavaa turvalogiikkaa, vaan hätäseis-piirin toteutukseen valittiin toiminnoiltaan hieman yksinkertaisempi vaihtoehto, eli turvarele. Duelcon mallistosta NST-3.2 turvarele osoittautui sopivaksi vaihtoehdoksi ominaisuuksiensa ja hyvän saatavuuden vuoksi. Releessä on kolme NO turvaulostuloa ja yksi NC.

Kuva 21. Duelco NST-3.2 turvarele (Duelco Oy, 2023)

Jotta imuvaunujen liikkeitä pystytään ohjaamaan ja seuraamaan, on niistä ja leikkauskoneista saatava logiikalle paikkatietoa. Imuvaunujen ja leikkauskoneiden paikoitustiedon saamista varten ratkaisua pohdittiin pulssi- ja absoluuttiantureiden välillä. Suurin ero näiden välillä on se, että absoluuttianturi ei hävitä paikkatietoaan sähkönsyötön katketessa, joten sitä ei tarvitsisi esimerkiksi kalibroida yhtä usein.

Tässä projektissa kuitenkin asiakkaan leikkauskoneen ohjausautomaatio on toteutettu niin, että kone on aina käynnistyksen yhteydessä joka tapauksessa ajettava manuaalisesti omaan nollapisteeseensä ja suoritettava paikoitustiedon nollaus. Olisi siis käyttäjän kannalta loogista tehdä sama myös imuvaunuille. Tässä tilanteessa imuvaunun nollaamisesta ei myöskään koituisi juurikaan ylimääräistä työtä. Imuvaunun kohdistus leikkauskoneeseen nähden saataisiin pysymään aina samana, kun molemmille laitteille asetettaisiin sama nollapiste. Imuvaunujen tulevaa törmäyssuojaa ajatellenkaan ongelmaa pulssianturin käytöstä ei tule, sillä logiikkaohjelmaan saadaan tallennettua pulssianturin viimeisin lukema, vaikka toisen imuvaunun pulssianturi vikaantuisi.

Emme lopulta löytäneet mitään ongelmaa perinteisemmän pulssianturin käyttöön, ja niitä oli saatavilla valmiiksi yrityksen varastosta, joten valitsimme käyttää niitä. Antureiksi valikoitui Lika Electronic I58S -sarjan pulssianturit.

Kuva 22. Lika Electronic I58S pulssianturi (Lika electronics, 2023)

Ohjauspuolen kuorma tulee olemaan hyvin pieni, joten logiikalle ja muille tasajännitettä vaativille komponenteille valikoitui virtalähteeksi Omronin S8VK-sarjan DIN-kisko asenteinen virtalähde pienemmällä viiden ampeerin ulostulovirralla.

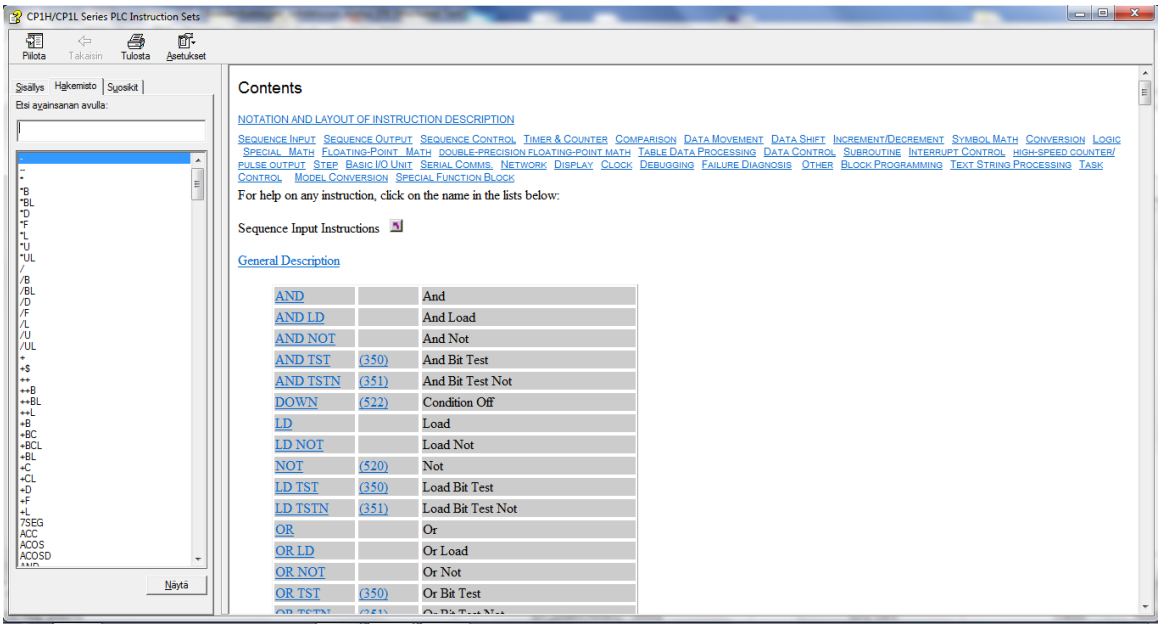
Kuva 23. Omron S8VK virtalähde (Omron Oy, 2023)

Yritys on aikaisemminkin käyttänyt Omronin ohjainlaitteita, joten tähänkin projektiin otettiin käyttöön Omronin ohjelmoitava logiikka. CP1L-sarjan ohjelmoitavat logiikat ovat suunniteltuja juuri tämäntyyppiseen laitteiston ohjaamiseen, joten käyttöön valittiin Omronin CP1L-EL. Suunniteltujen I/O-pisteiden perusteella logiikoista valikoitui käyttöön CP1L-EL sarjan pienin versio jossa on käytettävissä 11 sisääntuloa ja 7 ulostuloa. Omronin ohjelmoitavia logiikoita käytiin tarkemmin läpi kappaleessa 3.

Lopuksi oli mietittävä asennukseen sopiva kytkentäkaappi. Kytkentäkaapin on oltava fyysisesti sopivan kokoinen valituille komponenteille ja vähintään pölytiivis, sillä paikat joissa leikkauskoneita käytetään ovat usein erittäin pölyisiä. Kytkentäkaapiksi valikoitui Rittalin valikoimasta löytyvä sopivan kokoinen kytkentäkaappi, joka on suojausluokaltaan IP66.

4.5 Ohjelmointi

Logiikoiden ohjelmoimiseen käytettiin Omronin omaa CX-Programmer ohjelmistoa, johon tutustuttiin tarkemmin kappaleessa 3.2. Ohjelmointi oli haastavin ja työläin osuus tässä opinnäytetyössä. Alkuvaikeuksien jälkeen Omronin ohjelmointiympäristöön päästiin kuitenkin hyvin sisälle, ja se osoittautuikin lopulta melko helppokäyttöiseksi todella kattavien ohjelman sisäisten ohjeiden myötä. Ohjelmiston sisältä löytyy useita erilaisia manuaaleja, sekä erikseen ”Instruction Reference” osio, josta löytää jokaiselle toiminnolle laajan kuvauksen ja seikkaperäisen ohjeen käyttämistä varten.



Kuva 24. Instructions reference

Imuvaunun aikaisempaa logiikkaohjelmaa pystyttiin käyttämään mallina ohjelmoinnissa alkuun pääsemiseksi. Loppujen lopuksi hyödyntämään kuitenkin pystyttiin ainoastaan pohjaa moottorin ajokäskyille, muu osuus ohjelmasta tehtiin uudelleen. Symbolilista on tässä työssä esillä ensimmäisenä, mutta todellisuudessa sitä ei kuitenkaan tehty heti alussa vaan sitä tehtiin ja täydennettiin ohjelman etenemisen mukaan. Ohjelmointi aloitettiin moottoriajoista.

Name	Data Type	Address / Value	Rack Locati...	Usage	Comment
	CHANNEL	1		Work	"
	BOOL	100.02		Out	1/3 nopeus
	BOOL	100.03		Out	2/3 nopeus
	BOOL	100.04		Out	merkkivalo "automaattiajo estetty...
	BOOL	100.05		Out	Merkkivalo "nollaus suoritettu / a...
	CHANNEL	A272		Work	.
	BOOL	A531.00		Work	Nollaa koneen pulssianturin arvon
	BOOL	A531.01		Work	Nollaa moottorin pulssianturin ar...
	CHANNEL	D1		Work	.
	CHANNEL	D210		Work	erotusluku kun liike taakse
	CHANNEL	D220		Work	Erotusluku liike eteenpäi
	CHANNEL	D300		Work	Koneen pulssianturin pulssimäärä...
	CHANNEL	D310		Work	Koneen korjattu pulssimäärä
	CHANNEL	D320		Work	Koneen korjattu pulssimäärä muu...
	CHANNEL	D360		Work	Koneen kulkema pulssimäärä suh...
	CHANNEL	D400		Work	Moottorin pulssianturin ulssimäärä
	CHANNEL	D500		Work	.
	CHANNEL	D600		Work	Yhteenlaskettu tulos
	BOOL	H0.00		Work	Hätäseis_apubitti
	BOOL	H0.02		Work	Auto taakse 1/3 nopeus
	BOOL	H0.03		Work	Auto taakse 2/3
	BOOL	H0.04		Work	Auto taakse 3/3
	BOOL	H0.05		Work	Auto eteen 1/3
	BOOL	H0.06		Work	Auto eteen 2/3
	BOOL	H0.07		Work	Auto Eteen 3/3
	BOOL	H0.10		Work	törmäyssuoja apubitti, avautuva k...
	BOOL	H0.11		Work	apubitti interlockin eli "ajoneston...
	BOOL	H0.14		Work	vauhdin hidastus apubitti, avautu...
Auto_Man_kytkin	BOOL	0.05		In	Auto Man kytkin
ctbl	FB [_CPU011...	N/A [Auto]			

Kuva 25. Symbolilista 1/3

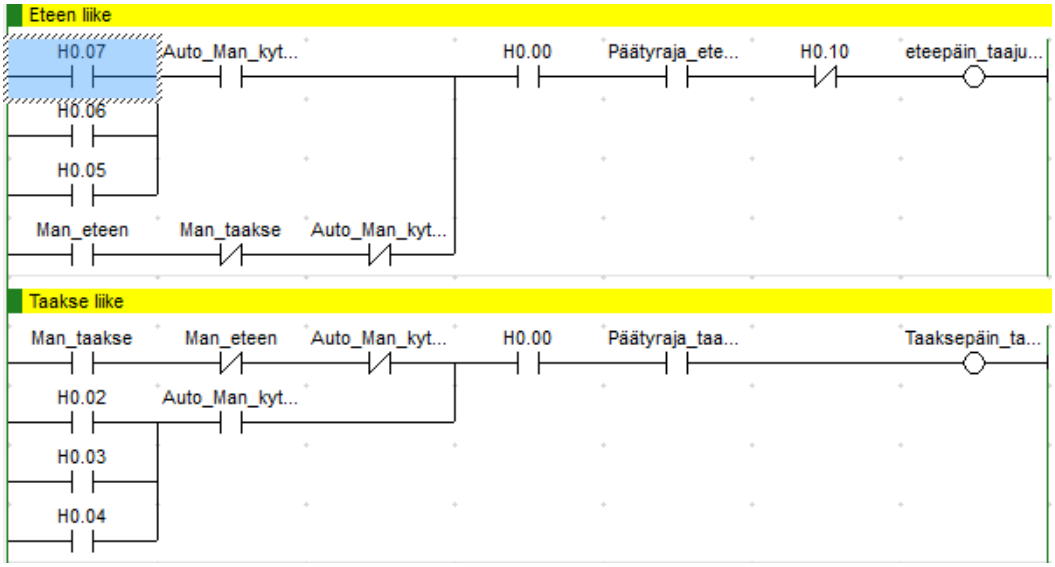
Name	Data Type	Address / Value	Rack Locati...	Usage	Comment
eteepäin_taajuusmuut...	BOOL	100.00		Out	2/3 Eteenpäin ohjaus taajuusmuu...
Hätäseis_tila_tieto	BOOL	0.04		In	Hätäseis releeltä tuleva tieto
Koneen_Pulssianturi_A...	BOOL	0.00		In	Koneen pulssianturi A pulssi
Koneen_pulssianturi_B...	BOOL	0.01		In	Koneen pulssianturi B pulssi
luedatanoodi76	FB [_CPU011...	N/A [Auto]			
Man_eteen	BOOL	0.08		In	Man eteen komento
Man_taakse	BOOL	0.09		In	man taakse komento
Moottorin_pulssiantur...	BOOL	0.02		In	Moottorin pulssianturin A pulssi
Moottorin_pulssiantur...	BOOL	0.03		In	Moottorin pulssianturin B pulssi
P_0_02s	BOOL	CF103		Work	0.02 second clock pulse bit
P_0_1s	BOOL	CF100		Work	0.1 second clock pulse bit
P_0_2s	BOOL	CF101		Work	0.2 second clock pulse bit
P_1min	BOOL	CF104		Work	1 minute clock pulse bit
P_1s	BOOL	CF102		Work	1.0 second clock pulse bit
P_AER	BOOL	CF011		Work	Access Error Flag
P_CIO	WORD	A450		Work	CIO Area Parameter
P_CY	BOOL	CF004		Work	Carry (CY) Flag
P_Cycle_Time_Error	BOOL	A401.08		Work	Cycle Time Error Flag
P_Cycle_Time_Value	UDINT	A264		Work	Present Scan Time
P_DM	WORD	A460		Work	DM Area Parameter
P_EM0	WORD	A461		Work	EM0 Area Parameter
P_EM1	WORD	A462		Work	EM1 Area Parameter
P_EM2	WORD	A463		Work	EM2 Area Parameter
P_EM3	WORD	A464		Work	EM3 Area Parameter
P_EM4	WORD	A465		Work	EM4 Area Parameter
P_EM5	WORD	A466		Work	EM5 Area Parameter
P_EM6	WORD	A467		Work	EM6 Area Parameter
P_EM7	WORD	A468		Work	EM7 Area Parameter
P_EM8	WORD	A469		Work	EM8 Area Parameter
P_EM9	WORD	A470		Work	EM9 Area Parameter

Kuva 26. Symbolilista 2/3

Name	Data Type	Address / Value	Rack Locati...	Usage	Comment
P_EM8	WORD	A469		Work	EM8 Area Parameter
P_EM9	WORD	A470		Work	EM9 Area Parameter
P_EMA	WORD	A471		Work	EMA Area Parameter
P_EMB	WORD	A472		Work	EMB Area Parameter
P_EMC	WORD	A473		Work	EMC Area Parameter
P_EQ	BOOL	CF006		Work	Equals (EQ) Flag
P_ER	BOOL	CF003		Work	Instruction Execution Error (ER) Flag
P_First_Cycle	BOOL	A200.11		Work	First Cycle Flag
P_First_Cycle_Task	BOOL	A200.15		Work	First Task Execution Flag
P_GE	BOOL	CF000		Work	Greater Than or Equals (GE) Flag
P_GT	BOOL	CF005		Work	Greater Than (GT) Flag
P_HR	WORD	A452		Work	HR Area Parameter
P_IO_Verify_Error	BOOL	A402.09		Work	I/O Verification Error Flag
P_LE	BOOL	CF002		Work	Less Than or Equals (LE) Flag
P_Low_Battery	BOOL	A402.04		Work	Low Battery Flag
P_LT	BOOL	CF007		Work	Less Than (LT) Flag
P_Max_Cycle_Time	UDINT	A262		Work	Maximum Cycle Time
P_N	BOOL	CF008		Work	Negative (N) Flag
P_NE	BOOL	CF001		Work	Not Equals (NE) Flag
P_OF	BOOL	CF009		Work	Overflow (OF) Flag
P_Off	BOOL	CF114		Work	Always OFF Flag
P_On	BOOL	CF113		Work	Always ON Flag
P_Output_Off_Bit	BOOL	A500.15		Work	Output OFF Bit
P_Step	BOOL	A200.12		Work	Step Flag
P_UF	BOOL	CF010		Work	Underflow (UF) Flag
P_WR	WORD	A451		Work	WR Area Parameter
pulssi	BOOL	0.11		In	nollapulssi
pulssi1	BOOL	0.10		In	nolllapulssi-
Päätyraja_eteenpäin_lii...	BOOL	0.06		In	+ raja
Päätyraja taaksepäin l...	BOOL	0.07		In	- raja

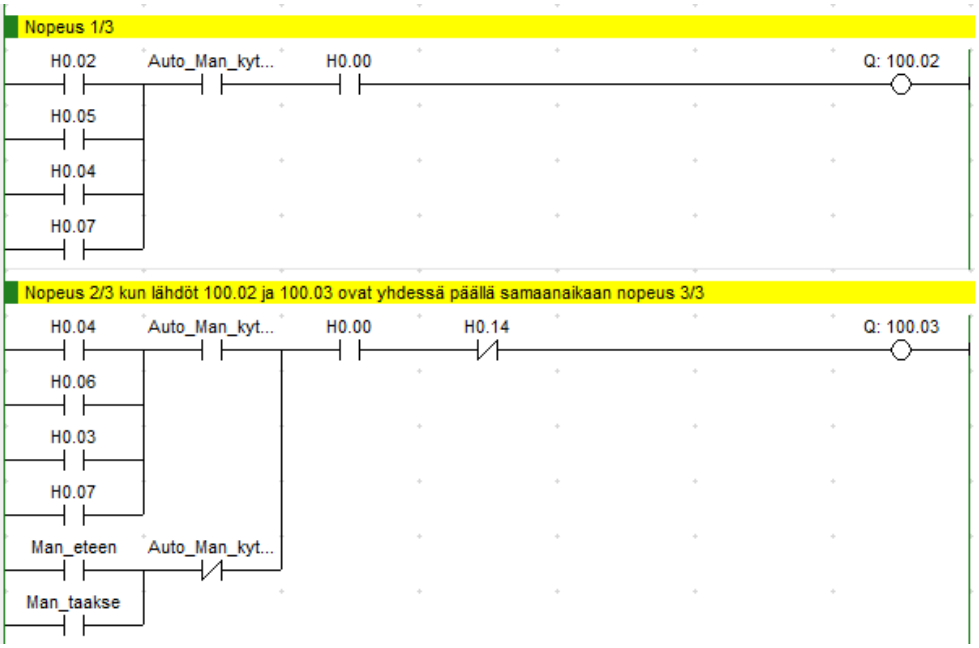
Kuva 27. Symbolilista 3/3

Ensimmäisenä ohjelmoitiin moottorin eteen ja taakse liikkeet. Molemmat liikkeet toimivat joko manuaali- tai automaattiajolla. Molemmat ajosuunnat vaativat, ettei turvarele ole lauenneena, eikä vaunua ole ajettu päätyrajaansa. Lisäksi eteenpäin liikkeen voi estää vaunujen törmäyssuojalla ohjautuva avautuva kosketin.



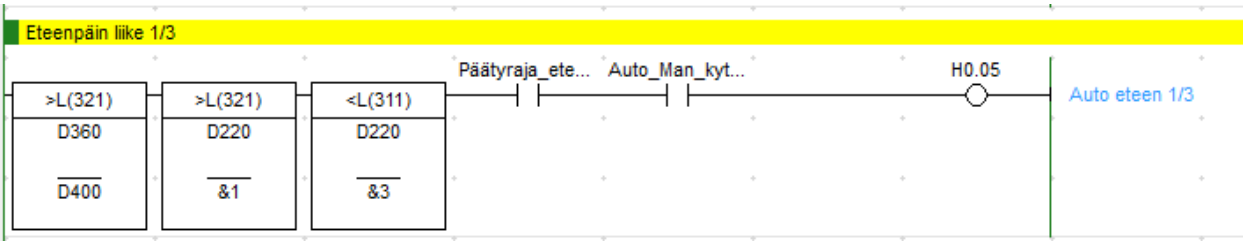
Kuva 28. Eteen ja taakse liikkeet

Moottoreille tehtiin kolme eri ajonopeutta molemmille ajosuunnille. Tässä käytettiin samaa pohjaa kuin aikaisemmassa ohjelmaversiossa, eli ohjelmaan tehtiin kaksi eri nopeutta ja kolmas nopeus kytkeytyy, kun ne ovat molemmat päällä samaan aikaan. Logiikkaohjelma ohjaa siis logiikan ulostuloja, joilla ohjataan taajuusmuuttajaa. Käytetyt ajonopeudet valitaan erikseen taajuusmuuttajan parametrointia tehdessä.



Kuva 29. Ajonopeudet

Manuaaliajolla käytössä on vain 1/3 nopeus ja kaksi isompaa ajonopeutta on tehty automaattiajoa varten. Leikkauskoneen leikatessa ajonopeus on usein melko hidas, mutta siirtymät leikkauspisteistä toisiin saatetaan kulkea suurellakin nopeudella. Niinpä automaattiajolla eri ajonopeudet määräytyvät leikkauskoneen ja imuvaunun pulssiantureiden lukemien kasvavan eron myötä.



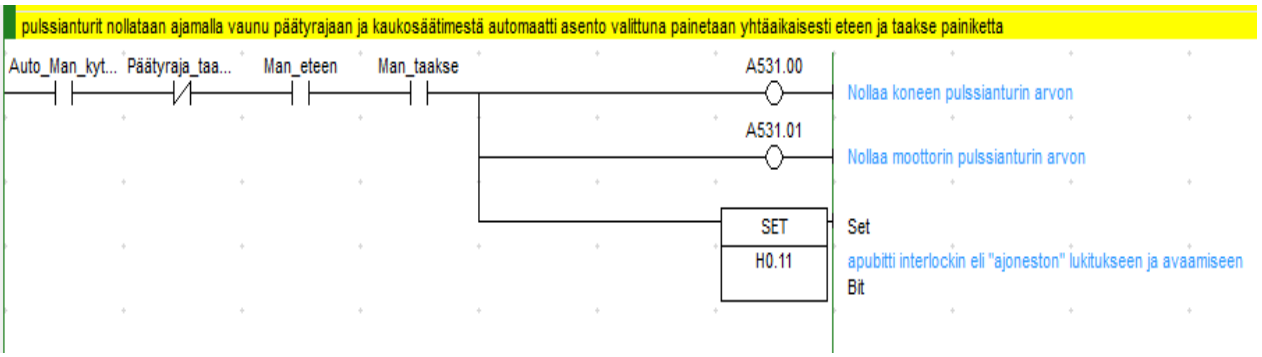
Kuva 30. Imuvaunun eteenpäin liike automaattiajolla

Pulssiantureiden käyttöönottoa varten oli logiikan asetuksista valittava ”High-speed counter”-toiminto aktiiviseksi. Käyttöönoton jälkeen esimerkiksi ensimmäisen pikalaskurin lukema tallentuu automaattisesti A270-muistipaikalle.

Kuva 31. Pikalaskuri

Imuvaunun kalibrointi leikkauskoneeseen nähden oli aikaisemmassa versiossa toteutettu niin, että logiikkaohjelma nollaa pulssiantureiden lukemat aina manuaaliajolta automaatille vaihdettaessa. Imuvaunu oli siis esimerkiksi tyhjennyksen jälkeen aina manuaalisesti kohdistettava keskelle leikkauskonetta. Tätä tapaa ei nähty enää käytännölliseksi ja toimivaksi kahden imuvaunun pöydällä, vaan paikkatiedon nollaamista lähdettiin suunnittelemaan uudestaan erilaisella tavalla.

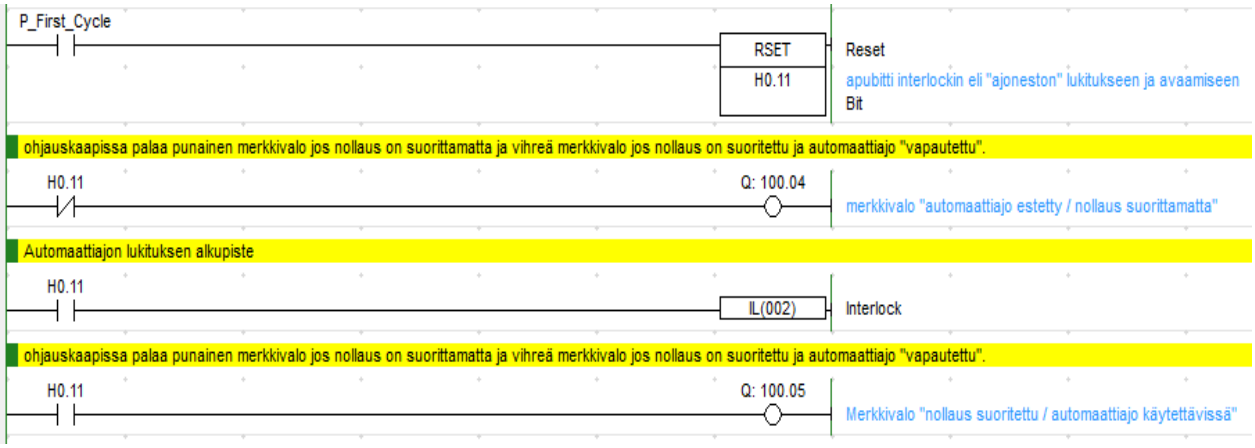
Nyt paikkatietojen nollaus toteutetaan niin, että imuvaunu on ajettava manuaaliajolla mekaaniseen päätyrajakytkimeensä ja valittava kaukosäätimestä automaattiajo päälle, jonka jälkeen manuaaliajon eteen ja taakse painikkeita yhtä aikaa painaminen nollaa imuvaunun ja leikkauskoneen pulssiantureiden lukemat.



Kuva 32 Pulssianturin nollaaminen

Imuvaunun päätyraja sijoitetaan fyysisesti leikkauskoneen oman nollapisteen mukaan niin, että siihen ajettaessa imuvaunu on aina keskellä leikkauspoltinta, jolloin kalibrointia ei tarvitse tehdä silmällä, vaan koneiden kohdistus pysyy aina täysin samana. Kalibrointi tehdään lähtökohtaisesti vain konetta käynnistettäessä, eli ohjelma mahdollistaa nyt myös sen, että manuaaliajon jälkeen automaatille vaihdettaessa, vaunu palaa automaattisesti leikkauskoneen alle ja ylimääräiseltä kalibroinnilta vältytään esimerkiksi imuvaunun roska-astian tyhjentämisen jälkeen.

Jotta leikkauskoneen kuski varmasti muistaisi tehdä nollauksen, ohjelma tehtiin niin, että imuvaunun ohjausta käynnistettäessä käytössä on ainoastaan manuaaliajo niin kauan, kunnes nollaus on suoritettu onnistuneesti. Ohjauskaapin kanteen lisättiin punainen ja vihreä merkkivalo osoittamaan nollauksen tilatietoa. Punainen valo kertoo, että nollaus on vielä suorittamatta ja automaattiajo lukittuna. Onnistuneen nollauksen jälkeen punainen merkkivalo sammuu ja vihreä syttyy. Automaatti ajon lukitus tehtiin logiikkaohjelmaan Interlock (IL(002)) käskyllä.

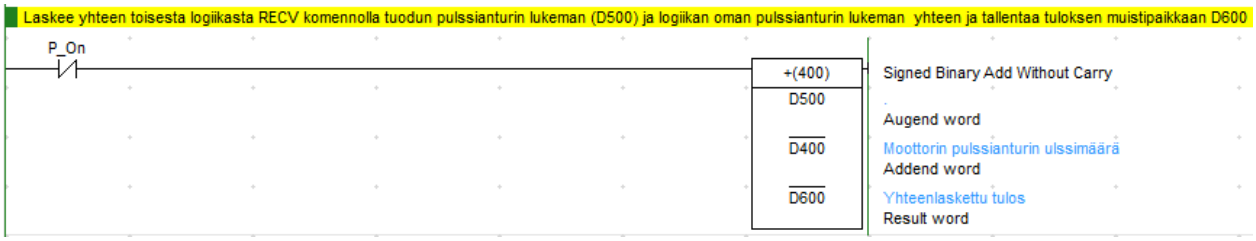


Kuva 33. Ajonesto

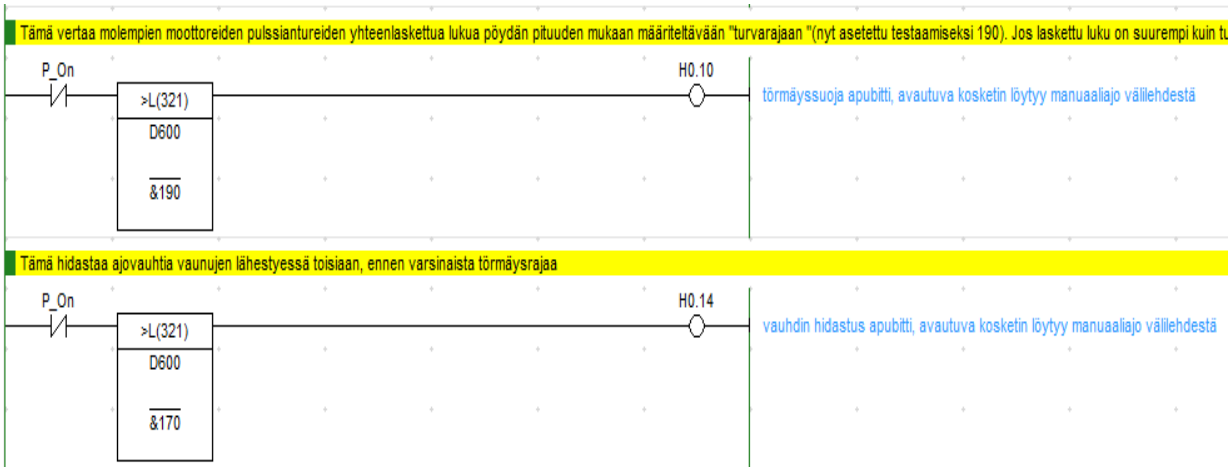
Imuvaunujen törmäyssuojaksi pohdittiin erilaisia ratkaisuita ja jonkinlaiset mekaaniset rajakytkimet vaikuttivat aluksi helpoimmalta vaihtoehdolta. Tällainen ratkaisu olisi kuitenkin käytännössä mahdoton toteuttaa asiallisesti, sillä rajakytkimet joutuisivat todella kovan kuormituksen alle. Lisäksi törmäystilanteessa niin pikainen äkkijarrutus rasittaisi imuvaunujen voimansiirtoa valtavasti. Vaunut luultavasti eivät ehtisi edes pysähtyä ajoissa, sillä täysvauhdista jarrutuksessa on joka tapauksessa pieni viive, ja rajakytkimet tulisivat varmasti hajoamaan ennemmin tai myöhemmin. Nopeasti tulikin selväksi, että suojaus täytyisi tehdä logiikkaohjelmaan.

Törmäyssuojausta alettiin toteuttamaan niin, että ohjelmaan tehtiin toiminto, joka vertailee imuvaunujen pulssiantureiden yhteenlaskettua lukemaa. Kun leikkauspöydän pituus saadaan selvitettyä pulssianturin antamina pulsseina, voidaan pulssiantureiden yhteenlaskettu lukema laittaa vertailtavaksi pöydän pituuteen nähden. Näin saadaan tehtyä erilaisia raja-arvoja, jossa imuvaunut aluksi hiljentäisivät ajonopeuttaan lähestyessään tiettyä yhteenlaskettua lukua, ja lopulta pysähtyisivät kokonaan. Imuvaunujen liikkuminen olisi estetty niin kauan, kunnes toinen vaunu lähtee liikkeelle eri

suuntaan ja yhteenlaskettu lukema laskee alle vertailuluvun. Tällä tavalla suojaus toimii, huolimatta siitä kummassa päässä leikkauspöytää vaunut työskentelevät.



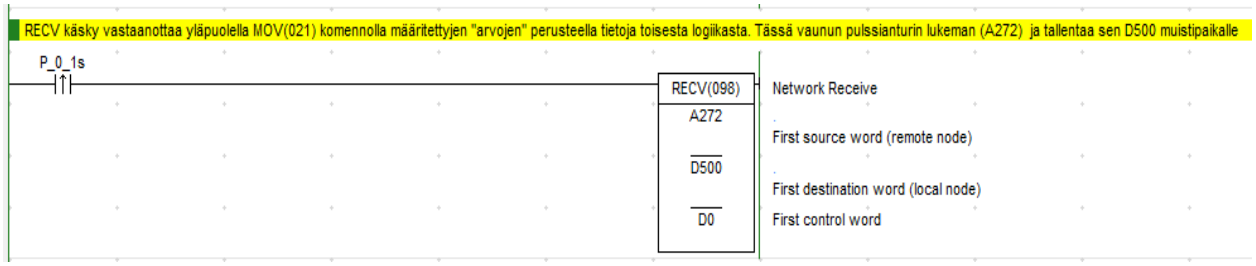
Kuva 34. Pulssiantureiden yhteenlasku



Kuva 35. Imuvaunujen törmäyssuoja

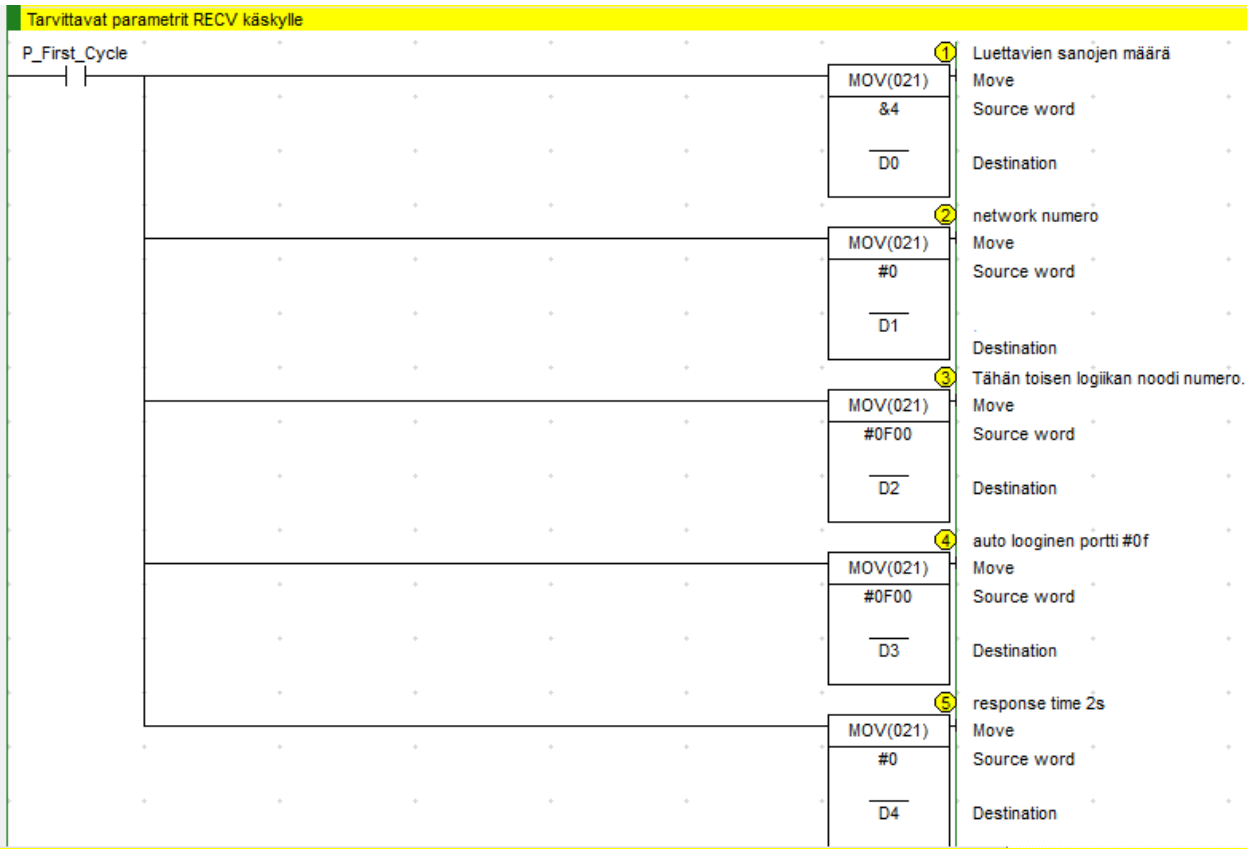
Hankalan törmäyssuojan toteutuksesta teki se, että käytössä oli nyt molemmille imuvaunuille omat ohjauskaappinsa ja ohjauslogiikkansa. Imuvaunujen paikkatieto meni luonnollisesti vain omalle logiikalleen, joten logiikoiden välillä oli saatava kulkemaan tieto toisen imuvaunun pulssianturilta. CX-Programmer ohjelmistossa on tarjolla muutamia eri käskyjä ja toimilohkoja tiedon siirtämiseksi logiikoiden välillä.

Tässä tapauksessa siirrettäväksi tiedoksi riitti vain imuvaunun paikkatieto pulssianturilta, joten käyttöön valittiin vaihtoehtoihin tutustumisen jälkeen toteutettavuudeltaan helpoimman oloinen Network Receive (RECV(098)) komento. Komento pyytää tiettyä dataa siirrettäväksi samasta verkosta löytyvästä IP-osoitteesta.



Kuva 36. Network Receive komento

Network Receive -komennolla määritetään siirrettävä data ja sen muistipaikka uudessa logiikassa, mutta se ei yksin toimi sellaisenaan. Network Receive vaatii toimiakseen tietyt parametrit, jotka asetetaan Move (MOV (021)) -komennon avulla.

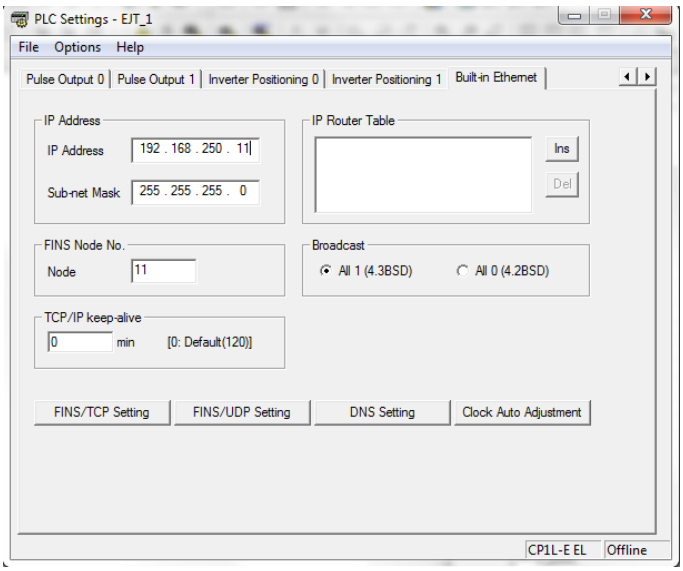


Kuva 37. Tarvittavat parametrit network receive -komennolle

4.6 Tiedonsiirto logiikoiden välillä

Logiikoiden tiedonsiirtoa varten oli logiikoiden välille asennettava Ethernet-yhteys ja verkkokytkin. Jotta tiedonsiirto toimisi oikein, oli logiikoiden tehdasasetuksia muutettava ja logiikat asetettava toimimaan samaan verkkoon muuttamalla niiden IP-osoitteita.

Osoitteiden kolme ensimmäistä numerosarjaa on asetettava samaksi molemmissa logiikoissa. Viimeisin numerosarja asetetaan erilaiseksi ja se yksilöi logiikan sekä toimii samalla tämän ”noodi” numerona. Myös aliverkon peite (Subnet mask) on oltava sama molemmissa logiikoissa.



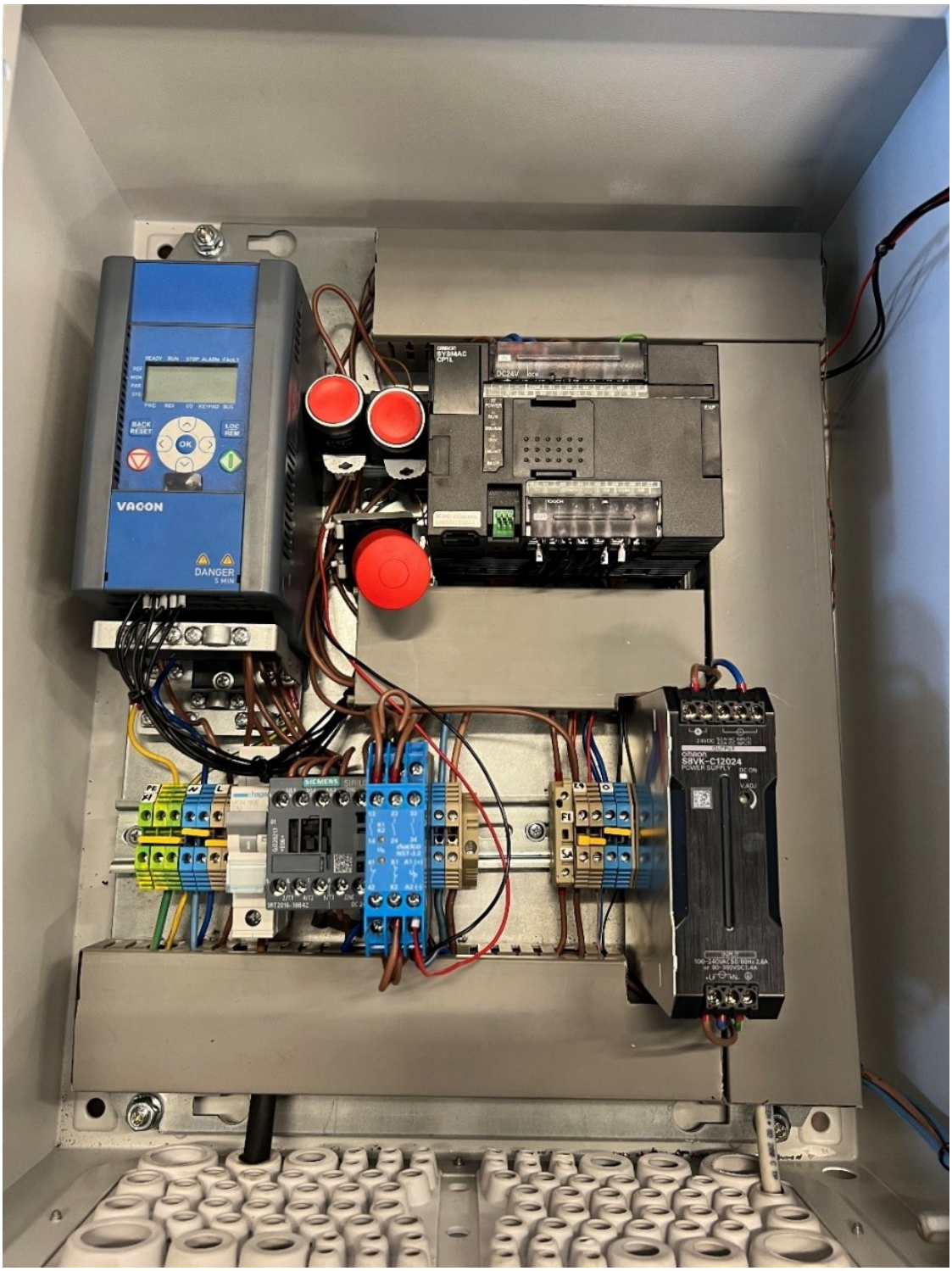
Kuva 38. Logiikan asetukset

Logiikalle asetettu noodi numero on kerrottava ohjelmointi vaiheessa toisen logiikan network receive -komennolle. Sen avulla ohjelma tietää mistä osoitteesta tietoa halutaan siirtää. Noodi numero kerrotaan move -komennon avulla, joka ymmärtää tässä tapauksessa vain heksadesimaali lukuja. Se on siis erikseen muunnettava desimaaliluvusta.

4.7 Testaaminen

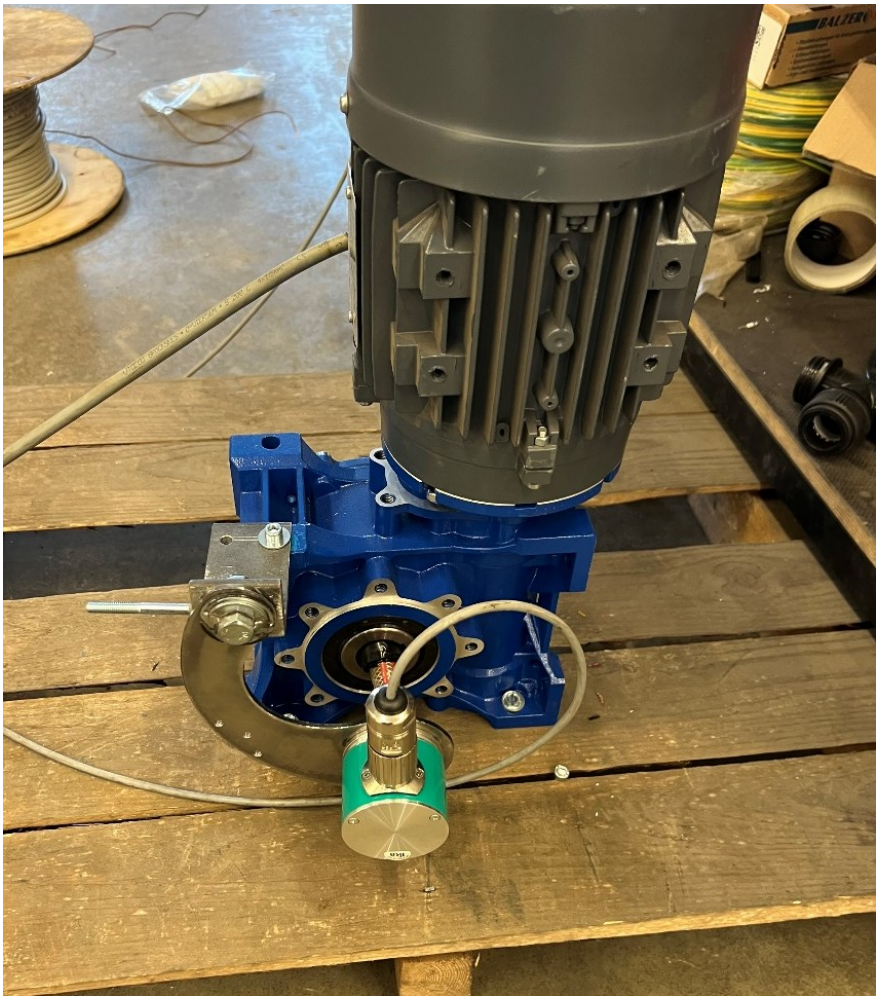
Jo projektin aloitusvaiheessa yrityksen tiloihin rakennettiin testipenkki vanhaan logiikkaohjelmaan tutustumista varten. Tutustumisen jälkeen testipenkkiä muutettiin komponenteiltaan tämän projektin mukaiseksi ja sitä hyödynnettiin paljon jo suunnittelu- ja ohjelmointivaiheessa. Logiikkaa ohjelmoidessa simulointi on toki mahdollista myös pelkällä tietokoneella, mutta ainakin itselleni logiikan ohjelmointi helpottui huomattavasti, kun testipenkissä pääsi heti konkreettisesti näkemään miten tehty muutos vaikuttaa laitteiston toimintaan.

Testipenkki toteutettiin niin, että rakennettiin kaksi erillistä raakaversiota ohjauskaapeista, joihin sijoitettiin mahdollisimman paljon pääkomponentteja, joita oikeastikin tultaisiin käyttämään. Osa lopullisista komponenteista toki korvattiin testausta varten jollain muilla tuotteilla. Esimerkiksi rajakytkimet korvattiin painonapeilla ja kaukosäätimen korvasi aluksi pelkät irralliset painikkeet. Oikosulkumoottorit laitettiin käymään ilman kuormaa ja pulssianturin kiinnitys vaihdelaatikkoon toteutettiin väliaikaisratkaisulla. Leikkauskoneen pulssianturi pidettiin irrallaan, jotta sen liikkumista päästiin simuloimaan manuaalisesti käsin.



Kuva 39. Ohjauskaappi testausvaiheessa

Kuva 40. Imuvaunun kauko-ohjain



Kuva 41. Moottori, vaihdelaatikko ja pulssianturi testipenkissä

Logiikka ohjelmaa ja ohjauslaitteiston yleistä toimintaa testattiin siis jatkuvasti ja useita kertoja projektin edetessä. Kun kaikki osa-alueet oli saatu valmiin oloisiksi, simuloitiin laitteiston toimintaa vielä kerran mahdollisimman realistisesti ja pyrittiin varmistamaan ohjelman toimivuus jokaisessa mahdollisessa tilanteessa. Kaikki saatiin toimimaan halutunlaisesti ja seuraavaksi päästiin suorittamaan laitteiston asennusta oikeaan asiakkaalle tulevaan imuvaunupöytään.

4.8 Käyttöönotto

Testausvaiheen jälkeen väliaikaiset komponentit poistettiin kytkennästä ja ohjauskaapit purettiin testipenkistä. Lisäksi ohjauskaappien kansiin lisättiin tässä vaiheessa pääkytkimet ja merkkilamput, jotka eivät olleet vielä testausvaiheessa käytössä. Imuvaunupöytä on tällä kertaa poikkeuksellisen suuri ja toteutetaan normaalista poikkeavalla runkorakenteella, joten se kasataan lähes alusta alkaen asiakkaan tiloissa. Niinpä myös kaikki kaapelivedot ja komponenttien lopullinen sijoittelu oli jätettävä paikanpäälle tehtäväksi.

Ohjauskaapit pyrittiin sijoittamaan mahdollisimman lähelle moottoreita leikkauspöydän molempiin päihin, jotta taajuusmuuttajan ja moottorin välinen syöttökaapeli pysyisi mahdollisimman lyhyenä. Rajakytkimet asennettiin omille paikoilleen ja kaukosäätimet sekä toiset pulssianturit johdotettiin leikkauskoneiden kylkiin. Imuvaunujen paikkatiedon nollaamiseen käytetyt rajakytkimet asennettiin leikkauskoneen oman nollapisteen mukaisesti niin, että rajakytkimiin ajettaessa imuvanu on automaattisesti kohdistettu keskelle leikkauspoltinta.

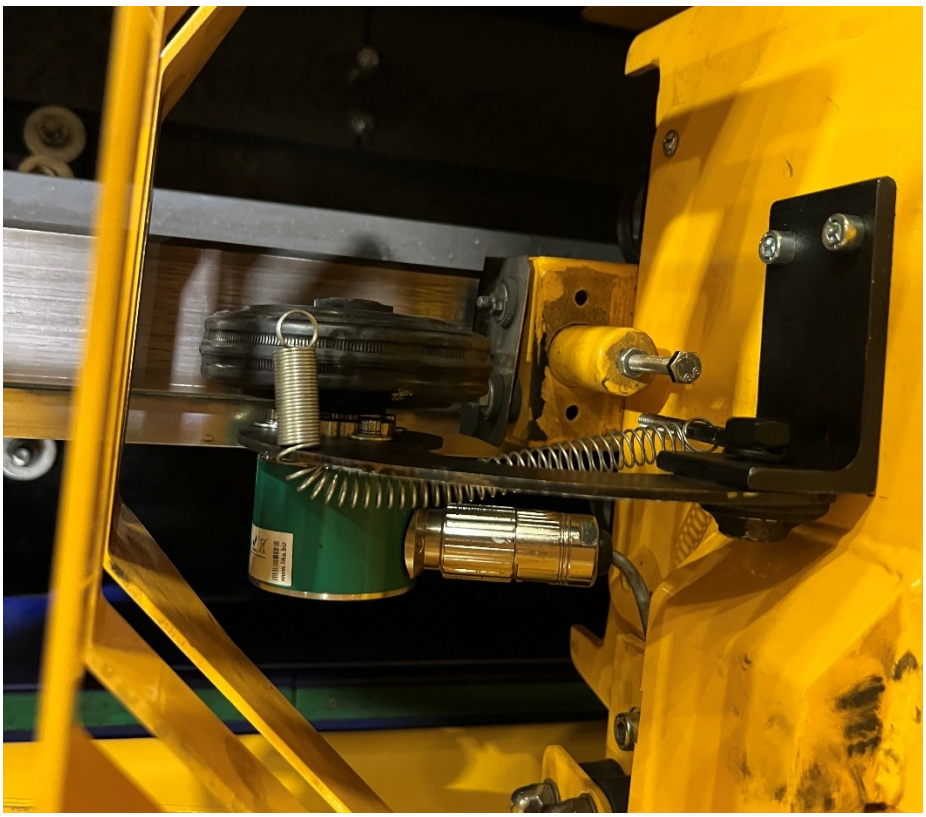
Pulssiantureiden pulssimäärä yhtä anturin kierrosta kohti oli tiedossa, joten paikan päällä oli helppo laskea imuvaunupöydän pituus pulsseina, mittaamalla pöydän toiminnallinen pituus ja leikkauskoneen pulssianturin seurantapyörän kehän pituus. Pöydän pituus pulsseina lisättiin sovellusohjelmaan imuvaunujen törmäyssuoja kohtaan.

Ennen koeajoa oli lopuksi vielä parametroitava taajuusmuuttajat valmiiksi. Taajuusmuuttajille aseteltiin ensimmäisenä moottorin perustiedot eli nimellisjännite, nimellistaajuus, nimellisnopeus, nimellisvirta ja tehokerroin. Perustietojen lisäksi taajuusmuuttajille asetettiin minimi- ja maksimitaajuudet, sekä laitettiin momentin maksimointi päälle. Jotta logiikkaohjelmaan tehdyt ajokäskyt toimisivat oikein, oli taajuusmuuttajalle vielä valittava kolme esiasetettua nopeutta. Nämä nopeusasetukset siis kytkeytyvät päälle taajuusmuuttajien digitaalitulojen mukaan, joita logiikkaohjelma ohjaa.

Lopullista kokoonpanoa koeajettaessa huomattiin, että normaalia suuremman vetoakselin hammaspyörän myötä imuvaunu lähti ajamaan pitkällä matkalla huomattavasti leikkauskoneen edelle. Hammaspyörän ja leikkauskoneen pulssianturin kiekon suhdelukua jouduttiin siis hienosäätämään. Tätä toimintoa varten oli jo vanhaan logiikkaohjelmaan tehty toimilohko, joka siirrettiin nyt myös tähän ohjelman uuteen versioon. Muuten laitteisto toimi halutunlaisesti, eikä isompia muutoksia jouduttu tekemään.

Kuva 42. Imuvaunupöytä

Kuva 43. Imuvaunun moottori, vaihdelaatikko ja pulssianturi



Kuva 44. Leikkauskoneen pulssianturi

Kuva 45. Imuvaunun rajakytkin

Kuva 46. Ohjauskaappi

5 YHTEENVETO

Tämän opinnäytetyön tavoitteena oli suunnitella ja toteuttaa ohjausautomaatio imuvaunupöydälle, jossa työskentelee samanaikaisesti kaksi imuvaunua yhdellä pöydällä. Tilaajalla aikaisemmin käytössä ollut ohjausautomaatio oli suunniteltu toimimaan vain yhdelle imuvaunulle, eikä sitä täten pystytty käyttämään tässä tilanteessa ilman muutoksia. Lopulta vanhaa ohjausautomaatiota pystyttiin vain osittain hyödyntämään mallina imuvaunun ohjauksen perusidean osalta, mutta kaikki muu esimerkiksi logiikkaohjelma tehtiin täysin uudelleen.

Työn tuloksena on toimiva ohjausautomaatio kahden imuvaunun imupöydälle. Tavoitteisiin siis päästiin eli uusi ohjausautomaatio ja logiikkaohjelma saatiin suunniteltua, tehtyä ja otettua käyttöön. Ohjausautomaatio jää yrityksen vakituiseen käyttöön imuvaunupöytien ohjaamiseen, sillä sitä pystytään hyvin käyttämään myös yhden imuvaunun pöydällä pienellä logiikkaohjelman muutoksella.

Opinnäytetyön teoriaosuutta rajattiin tämän projektin oleellisiin komponentteihin. Teoriaosuudessa perehdyttiinkin yleisellä tasolla ohjelmoitaviin logiikoihin ja niiden toimintaperiaatteeseen. Lisäksi tutustuttiin tarkemmin tässä projektissa käytettyihin Omronin valmistamiin ohjainlaitteisiin ja niiden ohjelmointiympäristöön.

Logiikoiden ohjelmointi oli itselleni ehdottomasti haastavin ja työläin osuus tässä projektissa. Minulla ei ollut juurikaan aikaisempaa käytännön kokemusta automaatiosuunnittelusta tai logiikoiden ohjelmoimisesta, joten projekti vaati alkuvaiheessa paljon yleistä aiheeseen tutustumista ja ohjelmointityökalun opettelua. Ohjelmoinnissa erityisesti alkuun pääseminen vei paljon aikaa ja tuntui haastavalta, kun ei ollut tietoinen kaikista ohjelman ominaisuuksista ja vaihtoehdoista.

Lisäksi logiikoiden välinen tiedonsiirto aiheutti paljon päänvaivaa, sillä logiikoiden konfiguroinnista ja ”Network Receive” komennon toimintaan saamisesta tuntui aluksi löytyvän todella huonosti neuvoa CX-Programmerin omista manuaaleista. Lisäksi logiikoiden IP-osoitteiden muokkaamisen jälkeen alkoi myös yhteysongelmat tietokoneen ja logiikoiden välillä, sillä en aluksi ymmärtänyt muokata myös ohjelmointiin käytetyn tietokoneen IP-osoitetta vastaamaan logiikoiden uusia osoitteita ja ratkaisua yhteysongelmaan ehdittiinkin hakemaan hetken aikaa kaikkialta muualta kuin siitä.

Pidin aihetta kuitenkin hyvin monipuolisena ja kiinnostavana, joka helpotti paljon opinnäytetyön tekemisessä. Opinnäytetyötä tehdessä jouduin luonnollisesti perehtymään aiheeseen hyvin laajasti, jonka myötä opin paljon uutta automaatiotekniikasta ja logiikoiden ohjelmoimisesta. Uskon, että tämän opinnäytetyön tekeminen antoi paljon oppia ja tietotaitoa, sekä valmiutta tulevaisuuden työtehtäviin.

LÄHTEET

Omron Oy. (2009). *CX-One ja logiikkaohjelmointi.* Haettu 10. 3. 2023 osoitteesta https://www.myomron.com/downloads/9.local%20material/finnish/cx-one%20ja%20logiikkaohjelmointi%202009_2.pdf

Aimo Tikka, Asko K. Kippo. (25. 3. 2022). *Automaatiotekniikan perusteet, Edita Publishing Oy.* Haettu 10. 3. 2023 osoitteesta https://www.ellibslibrary.com/fi/book/951-37-4912-5

International Electrotechnical Commission. (2013). *IEC STANDARD 61131-3*. Haettu 10. 3. 2023 osoitteesta https://plcopen.org/sites/default/files/downloads/iec_61131-3_preview.pdf

Toimi Keinänen, Pentti Kärkkäinen, Markku Lähetkangas, Matti Sumujärvi. (2007). *Automaatiojärjestelmien logiikat ja ohjaustekniikat, Sanoma Pro Oy.* Haettu 12. 3. 2023 osoitteesta https://www.ellibslibrary.com/book/978-951-0-34365-4

Automaation tietotekniikka. (15. 8. 2011). *Opas toimilohko-ohjelmointiin*. Haettu 11. 3. 2023 osoitteesta https://mycourses.aalto.fi/pluginfile.php/1306423/mod_resource/content/1/FDB_opas2.pdf

W. Bolton (10. 9. 2009). *Programmable Logic Controllers, Elsevier Science & Technology.* Haettu 12. 3. 2023 osoitteesta https://ebookcentral-proquest-com.ezproxy.savonia.fi/lib/savoniafi/reader.action?docID=535058&ppg=124

Omron Oy. (2023). *CP1L Best in class compact machine controller.* Haettu 14. 3. 2023 osoitteesta https://industrial.omron.fi/fi/products/cp1l

Omron Oy. (2023). *CP series CP1L CPU Unit datasheet.* Haettu 14. 3. 2023 osoitteesta https://assets.omron.eu/downloads/datasheet/en/v5/p081_cp-series_cp1l_cpu_unit_datasheet_en.pdf

Omron Oy. (2023). *Automation systems.* Haettu 14. 3. 2023 osoitteesta https://industrial.omron.fi/fi/products/automation-systems

Leea Hiltunen, Petri Mäkikyrö, Antti Rantamäki. (10. 10. 2004.) Ohjelmoitavat logiikat. Haettu 14. 3. 2023 osoitteesta https://heikkilaakso.com/opetus/op/H_1_Ohjelmoitavat_logiikat.pdf

Jaakko Fonselius. (1996). *Koneautomaatio, Automaatiolaitteet. Edita Oy.* Haettu 15. 3. 2023

Danfoss Oy. (2023). *VACON 20.* Haettu 20. 3. 2023 osoitteesta https://www.danfoss.com/fi-fi/products/dds/low-voltage-drives/vacon-drives/vacon-20/#tab-overview

Lika Electronics Oy. (2023). *I58-I58S.* Haettu 20. 3. 2023 osoitteesta https://www.lika.it/eng/products/rotary-encoders/incremental/i58-i58s#

Duelco Safety Solutions. (2023). *Emergency stop relay NST-3.2.* Haettu 20. 3. 2023 osoitteesta https://www.duelco-safety.com/en/product/id-different-versions/emergency-stop-relay-nst-32

Omron Oy. (2023). *S8VK-S datasheet.* Haettu 20. 3. 2023 osoitteesta https://assets.omron.eu/downloads/datasheet/en/v14/t205_s8vk-s_switch_mode_power_supply_datasheet_en.pdf

AirWell Oy. (2023). *Referenssit, imuvaunu.* Haettu 20. 3. 2023 osoitteesta https://airwell.fi/referenssit/

Moves Oy. (2023). *Moves moottorit*. Haettu 21. 3. 2023 osoitteesta https://moves.fi/moves-moottorit/

Omron Oy. (2008). *CX-Simulator introduction guide.* Haettu 29. 3. 2023 osoitteesta https://www.tecni-cal.cat/PDF/OMRON/Software/CX-One/R151-E1-02.pdf