jamk.fi

Hissisimulaattorin automatisointi

Markus Roiha

Opinnäytetyö
Syyskuu 2021
Tekniikan ja liikenteen ala
Insinööri (AMK), Sähkö- ja automaatiotekniikan tutkinto-ohjelma
Automaatiotekniikka

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Roiha, Markus	Opinnäytetyö, AMK	Syyskuu 2021
	Sivumäärä 46	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: x

Työn nimi
Hissisimulaattorin automatisointi

Tutkinto-ohjelma
Sähkö- ja automaatiotekniikka

Työn ohjaaja(t)
Juho Riekkinen

Toimeksiantaja(t)
KONE Oyj

Tiivistelmä

Työn aiheena oli KONE Oyj tuotekehityksessä käytetyn hissin ohjelmiston turvaominaisuuksien testaukseen käytetyn simulaattorin automatisointi. Hissin ohjelmiston turvaominaisuuksien testaaminen vaatii paljon manuaalista työtä, mikä aiheuttaa virheiden ja epäluotettavuuden lisääntymistä. Tämä taas vähentää testaamisen tehokkuutta ja nopeutta. Simulaattoriin oli myös tullut ajan saatossa lisää johdotuksia ja testattavia piirikortteja, joten kaappitila alkoi uusia päivityksiä silmällä pitäen loppumaan.

Automatisoinnin suunnittelu aloitettiin tekemällä ohjelmistotestausta vanhalla kokoonpanolla. Tällä saavutettiin hyvä näkemys siitä, miten testejä täytyy suorittaa ja mitä pitäisi automatisoida. Ohjelmistotestauksen aikana tuli myös selväksi, mitkä funktiot olisivat turhia. Näin ollen ne voitiin karsia pois automatisointiprosessista.

Testauksen jälkeen määriteltiin tarvittava laitteisto automatisointia varten. Laitteistoksi valittiin Beckhoffin valmistama logiikka, mikä määritteli myös kommunikoinnissa käytettävän tietoliikenneväylän tyypin ja ohjelmointiin tarvittavan ohjelmiston.

Viimeisenä vaiheena oli tehdä automatisoinnissa tarvittava ohjelmisto ja rakentaa se mahdollisimman yksinkertaiseksi, jotta manuaalista työtä ei testauksen aikana tarvitse tehdä. Tämän lisäksi täytyy rakentaa käyttöliittymä, jolla simulaattoria ohjataan ja joka näyttää tarvittavan tiedon testaajalle.

Työn tuloksena oli osittain valmis ohjelmisto simulaattoria varten. Simulaattoria varten tarvittavien osien pitkien toimitusaikojen takia ei fyysinen laitteisto ehtinyt valmistua. Tämä tarkoitti myös sitä, että fyysistä ohjelmiston testausta ei voitu tehdä.

Avainsanat (asiasanat)
Beckhoff, servo-ohjain, lineaarimoottori, automaatio, hissi, logiikka, suunnittelu, PLC

Muut tiedot (Salassa pidettävät liitteet)

jamk.fi

Description

Author(s)	Type of publication	Date
Roiha, Markus	Baschelor's thesis	September 2021
		Language of publication: Finnish
	Number of pages 46	Permission for web publication: x

Title of publication
Elevator simulator automation

Degree programme
Electrical and automation engineering

Supervisor(s)
Juho Riekkinen

Assigned by
KONE Oyj

Abstract

The topic of the work is the automate elevator simulator used to test the safety features of the elevator software used in KONE Corporation's product development. Testing the security features of the elevator software requires a lot of manual work, which causes an increase in errors and unreliability. This will reduce the efficiency and speed of testing. Added wirings and circuit boards had also been added to the simulator over time, so the cabinet space began to run out with new upgrades in mind.

Planning for automation was started by conducting software testing on the old configuration. Based to the test results, achieved a good view of how tests need to be performed and what should be automated. During software testing, it also became clear which functions would be pointless. Consequently, they could be pruned out of the automation process.

After testing, the necessary hardware for automation was specified. Logic manufactured by Beckhoff was chosen as the hardware, which also defined the type of communications bus used in communication and the software required for programming.

The final step was to make the software needed for automation and build it as simple as possible so that manual work does not have to be done during testing. In addition to this, you need to build an interface to control the simulator and display the necessary information to the tester.

The result of the work was partially finished software for the simulator. Due to the long delivery times of parts needed for the simulator, the physical hardware did not have time to be completed. This also meant that physical software testing could not be done.

Keywords/tags
Beckhoff, servo-drive, linear motor, automation, elevator, logic, design, PLC

Miscellaneous (Confidential information)

Sisältö

Kuviot

Taulukot

Lyhenteet ja käsitteet

A	Ampeeri (Sähkövirran yksikkö).
AC	Alternating current (Vaihtovirta).
BOOL	Boolean tietotyyppi. Kaksi totuusarvoa. False (0) ja True (1).
CFast	CompactFlash. Muistikorttityyppi.
CFC	The continuous function chart editor (Graafinen ohjelmointikieli).
CPU	Central processing unit (Keskusyksikkö).
DC	Direct current (Tasavirta).
DI	Digital input (Digitaalinen tulo).
DO	Digital output (Digitaalinen lähtö).
DVI-I	Digital visual interface – Integrated. Kuvasignaalin siirtämiseen tarkoitettu liitäntä.
FBD	Function block diagram (Graafinen ohjelmointikieli).
GHz	Gigahertsi. Kuvaa tietokoneen komponenttien kellotaajuutta.
HMI	User interface (Käyttöliittymä).
I/O	Input/Output (Tulo/Lähtö).
IL	Instruction list (Tekstipohjainen ohjelmointikieli).
Intel Atom	Intelin vuonna 2008 esittelemä vähävirtainen suoritinperhe ja prosessori eli keskusyksikkö.
LD	Ladder diagram (Graafinen ohjelmointikieli eli tikapuukaavio).
mA	Milliampeeri (Sähkövirran yksikkö).
MicroSD	Micro Secure Digital. Muistikorttityyppi.
PC	Personal computer eli tietokone
PLC	Programmable logic controller (Ohjelmoitava logiikka).
RJ45	parikaapeli, joka on yleinen sähköisten signaalien siirtämiseen käytetty kaapelityyppi.
RS232	Tietoliikenteeseen tarkoitettu tietoliikenneportti
SFC	Sequential function chart (Graafinen ohjelmointikieli).
ST	Structured text (Tekstipohjainen ohjelmointikieli).
USB 2.0	Universal serial bus. Sarjaväylä oheislaitteiden liittämiseksi tietokoneeseen.
V	Voltti (Sähköjännitteen yksikkö)

1 Johdanto

Tuotekehitys on yrityksen tärkein vaihe, kun markkinoille halutaan tuoda tarjolle uusi tuote tai palvelu. Tuotteen tai palvelun tarkoituksena on vastata asiakkaiden kysyntään ja tarjoamalla sopivaa tuotetta yrityksen tavoite on menestyä markkinoilla. Tuotekehityksen aikana palvelua tai tuotetta testataan ja suunnitellaan käytettäväksi parhaalla mahdollisella tavalla ja prosessin aikana voi mahdollisesti syntyä uusia innovaatioita. Olemassaolevaa tuotetta pitää myös jatkuvasti kehittää eteenpäin, jotta vastataan asiakkaan kysyntään ja yritys pysyy mukana kilpailussa, jossa kilpailijat tarjoavat samaa tuotetta tai palvelua paremmilla ominaisuuksilla.

Tuotetta tai palvelua on testattava, jotta mahdolliset suunitteluvirheet tulevat esille ennen tuotteen saattamista markkinoille. Huono tai olematon tuotetestaus saattaa altistaa käyttäjänsä vaaraan. Nämä mahdolliset vaaraan altistavat ominaisuudet pyritään löytämään tuotetta suunitellessa sekä tuotetestauksen aikana. Tästä huolimatta markkinoille on päätynyt useita tuotteita ja palveluita, jotka ovat aiheuttaneet käyttäjälleen ongelmia tai jopa hengenvaaraa.

1.1 Opinnäytetyön tausta ja tavoite

Opinnäytetyön toimeksiantajana KONE Oyj tuotekehitysyksikkö Hyvinkäällä. KONE Oyj on suomalainen hissejä, liukuportaita ja automaattiovia valmistava yritys sekä tarjoaa näille huolto- ja modernisointipalveluita. Tuotekehitysyksikön tarkoitus on kehittää uusia tuotteita ja palveluita, tehdä parannuksia nykyisiin tuotteisiin ja palveluihin, testata osien kestävyyttä sekä toimintaa ja analysoida olemassa olevien tuotteiden äkillisesti ilmaantuvia tai tiedossa olevia ongelmia.

Yksikössä on käytössä hissisimulaattori, jossa testataan hissin uusien ohjelmistoversioiden turvatoimintoja tilanteessa, jossa hissikori hitaasti tai nopeasti karkaa kerrosalueelta ovien ollessa auki. Simulaattori on vuosien saatossa palvellut hyvin, mutta uusien tuotteiden takia se on laajentunut uusilla korteilla ja johdotuksilla. Tämän takia vanha kytkentäkaappi on jäänyt liian pieneksi eikä kaikkia

laitteita saada enää mahtumaan kaapin sisälle. Tämä on aiheuttanut sen, että testiympäristö ei ole enää siisti eikä turvallinen.

Toimeksiantona on suunnitella ja rakentaa hissisimulaattorille parempi, nykyaikaisempi ja turvallisempi testiympäristö, jossa on myös huomioitu tulevaisuuden laajennusvara sekä saada testaaminen nopeammaksi ja luotettavammaksi. Ohjelmistotestaamisen tärkeimpänä tavoitteena on saada automatisoitua testien teko mahdollisimman pitkälle, kun tällä hetkellä testien teko on pitkälti manuaalista. Mitä pidemmälle simulaattoria saadaan automatisoitua, sitä vähemmän testaajasta aiheutuvien virheiden ja epäluotettavuuden määrä vähenee.

1.2 Opinnäytetyön rajaus

Opinnäytetyössä halutaan vastaukset kolmeen hissisimulaattoriin liittyvään kysymykseen, jotka samalla rajaavat työtä.

1. Mitä hissisimulaattorissa halutaan automatisoida?
2. Miksi hissisimulaattori halutaan automatisoida?
3. Miten hissisimulaattorin automatisoidaan?
    1. Laitteiston määrittely.
    2. Logiikkaohjelman määrittely.

1.3 Tutkimus ja kehitysmenetelmät

Tyypiltään opinnäytetyö on kehittämistutkimus. Kehittämistutkimusta tehtäessä taustalla on jokin ilmiö, asia tai prosessi, jota halutaan kehittää. Kehityksen jälkeen kehitettävä kohde toimisi paremmin. Opinnäytetyössä tullaan käyttämään tutkimusmenetelminä laadullisen tutkimuksen tiedonkeruumenetelmiä. Tietoperustana käytetään mahdollisesti kirjallisia lähteitä, kuten valmistajien manuaaleja ja yrityksen sisäisiä toimintakuvauksia. Erityisesti Beckhoffin materiaalit olivat tärkeitä tämän työn kannalta sekä Beckhoffin tarjoamaa ohjelmointiin liittyvää peruskurssia hyödynnettiin työssä. Näiden lisäksi tullaan käyttämään osallistuvaa havainnointia ja strukturoimatonta haastattelututkimusta eli avointa haastattelua. Kehittämistyöhön liittyy kokouksia erilaisilla kokoonpanoilla ja haastateltavina käytetään toimeksiantajan henkilökunnan lisäksi ulkopuolista henkilöstöä.

2 Toimeksiantaja

2.1 KONE Oyj

KONE Oyj on suomalainen yritys ja se on yksi maailman johtavista yrityksistä, joka valmistaa hissejä, liukuportaita ja automaattiovia sekä tarjoaa kunnossapidon ja modernisoinnin palveluita. Se on perustettu Helsingissä alun perin Suomen vielä ollessa Venäjän keisarikunnan alla vuonna 1908 konepajana nimeltä Tarmo ja jo vuonna 1910 nimi on muutettu KONE osakeyhtiöksi. Ensimmäisien vuosian aikana yhtiö valmisti ja korjasi sähkömoottoreita sekä toi maahan ja asensi ruotsalaisia Graham Brothersin hissejä. Yhtiön pääkonttori sijaitsee Helsingissä.

Missiona KONEella on tehdä kaupungeista parempia paikkoja elää ja täten tarjota käyttäjälle paras käyttäjäkokemus. Pääsemällä tähän KONE kehittää jatkuvasti ratkaisuja ja palveluita palvellakseen käyttäjiä paremmin kaupungistuvassa ympäristössä. Mainitsemisen arvoisena vuonna 1996 KONE julkisti maailman ensimmäisen konehuoneettoman MONOSPACE®-hissin, josta myöhemmin on tullut osa hissiteollisuuden standardia. Kyseinen innovaatio oli tuohon aikaan ylivoimaisesti ympäristöystävällisin hissi. Tänä päivänä suurin osa uusien kerrostalojen hisseistä onkin konehuoneettomia. (KONE historia)

Vuonna 2020 KONEen liikevaihto oli 10 miljardia euroa ja työntekijöitä yrityksessä on päälle 60 000 henkilöä. Yhtiön b-sarjan osake noteerataan Nasdaq Helsinki Oy:ssä. Toimintaa on yli 60 eri maassa. (KONE osavuosikatsaus 2020.)

2.2 Tuotekehitysyksikkö

KONEella on kahdeksan tuotekehitysyksikköä ympäri maailman. Suomessa tuotekehitysyksikköjä on Hyvinkäällä sekä Lohjan Tytyrissä, jossa jälkimäisessä on korkean rakentamisen testilaboratorio eli toisin sanoen pilvenpiirtäjissä käytettyjen hissien testaamista ja kehittämistä. Tytyrissä on maailman syvin hissien testaukseen tarkoitettu yksikkö. Syvimmän hissikuilun nostokorkeus on 305m.

Muita yhtiön tuotekehitysyksikköjä löytyy Yhdysvalloista kaksi, Meksikosta, Intiasta, Saksasta, Italiasta ja Kiinasta. Kiinan Kunshanissa on 235m korkeuteen yltävä testitorni, jossa on kahdeksan hissikuilua ja 36 kerrosta. Täällä testataan Lohjan Tytyrin tavoin pilvenpiirtäjiin tarkoitettua hissiteknologiaa. (KONEen hissilaboratoriot.)

3 Hissisimulaattori

3.1 Hissi

Hissi on ihmisiä tai tavaraa kuljettava nostolaite, joka kulkee kerrosten välillä johteita pitkin. Hissi kulkee yleensä pystysuunnassa, mutta myös muita sovellutuksia on rakennettu. Hissin kyky kuljettaa matkustajia ja rahtia nopeasti eri kerrosten välillä on tuonut mahdollisuuden rakentaa toinen toistaan korkeampia rakennuksia. Nykyaikana hissillä on tärkeä tehtävä kuljettaa ihmis- ja tavaramassoja kaupungistuvassa maailmassa. Ilman hissejä ihmisten liikkuminen korkeisiin rakennuksiin olisi aikaa vievää ja raskasta ellei jopa mahdotonta.

Nykyaikaisen turvallisen hissin voidaan sanoa saaneen alkunsa vuonna 1853, kun Elisha Otis esitteli hissikoriin asennettavan tarraajan. Tarraajan tarkoitus on tarrata hissi kiinni johteisiin, jos hissi jostakin syystä sattuisi putoamaan tai kulkemaan liian nopeasti.

Nykypäivän köysihissin toimintaperiaate on hyvin samanlainen kuin Elisha Otisin aikoihin. Hissi sijaitsee yleensä omassa hissikuilussa, jossa se koostuu hissikorista ja vastapainosta, joita kannattelee joukko kannatinköysiä- tai hihnoja. Sähkömoottori liikuttaa hissiä kerrosten välillä. Sähkömoottoria ohjataan joko suoraan sähköverkosta tai taajuusmuuttajan avustuksella. Hissin ohjausjärjestelmä ohjaa taajuusmuuttajaa sekä muita hissin toimintoja. Jokaisella kerrostasolla on hissikuilun ovet, jotka estävät putoamasta hissikuiluun. Moderneissa hisseissä on myös korinovet, jotka suojaavat hissikorissa olevia matkustajia. Hissikoria voidaan myös liikuttaa suoraan

tai köysivälitteisesti hydraulisylinterien avustuksella. Tällöin hissiä kutsutaan hydraulihissiksi. (Britannica Elevator n.d.)

3.2 Hissisimulaattorin rakenne

Hissisimulaattorin tarkoituksena on tuoda ja jäljitellä tavallisen hissin toiminnot huomattavasti kustannustehokkaammin pienemmässä tilassa. Tämä tarkoittaa sitä, että hissin ohjelmistoa voidaan testata huomattavasti nopeammin ja turvallisemmin, kuin jos testausta suoritettaisiin oikeassa hissiympäristössä. Oikeassa hissiympäristössä liikuteltavat massat olisivat huomattavasti suurempia, koska massat ovat satoja tai tuhansia kiloja ja korkeudet peräti kymmeniä metrejä. Se tarkoittaisi korkeampia kustannuksia testejä tehtäessä. Myös turvallisuuden ylläpitäminen oikeassa hissiympäristössä on vaikeampaa edellä mainittujen asioiden takia.

Hissisimulaattori sisältää hissin kokonaisen ohjauskeskuksen, sähkömoottorin ja tämän ohjaukseen käytetyn taajuusmuuttajan. Hissikuilu, kori ja vastapaino on korvattu ja simuloitu leveällä hammashihnalla, jossa on kiinni paino sekä kelkka. Kelkan on tarkoitus simuloida hissikoria, jossa on kiinni sijainnin määritykseen tarvittava laitteisto. Simulaattorissa on kolme kerrosta, joista toisen kerroksen sijainnin määritykseen tarkoitettua magneettia liikutellaan lineaarimoottorilla. Hammashihnaa, jossa kelkka on kiinni, liikutetaan sähkömoottorilla. Hihnan ja moottorin välistä välityssuhdetta muutetaan paremmin simulaattoriin sopivaksi hidastaen hihnan ja kelkan nopeutta. Simulaattorin rakenne käy ilmi kuviosta 1. Lineaarimoottori näkyy kaapin sisällä keskellä kaappia hammashihnan vieressä.

Kuvio 1. Hissisimulaattori.

3.3 Hissisimulaattorin tarkoitus

Hissisimulaattorilla on pääasiallinen tarkoitus testata hissin ohjausjärjestelmän turvaominaisuuksia ja reagointinopeutta kuvitteellisessa tilanteessa, jossa hissikori ovien ollessa auki karkaa kerroksesta ollessaan pysähtyneenä tai tasausajon aikana. Tasausajo tarkoittaa ajoa, missä hissin kuorman ja kannatinköysien pituuden muuttuessa, hissikorin lattia ei ole enää tasan kerroksen lattian suhteen vaan näiden välissä on kynnys, joka taas voi aiheuttaa kompastumisvaaran. Tällöin hissi yrittää korjata kynnystyksen ajamalla hitaasti hissikoria haluttuun suuntaan ovet auki. Koska hissikorin tai simulaattorin tapauksessa kelkan nopea liikuttaminen pois kerrosalueelta on haastavaa, on kerrosalueelta karkaaminen toteutettu lineaarimoottorilla. Lineaari-

moottorin tarkoitus on toistettavasti ja säädettävästi liikuttaa nopeasti hissin paikannukseen liittyvää kerrosmagneettia pois siihen tarkoitetulta alueelta ja saada täten hissin turvatoiminnot aktiiviseksi. Lineaarimoottorin ohjaus on toteutettu pienillä vipukytkimillä. Vipukytkimien käyttäminen vaatii tarkkaa ajoitusta, jotta haluttu vaaratilanne saadaan tehtyä ja johon hissin logiikan halutaan reakoivan.

Simulaattorilla testataan myös hissin ohjelmiston reaktioaikoja turvapiirin auetessa pysäyttäen hissin välittömästi. Turvapiiri tarkoittaa hissiin sijoitettuja kytkimiä, magneetteja ja koskettimia, jotka valvovat turvallisuutta koskevia asioita. Tallaisia ovat muun muassa erilaiset rajakytkimet, ovien lukitusta valvovat koskettimet ja lisälaitteita valvovat sensorit.

Testejä suoritettaessa, on annettu mitattaville turvatoiminnoille määritetyt toiminta-ajat, joiden sisällä turvalaitteen täytyy toimia. Esimerkiksi tietyn turvalaitteen toimiessa, mitataan miten nopeasti hissin jarrulta katoaa jännite. Nämä toiminta-ajat mitataan erillisellä mittalaitteella, mihin ei tässä työssä syvennytä enempää.

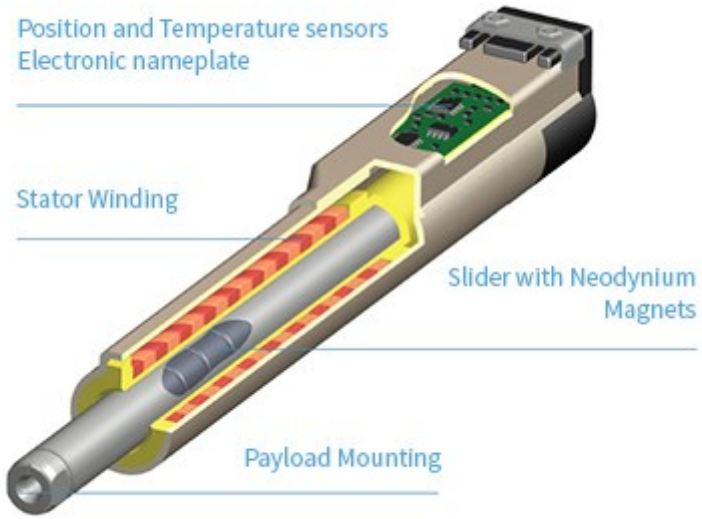
Hissisimulaattoriin halutaan liittää ohjelmoitava logiikka, jotta testaaminen voitaisiin automatisoida ja nopeuttaa mahdollisimman pitkälle. Samalla täytyy uusia lineaarimoottorille tarkoitetun servo-ohjaimen tyyppi sellaiseksi, joka toimii tietoliikenneväylän kautta. Väylään liitetty servo-ohjain säästää tilaa, koska vipukytkimiä ei tarvitse kokoonpanossa käyttää. Myös servo-ohjaimen parametreja ei tarvitse testien välissä muutella, koska halutut arvot ohjelmoidaan testeihin ja välitetään väylän kautta ohjaimelle. Väylän käyttö siis mahdollistaa myös monipuolisemman lineaarimoottorin ohjaamisen.

3.4 Lineaarimoottori

Lineaarimoottori on sähkömoottori, jossa liike muutetaan lineaariseksi pyörimisliikkeen sijasta. Ratkaisulla voidaan helposti korvata aikaisemmin pyörivällä liikkeellä ja vaihteistolla aikaiseksi saatu lineaarinen liike suoraan lineaariseksi, jolloin ratkaisusta saadaan yksinkertaisempi.

Rakenteeltaan työssä käytettävä LinMot lineaarimoottorit ovat kestomagneettiservomoottoreita, joka mahdollistaa moottorille suuret kiihtyvyydet sekä nopeudet ilman mekaanista kulumista. Moottorit koostuvat liu'usta ja staattorista, joka näkyy kuviossa 2. Liuku on ruostumattomasta teräksestä tehty putki, jonka sisällä on neodymium magneetti. Staattori sisältää moottorin käämitykset, laakerit liu'ulle ja mikroprosessorin, joka valvoo moottorin toimintaa ja asemaa mittaavan sensorin. Paikkatietoa mittaava sensori mittaa ja valvoo moottorin asentoa moottorin ollessa pysähdyksissä ja liikkeessä. Tämän ansiosta asennon poikkeavuus huomataan välittömästi.

Lineaarimoottori on hyvä vaihtoehto pneumaattisen toimilaitteen korvaamiseen. Tällöin voidaan luopua paineilmalinjan rakentamisesta, ei tarvitse huolehtia mahdollisista vuodoista ja energiaa kuluu ainoastaan työliikkeen aikana. Lineaarimoottorin rakenteen takia voidaan liikkeestä tehdä myös erittäin tarkka sekä kiihtyvyyttä ja nopeutta voidaan tarvittaessa muuttaa nopeasti ja tarkasti. (LinMot linear motor 2021.)



Kuvio 2. LinMot lineaarimoottorin rakenne. (LinMot, linear motor 2021).

3.5 Servo-ohjain

Servo-ohjain on lineaarimoottorin paikoitukseen liittyvä ohjain. Ohjain hoitaa kaikki moottorin virran syöttöön, liikuttamiseen ja asemointiin liittyvät asiat sekä vastaanottaa ohjaimelle tuodut komennot, jotka se välittää eteenpäin lineaarimoottorille. Lineaarimoottorin toteuttaessa halutut liikkeet, on moottorin ja ohjaimen välissä takaisinkytkentä, minkä avulla moottori kertoo takaisin ohjaimelle muun muassa sijaintinsa. Ohjaimia löytyy erilaisilla kokoonpanoilla, ominaisuuksilla ja liitännöillä, mutta perusperiaate kaikissa on sama eli ohjata moottoria. (LinMot Servo Drives 2021.)

3.6 Ohjelmoitava logiikka

Ohjelmoitava logiikka eli PLC on ohjelmoitava tietokone, jota käytetään eri prosessien automatisoinnissa. Ennen kuin ohjelmoitavia logiikoita oli olemassa, automatisointi hoidettiin releteauluilla. Halutut toiminnot saatiin aikaan kosketin- ja aikareleiden eri kokoonpanoilla. Releteaulujen ongelma oli hankala vianetsintä, ne veivät paljon tilaa ja muutosten tekeminen oli työlästä. 1960-luvun lopulla esiteltiin PLC, jonka tarkoituksena oli korvata releteaulut. Nykyään ohjelmoitavalla logiikalla korvataan pieniäkin sovellutuksia ja releteknikkaa käytetään ainoastaan kaikista yksinkertaisimmissa sovelluksissa. Tunnettuja PLC:n valmistajia ovat mm. Siemens, Omron, Beckhoff ja Allen-Bradley.

Havainnollistavana esimerkkinä automatisoitavasta kohteesta voidaan käyttää vesisäiliön täyttämistä. Normaalisti henkilö avaa säiliöön menevän vesihanan, kun vettä on tarpeeksi vähän ja sulkee hanan veden saavuttaessa haluttu määrä. Automatisoinnilla logiikka huolehtii annettujen parametrien mukaisesti säiliön veden määrästä täysin automaattisesti ilman ulkopuolisen valvontaa ympäri vuorokauden. Näin ollen henkilö voi käyttää säiliön täyttämiseen käytetyn aikansa muuhun.

Ohjelmoitava logiikka koostuu keskusyksiköstä ja siihen integroiduista tai siihen liitettävistä tulo- ja lähtöporteista eli terminaaleista. Näistä käytetään nimitystä tulo- ja lähtö sekä I/O, joka on lyhenne sanoista input ja output. Tulo- ja lähtöportteihin voidaan kytkeä digitaalisia tai analogisia signaaleja. Digitaalinen signaali on verrattavissa

releeseen, joka on päällä tai pois päältä ja se käyttäytyy samoin kuin kytkin. Analogista signaalia voidaan verrata äänenvoimakkuus säätimeen. Säädintä pyörittäessä säätimeltä tulee eri arvoja toisin kuin digitaalinen signaali, joka on joko päällä tai pois päältä. Analogista signaalia käyttäviä laitteita ovat esimerkiksi lämpötila-, paine- ja virtausanturit. (What is a programmable logic controller 2015.)

Usein mahdolliset mittauspisteet tai ohjattavat laitteet ovat hajautetusti kaukana keskusyksiköstä. Tällöin terminaalien ja keskusyksikön liittäminen toisiinsa tapahtuu tiedonsiirtoon soveltuvan kaapelin avulla. Laitteiden liittämistä toisiinsa sanotaan kenttäväyläksi. Eri valmistajilla on omia kenttäväylätekniikoita, mutta perusperiaate kaikissa tekniikoissa on sama eli kenttäväylän avulla siirretään tietoa terminaalien ja keskusyksikön välillä. Kenttäväylän etuna on, että jokaiselle ohjattavalle laitteelle tai luettavalle signaalille ei tarvitse viedä omaa johtoa keskusyksiköltä asti vaan tämä korvataan yhdellä kenttäväylään soveltuvalla kaapelilla, jossa kulkee digitaalisessa muodossa jopa satojen antureiden ja laitteiden tietoa. Tämä yksinkertaistaa huomattavasti asennusta ja säästää kustannuksissa. (Tutorial – Introduction to Fieldbus 2021.)

3.6.1 EtherCAT

EtherCAT (Ethernet for Control Automation Technology) on Beckhoff Automation GmbH:n kehittämä ja IEC 61158 standardisoitu automaatiojärjestelmien kenttäväylä-protokolla. Vuonna 2003 perustettiin The EtherCAT Technology Group (ETG) organisaatio EtherCAT-tekniikalle. Organisaation päätavoite on pitää EtherCAT-väylän pitäminen vakaana ja yhteentoimivana, mikä tarkoittaa kyseisen tekniikan kehittämistä ja levittämistä.

EtherCAT perustuu Ethernet-verkkotekniikkaan. Ethernet on yleisin käytettävissä olevista verkkotekniikoista erityisesti toimistopuolella, mutta se ei ole paras ratkaisu prosessiteollisuuteen. Ethernetin heikkoudet tulevat esiin lähettäessään lyhyitä viestejä ja kytkennässä käytetty tähtitopologia on epäedullinen kaapeloinnin suhteen. Jokainen solmukohta verkossa vaatii myös erillisen ohjelmaa suorittavan laitteen ja monitasoinen sarjakytkentä aiheuttaa riippuvuuksia kommunikointiin. Nämä kaikki

aiheuttavat viiveitä tiedonsiirtoon, mikä on prosessiteollisuuden tärkeimpiä vaatimuksia.

EtherCAT on kehitetty ratkaisemaan Ethernetin ongelmia prosessiteollisuudessa. Se ei tarvitse kytkimiä tai reitittimiä, jos väylässä on ainoastaan EtherCAT-laitteita. Laitteet yhdistetään suoraan toisiinsa väyläkaapelin avulla. Kenttäväylän tiedonsiirtoa on optimoitu nopeammaksi saaden viiveajat lyhyeksi ja teollisuuden sovelluksiin paremmin soveltuvaksi. Yleistasolla EtherCAT-väylässä on yksi master-laite, joka lähettää yhden kehyksen väylään slave-laitteille. Slave-laitteet pystyvät ainoastaan lukemaan ja kirjoittamaan lennosta niille tarkoitettuun lähetettyyn kehykseen ja välittämään kehystä eteenpäin muille slave-laitteille ja takaisin. Tämän ansiosta arvaamattomat viiveet saadaan poistettua ja taataan laitteiden reaaliaikavaatimukset. (Beckhoff EtherCAT 2021.)

3.6.2 TwinCAT 3

TwinCAT 3 (The Windows Control and Automation Technology) on Beckhoff Automation GmbH tarjoama kehitysympäristö Windows tietokoneille. Ohjelmalla on tarkoitus ohjelmoida ja hallita Beckhoffin ohjelmoitavia logiikoita. TwinCAT 3 integroituu osaksi Microsoftin Visual Studiota ja ohjelmointiympäristö on tuttu henkilöille, jotka ovat Visual Studiota käyttäneet. TwinCAT 3 voidaan asentaa ja integroida olemassa olevan Visual Studion päälle tai jos Visual Studiota ei löydy ennestään, voidaan TwinCAT 3 asentaa ilman Visual Studiota. Tällöin asennuksen yhteydessä asennetaan Visual Studion Shell-versio, joka mahdollistaa TwinCAT 3 projektien kehittämisen, mutta ei Visual Studiolla tehtäviä ohjelmointeja kuten Visual Basicia. (Beckhoff TwinCAT 2021.)

TwinCAT ympäristössä ohjelmointi on mahdollista standardin IEC61131-3 mukaisilla ohjelmointikielillä, jotka on listattu alla

- FBD (Function Block Diagram)
- IL (Instruction list)
- LD (Ladder Logic Diagram)
- SFC (Sequential Function Chart)
- ST (Structured Text)

Standardissa esiintyvien kielien lisäksi ohjelmointi on mahdollista CFC – kielellä (Continous Function Chart)

4 Ohjausjärjestelmän suunnittelu ja valinta

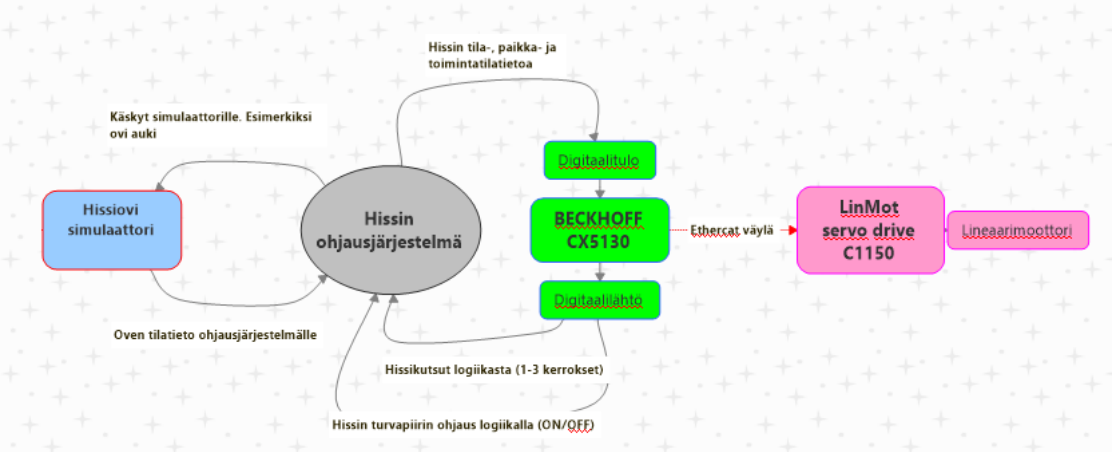
4.1 Simulaattoriin tutustuminen

Simulaattoriin ja sen toimintoihin tutustuminen aloitettiin tekemällä ohjelmistotestausta vanhalla kokoonpanolla. Tutustumisen aikana pääsi hyvin tutustumaan simulaattorin toimintoihin, miten testaaminen pitäisi edetä ja mitä testeiltä vaaditaan. Vanhalla kokoonpanolla testaaminen tapahtuu tarkasti tietyssä järjestyksessä täysin manuaalisesti eikä testaamisessa ole mitään mikä toimisi automaattisesti. Testaaminen manuaalisesti aiheuttaa simulointiin ja täten mittauksiin enemmän virheitä ja ongelmia, koska toistoja on tehtävä peräkkäin kymmeniä kertoja. Edellä mainitut asiat ovat tärkein syy, miksi testaaminen halutaan automatisoida mahdollisimman pitkälle.

4.2 Simulaattorin laitteiston määrittely

4.2.1 Olemassa olevien toimintojen määrittely

Simulaattorin määrittely aloitettiin kirjaamalla ylös vanhassa kokoonpanossa olevien vipukytkimien määrä ja niiden funktio, jotta tarvittavan logiikan laajuus voidaan määrittää. Samalla piirrettiin myös simulaattorista toimintakaavio. Toimintakaaviosta käy ilmi laitteiden rajapinnat ja se helpottaa hahmottamaan koko järjestelmän kokonaiskuvaa, joka käy ilmi kuviosta 2. Laajuudella tarkoitetaan Beckhoffin logiikan mallia ja siihen liitettävien I/O moduulien määrää ja tyyppiä. Kytkimiä käytettiin vanhassa kokoonpanossa katkomaan hissin ohjausjärjestelmän ja laitteiden 24VDC sähkönsyöttöä, ja turvapiiriketjuja sekä kuittaamaan turvatoimintoja.



Kuvio 3. Simulaattorin toimintakaavio

4.2.2 Uusien toimintojen määrittely

Kun olemassa olevien kytkimien määrittely oli tehty oli aika määritellä mitä uutta tulevaa kokoonpanoa varten tarvitaan, jotta logiikkaohjelma saa kaiken tarvittavan tiedon testauksia automatisoidessa. Kävi ilmi, että kokoonpanoon täytyy lisätä hissin omia lisäkortteja, joista saadaan logiikalle releteitoa hissin sijainnista ja tilasta. Esimerkkinä hissin sijainti on ensimmäisessä kerroksessa, hissin tila on pois käytöstä. Toinen testausta helpottava ja nopeuttava asia on instrumentoida hissin HMI-käyttöpaneelin napisto ja liittää se logiikkaan. Käyttöpaneelissa on esimerkiksi napisto hissin huoltoajoa varten sekä resetointinappi turvatoimintojen resetoimiseen. Hissin oma ohjausjärjestelmä hoitaa siis kaiken hissiin liittyvän esimerkiksi taajuusmuuttajalle menevän tiedon, mutta ulkoisella logiikalla annetaan tarvittavat hissikutsut hissille, liikutetaan lineaarimoottoria, ohjataan esimerkiksi hissin turvapiirin osia päälle tai pois, jotka aikaisemmin tehtiin vipukytkimiä käyttämällä ja luetaan hissin tilatietoja. Ulkoista logiikkaa ei toisin sanoen liitetä hissin ohjausjärjestelmän tietoliikenneväylään vaan on täysin erillinen laitteisto hissin ohjausjärjestelmän rinnalla.

Simulaattorissa ei ole fyysisesti asennettu hissin tason- ja korinovia. Fyysiset ovet ohjauksineen vievät paljon tilaa eikä simulaattorissa päätarkoitus ole testata ovien toimintaa. Tämän takia ovien toiminta simuloidaan erillisellä ovisimulaattorilla. Ovisimulaattori on erillisellä I/O-kortilla, johon on asennettu erillinen ohjelmisto ja mikä osaa simuloida ovien toimintaa. Käytännössä tämä toimii siten, että

simulaattorikortille tuodaan hissin ohjausjärjestelmästä sähkönsyöttö sekä esimerkiksi ovi auki käsky, jota simulaattori noudattaa antaen tilatietoa hissin ohjaukselle. Tilatiedot ovisimulaattori antaa relelähtöjen kautta ja kyseiset tilatiedot ovat ovi kiinni, ovi sulkeutumassa, ovi auki ja ovi aukeamassa. Ovisimulaattori toimii pienen logiikkaohjelman tavoin.

5 Valitut komponentit

Työssä päätettiin valita logiikan toimittajaksi Beckhoff hyvien kokemusten ja hyvän teknisen tuen takia. Beckhoffin käyttöä puoltaa myös mahdollisuus ohjelmoida ja testata ohjelmistoa ilman pakollisia lisenssejä, joka madaltaa kynnystä opetella kyseisen laitteiston käyttöä ja ohjelmointia. Aloittelijoille tämä mahdollistaa opettelun ilman, että joutuu ostamaan laitteistoa tai lisenssejä. Tällöin TwinCAT-ohjelmiston kautta täytyy generoida ilmainen lisenssiavain 7 vuorokauden ajaksi. 7 vuorokauden jälkeen ohjelmisto pyytää uudelleen generoimaan lisenssin. Jos logiikan on tarkoitus olla päällä ympäri vuorokauden, voi yritykseltä ostaa kiinteän lisenssin, jota ei tarvitse generoida uudelleen.

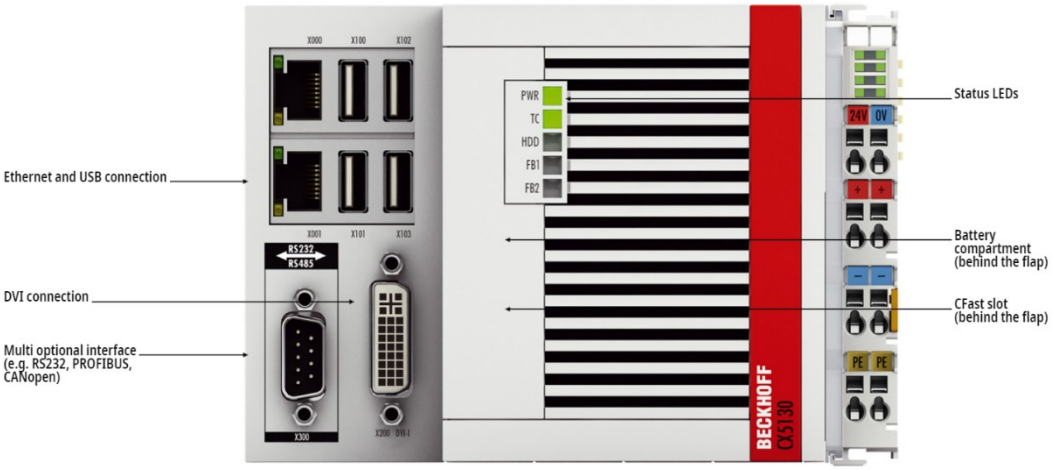
Laitteiston määrittelystä selvisi, että tulevan logiikan kokoonpanoon tarvitaan yhteensä 23 digitaalista sisääntuloa ja 47 digitaalista lähtöä. Digitaalisiin lähtöihin on laskettu myös niin sanottu relelähtö. Relelähtö tarkoittaa terminaalissa olevaa tilaa vaihtavaa vaihtokosketinta, kun taas tavallinen digitaalilähtö antaa tiettyä jännitetasoa. Simulaattorin eri jännitetasojen takia joudutaan käyttämään eri I/O-terminaalityyppejä. Taulukossa 1 on taulukoitu tarvittavat terminaalit, niiden tyyppi, terminaalien vaadittu määrä ja terminaaleissa olevien I/O yhteenlaskettu kokonaismäärä.

Taulukko 1. Tarvittavien IO-terminaalien määrät ja tyyppi havainnollistettu.

Nimi	Tyyppi	Määrä	I/O Tyyppi	Lähtöjen/tulojen kokonais määrä
Terminaali	EL1809	2	Digitaalitulo	32
Terminaali	EL2008	1	Digitaalilähtö	8
Terminaali	EL2652	8	Relelähtö	16
Terminaali	EL2624	8	Relelähtö	32
Terminaali	EL1722	1	Digitaalitulo	2
Terminaali	EK1110	1	Väyläliitin	
Terminaali	EK1110	2	Väyläliittimen pääty	
PLC	CX5130	1	Ohjainyksikkö	

5.1 CX5130

Logiikaksi valittiin vähävirtainen ja ilman tuuletinta oleva Beckhoff CX5130-tyypin sulautettu PC. Sulautettu PC nimestään huolimatta laite on täysiverinen logiikka, jossa taustalla pyörii muokattu Windows 10-käyttöjärjestelmä. Logiikassa on kaksiytiminen Intel Atom E3287-prosessori, jonka kellotaajuus on 1.75GHz. Keskusmuistia logiikassa on 4GB DDR3 eikä se ole laajennettavissa. Tallennustilana toimii erikseen hankittava CFast- ja microSD-kortit. Liittiminä löytyy neljä USB 2.0-porttia, kaksi RJ45-porttia ja yksi DVI-I-näyttöliitin. RJ45-portteja käytetään EtherCAT-väylää varten. Kuviossa 3 näkyy liitännät ja niiden sijainnit. Valitussa logiikassa ei RS232 liitäntää kuvasta huolimatta logiikassa ole olemassa. Logiikassa ei ole itsessään kiinni I/O-portteja vaan nämä tulee hankkia erikseen lukuisista eri vaihtoehdoista. Laitteen ulkonäkö ja liitännät käy ilmi kuviosta 4. (Beckhoff CX5130 2021.)



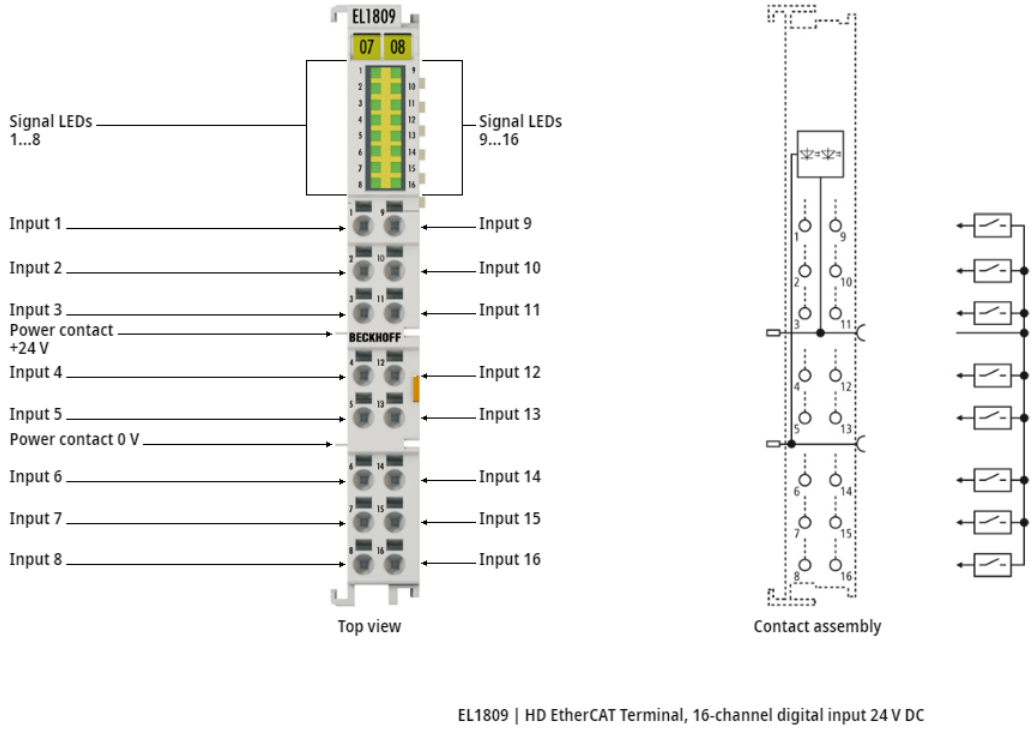
Kuvio 4. CX5130 ja sen liitännät. (Beckhoff CX5130 2021).

5.2 I/O-terminaalit

Logiikkaan liitetään halutut I/O-terminaalit. Suunnitteluvaiheessa määriteltiin tarvittavien terminaalien tyyppi ja määrä. Analogisille tuloille ja lähdöille ei automatisoinnissa ollut tarvetta ja logiikka on täten toteutettu pelkästään digitaalisilla tuloilla ja lähdöillä.

5.2.1 EL1809

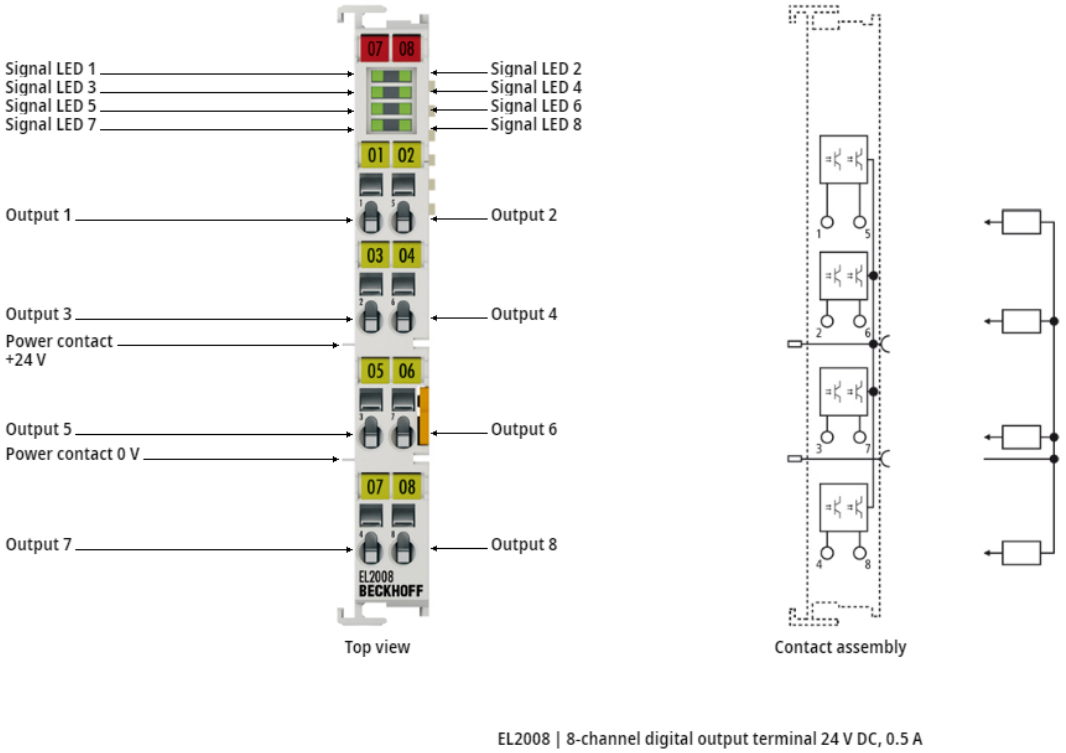
EL1809 on 16-kanavainen DI-terminaali. Sisääntulot toimivat 24V DC jännitetasolla. Liitäntä toimii 1-johdin periaatteella, jolloin signaali tuodaan hissin ohjausjärjestelmästä vain yhdellä johtimella sisääntuloon. Signaali on päällä jännitteen ollessa 11-30 V ja pois päältä jännitteen ollessa alle 5 V. Terminaalin yläosassa on jokaiselle kanavalle oma kanavan tilaa indikoiva led-valo. Kytkentäkuva käy ilmi kuviosta 5. (Beckhoff EL1809 2021.)



Kuvio 5. EL1809 kytkentäkuva. (Beckhoff EL1809 2021).

5.2.2 EL2008

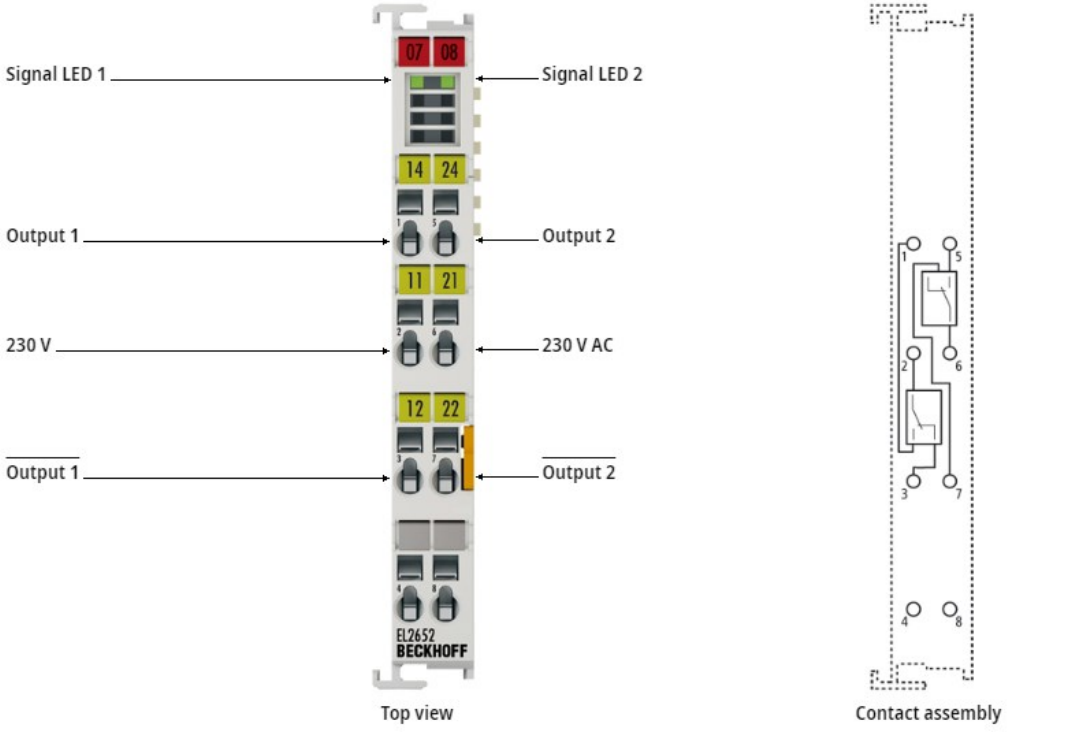
EL2008 on 8-kanavainen DO-terminaali. Terminaali lähettää 24V DC ohjaussignaalia hissin ohjausjärjestelmään. Lähdöillä voidaan ohjata resistiivistä tai induktiivista kuormaa. Kanavan maksimi virranantokyky on 500mA. Signaali kytketään 1-johtimisena kuormaan. Terminaalin yläosassa on jokaiselle kanavalle oma kanavan tilaa indikoiva led-valo. Kytkentäkuva käy ilmi kuviosta 6. (Beckhoff EL2008 2021.)



Kuvio 6. EL2008 kytkentäkuva. (Beckhoff EL2008 2021).

5.2.3 EL2652

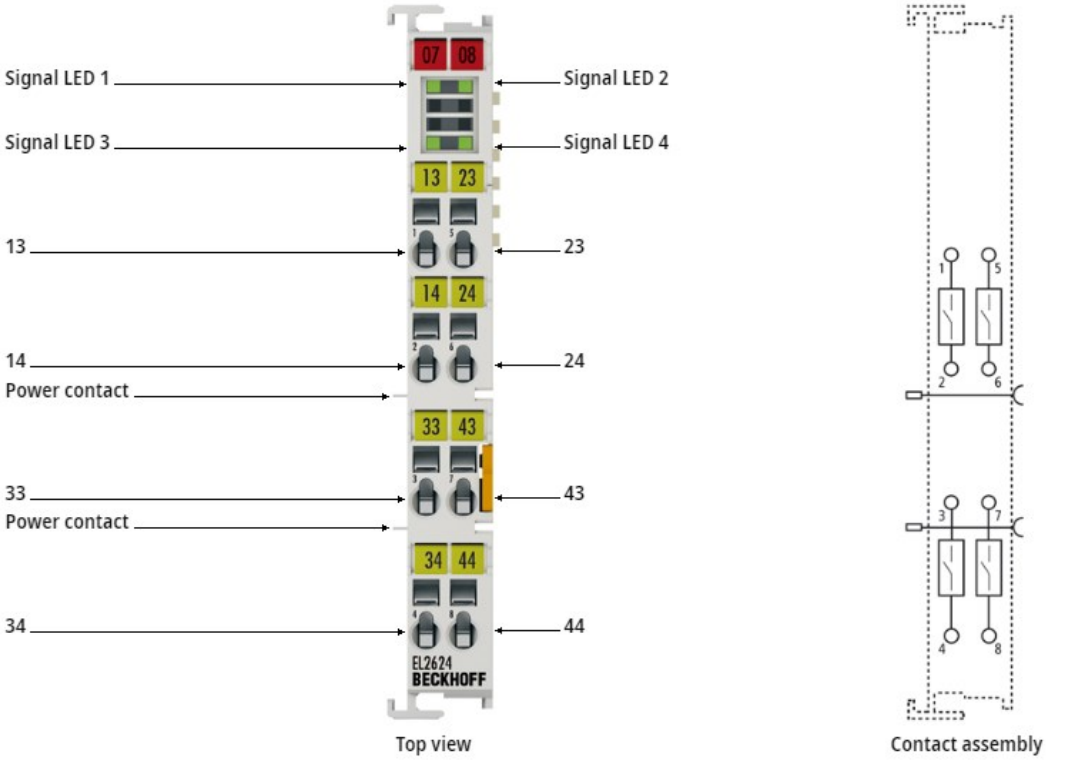
EL2652 on 2-kanavainen relelähtö potentiaalivapailla vaihtokoskettimilla. Kosketti-mien jännitekestoisuus on 230V AC tai 300V DC. Maksimivirta vaihtelee jännitetason- ja tyypin mukaan. Esimerkkinä suurin sallittu virta on 1A 230V AC ja 0,15A 300V DC. Terminaalin yläosassa on jokaiselle kanavalle oma kanavan tilaa indikoiva led-valo. Kytkentäkuva käy ilmi kuviosta 7. (Beckhoff EL2652 2021.)



Kuvio 7. EL2652 kytkentäkuva. (Beckhoff EL2652 2021).

5.2.4 EL2624

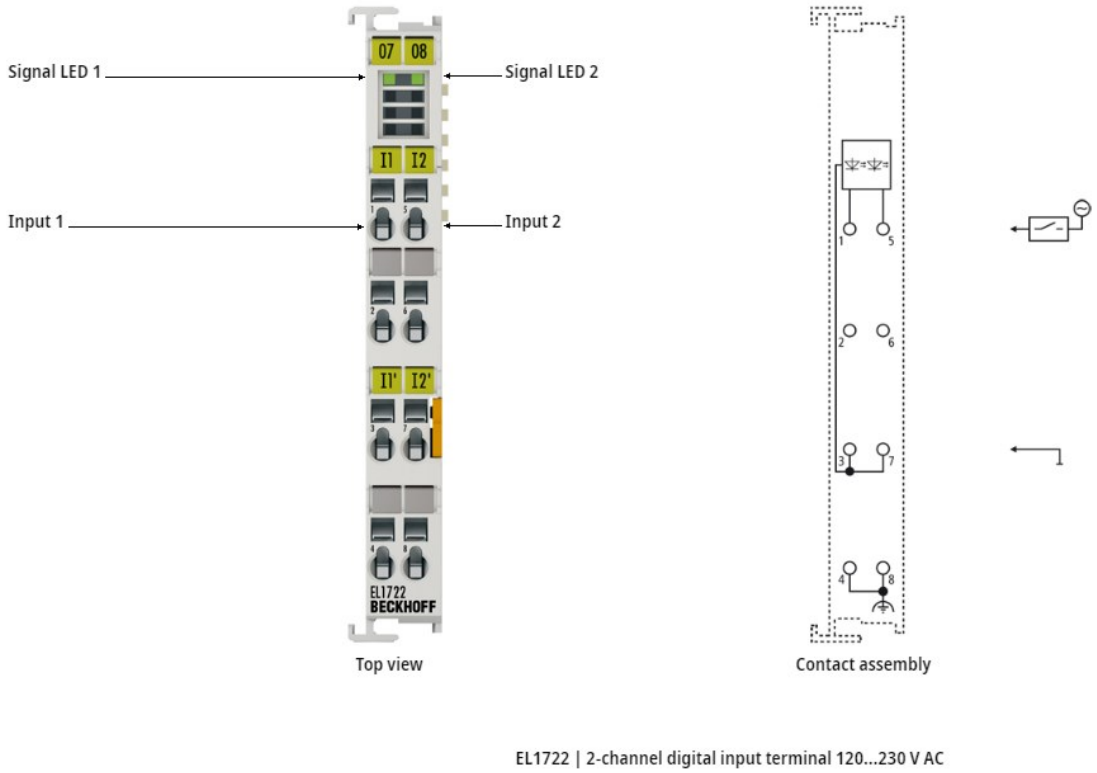
EL2624 on 4-kanavainen potentiaalivapaa relelähtö. Koskettimien jännitekestoisuus on 125V AC ja 30V DC ja suurin sallittu virta kanavalla on 0.5A AC ja 2A DC. Terminaalin yläosassa on jokaiselle kanavalle oma kanavan tilaa indikoiva led-valo. Kytkentäkuva käy ilmi kuviosta 8. (Beckhoff EL2624 2021.)



Kuvio 8. EL2624 kytkentäkuva. (Beckhoff EL2624 2021).

5.2.5 EL1722

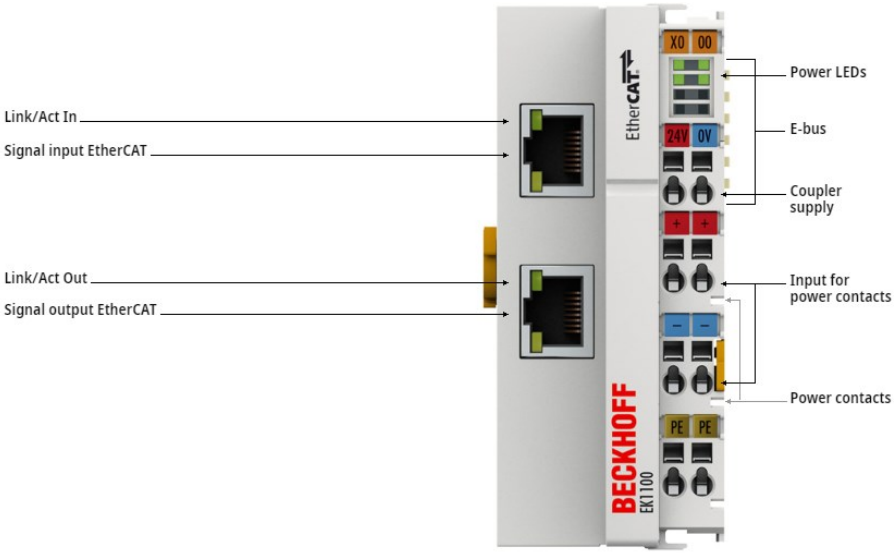
EL1722 on 2-kanavainen DI-terminaali. Sisääntulot toimivat 120-230V AC jännitteellä. Tulo pysyy pois päältä 0-40V jännitteellä ja menee aktiiviseksi 79-260V jännitealueella. Terminaalin yläosassa on jokaiselle kanavalle oma kanavan tilaa indikoiva ledvalo. Kytkentäkuva käy ilmi kuviosta 9. (Beckhoff EL1722 2021.)



Kuvio 9. EL1722 kytkentäkuva. (Beckhoff EL1722 2021).

5.2.6 EK1100

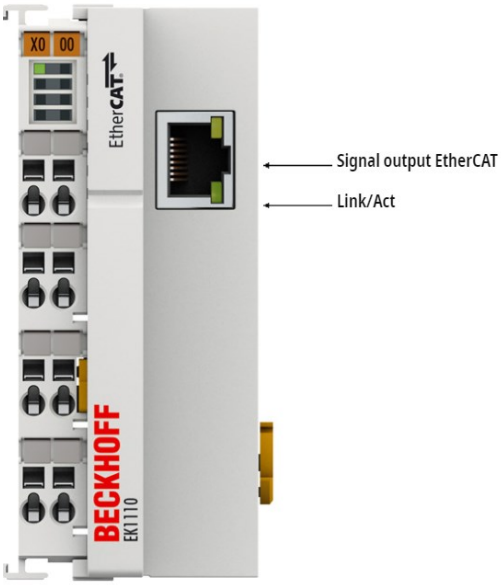
EK1100 EtherCAT Coupler on väyläliitin, jolla voidaan muualle hajautetut terminaalit yhdistää EtherCAT väylän kautta logiikalle. EK1100 perään liitetään tarvittavat IO-terminaalit. Kytkimessä on kaksi RJ45 liitäntää väylää varten. Väylän lisäksi kytkimeen on tuotava 24V DC sähkönsyöttö terminaaleille. Kytkentäkuva käy ilmi kuviosta 10. (Beckhoff EK1100 2021.)



Kuvio 10. EK1100 liitännät ja ulkonäkö. (Beckhoff EK1100 2021).

5.2.7 EK1110

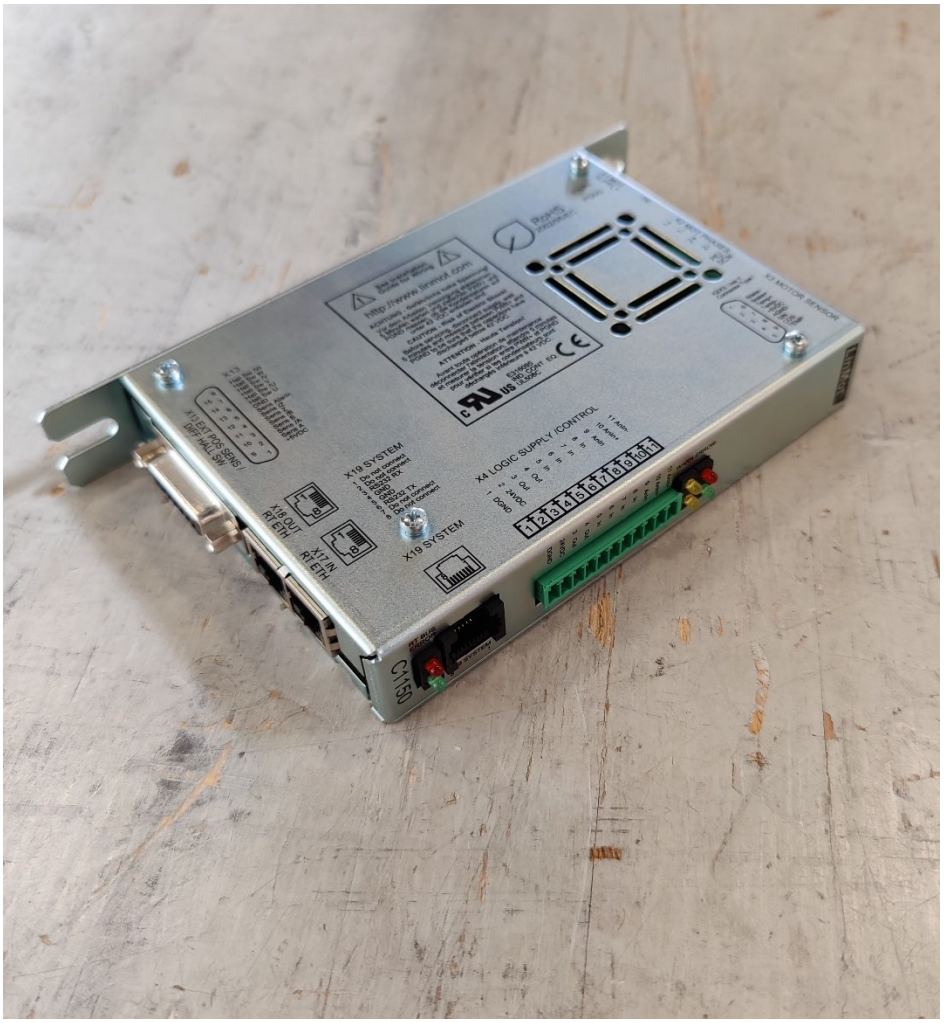
EK1110 on terminaaliblokkien perään liitettävä väyläliitin. Kuviosta 11 käy ilmi, että liittimenä löytyy yksi RJ45, josta voidaan jatkaa EtherCAT-väylää. (Beckhoff EK1110 2021).



Kuvio 11. EK1110 liitännät ja ulkonäkö. (Beckhoff EK1110 2021).

5.3 LinMot C1150 servo drive

Vanhassa kokoonpanossa lineaarimoottoria ohjasi B1100 servo-ohjain, mihin tarvittavat käskyt annettiin vipukytkimillä Digital Input-muodossa. Tämä asetti rajoituksia tarvittaville komennoilla ja vaati testien välissä parametrien muutoksia kuten nopeuden hidastamista. Vanhassa ohjaimessa ei ollut liitäntää EtherCAT-väylälle, joten uudeksi ohjaimeksi valittiin kuviossa 12 oleva LinMot C1150 servo-ohjain. Kyseisessä ohjaimessa on EtherCAT-väyläliitäntä, mikä tarkoittaa sitä, että moottoria voidaan ohjata kyseisen väylän kautta. Ohjaimen käskyttäminen väylän kautta vähentää tarvittavaa johdinmäärää ja tarjoaa enemmän tallennettavia ohjausparametrejä. (LinMot servo drives 2021.)



Kuvio 12. LinMot servo C1150

6 Ohjelmointi

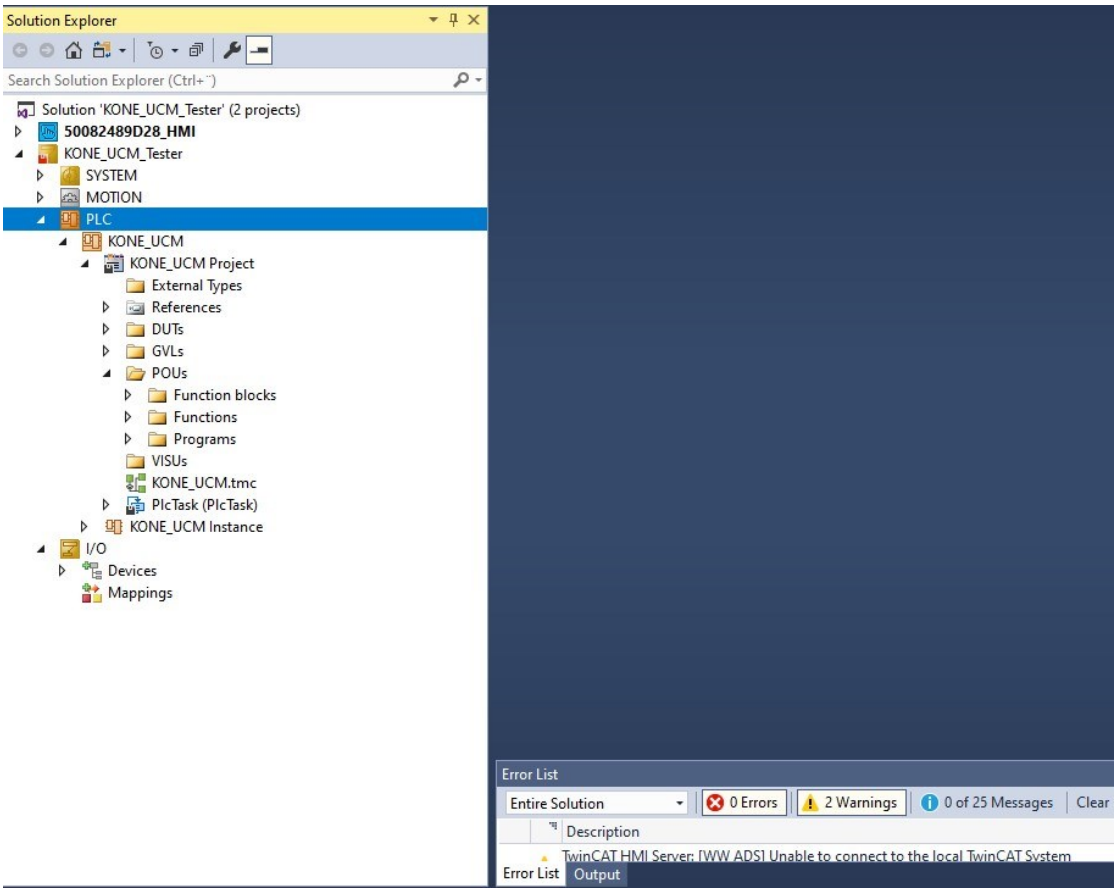
6.1 TwinCAT-ohjelmointi

Ohjelmointia aloitettaessa tärkein asia on päättää millä kielellä ohjelmointi tehdään. TwinCAT ympäristössä ohjelmointi on mahdollista standardin IEC61131-3 mukaisilla ohjelmointikielillä. Standardin mukaiset kielet ovat FBD (Function Block Diagram), IL (Instruction list), LD (Ladder Logic Diagram), SFC (Sequential Function Chart), ST (Structured Text). Edellä listattuja kieliä löytyy myös muiden valmistajien ohjelmistoista, mutta näissä saattaa olla pieniä valmistajakohtaisia eroja muun muassa nimeämisissä. (Beckhoff Programming languages n.d.)

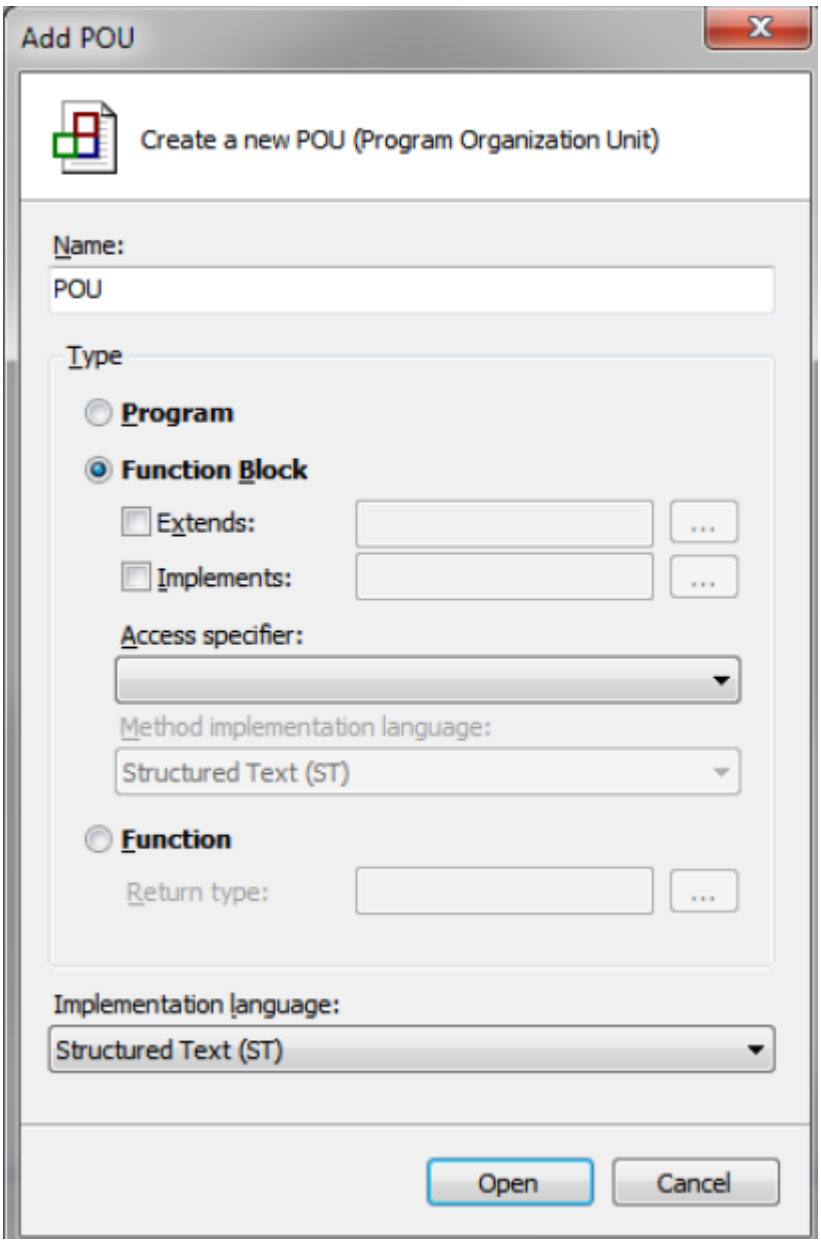
Ohjelmointikieleksi valitsin ensin FBD-kielen aikaisempien kokemusten perusteella ja graafisen ympäristön helppoudesta, mutta suosituksesta ja toimeksiantajan toivomuksesta muutin ohjelmointikielen ST-kieleksi. ST-kielellä on helpompi tehdä monimutkaisempia kokonaisuuksia ja kokemuksen myötä ohjelmakoodin lukeminen ja ohjelman etenemisen seuraaminen pitäisi olla myös hieman helpompaa. Yleensä suurempien yritysten sisällä on valmiiksi ohjeistus, miten ohjelmointi pitää suorittaa. Tämän tarkoituksena on helpottaa monimutkaisten ohjelmistojen ymmärtämistä ja tekemistä varsinkin, kun projekteissa saattaa olla mukana jopa kymmeniä ihmisiä.

Ohjelmointi ja sen rakentaminen keskittyy ”Solution Explorer”-ikkunaan ja sen sisältämiin kansioihin. Kansioiden sisälle rakennetaan kaikki aina logiikan määrittelystä ohjelman tekemiseen. Toimeksiannon osalta oleellisimmat kohdat ”Solution Explorer”-ikkunassa on ”System”-, ”Motion”-, ”IO”- ja ”PLC”-kansiot, jotka käyvät ilmi kuviosta 13. ”Motion”-kansiossa määritellään liikkeenohjaustoimintoihin liittyvät asiat. Työn osalta sinne määritellään lineaarimoottorin parametrit. ”IO”-kansiossa määritellään tietoliikennerajapinnat, kenttäväylän konfiguraatio ja väylästä löytyvien laitteiden yhdistäminen globaaleihin muuttujiin. ”PLC”-kansiossa on taas kaikki ohjauslogiikkaan liittyvät alikansiot. Nämä kansiot ovat ”DUTs”, ”POUs” ja ”GVLs”. ”DUTs”-kansiossa on käyttäjän määrittelemät datatyypit. ”GVLs”-kansiossa on globaalit muuttujalistat. ”POUs”-kansio sisältää ohjelmalogiikkaan liittyvät ohjelmat. Ohjelmat

tehdään joko ohjelmina (Programs), funktioina (Functions) tai funktiolohkoina (Function blocks) tarpeen mukaan. Ohjelmaa käytetään pääohjelman ja suurien aliohjelmien rakentamiseen, joissa ei ole tarvetta monistettavuudelle. Ohjelman sisällä myös arvot säilyvät. Funktiolohkoja käytetään monistettavien ohjelmalohkojen tekoon, joissa on tarvetta muuttujien arvojen muistamiselle. Funktio on muuten samanlainen kuin funktiolohko, mutta funktion sisäiset muuttujat eivät pysy muistissa. Ohjelman tyyppi valitaan aina erikseen (Kuvio 14).



Kuvio 13. Solution Explorer



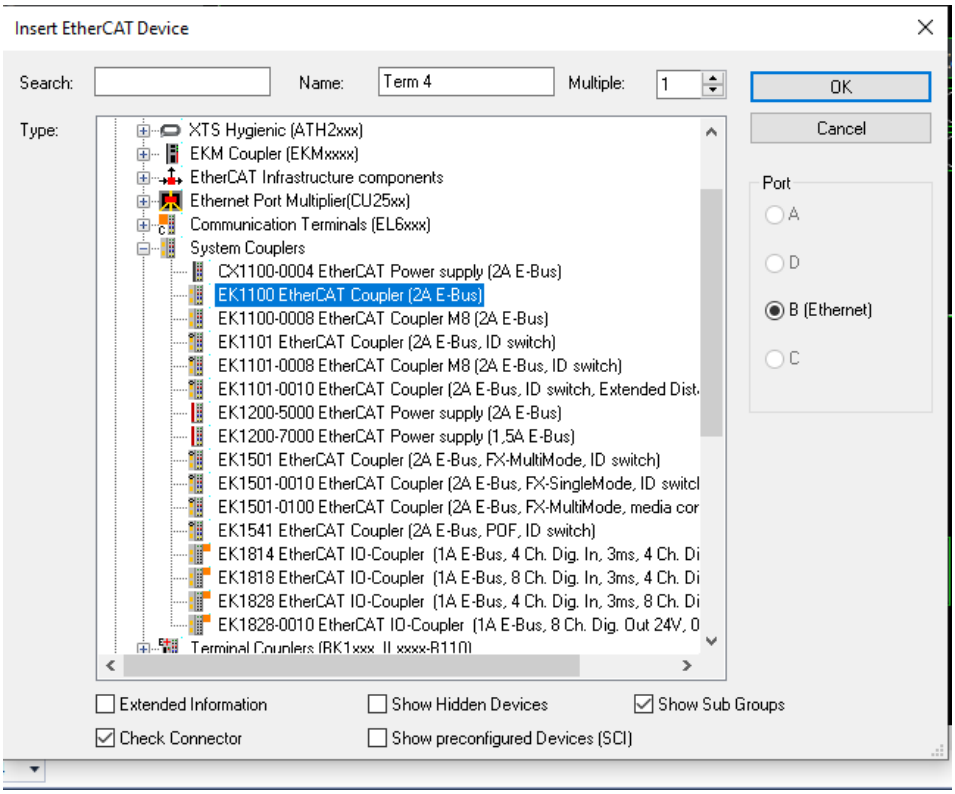
Kuvio 14. Ohjelmalogiikan tyypin valitseminen

6.2 Projektin avaaminen

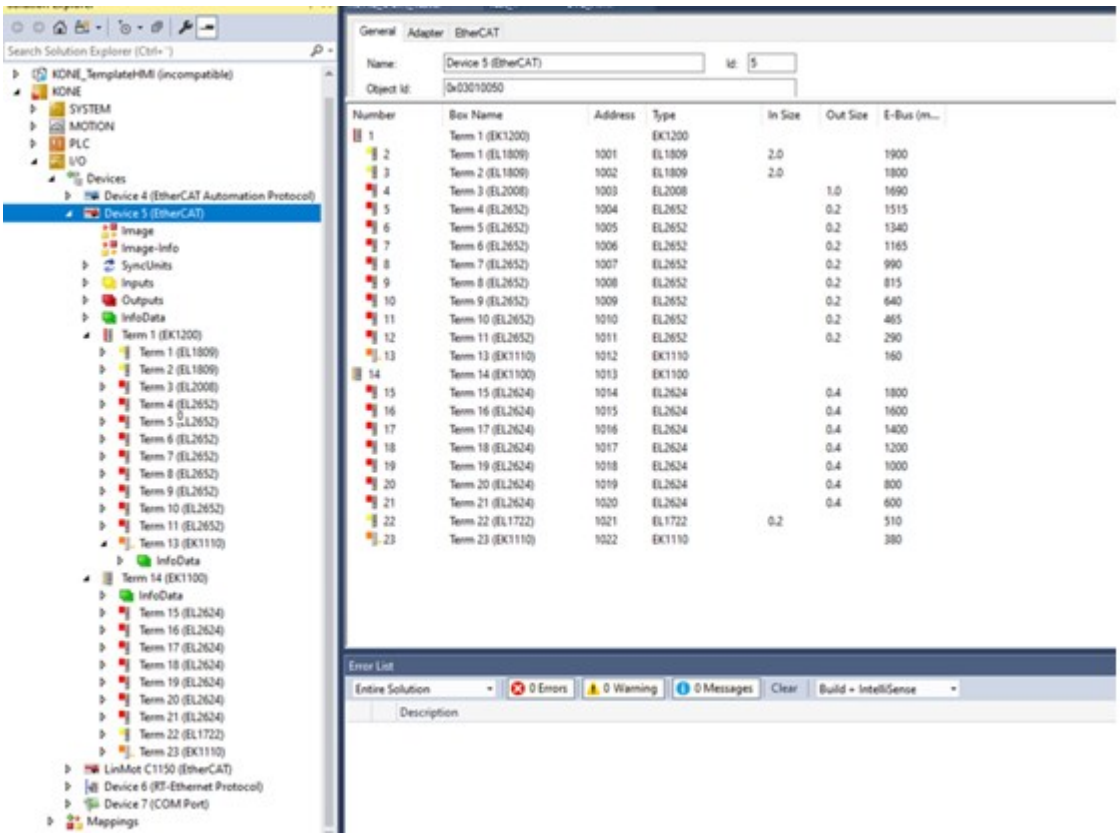
Projekti aloitetaan toimeksiantajan valmiille TwinCAT-projektipohjalle, jossa on valmiina tiettyjä simulaattoreihin liittyviä toimintoja ja muuttujia. Näihin toimintoihin ei syvennytä eivätkä ne vaikuta tämän projektin kulkuun mitenkään. Valmiin projektipohjan ja sille tehdyn ohjeistuksen tarkoituksena on standardisoida kaikki toimeksiantajan simulaattorit samanlaisiksi. Tämä helpottaa simulaattoreiden yhteensopivuutta sekä nopeuttaa simulaattoreiden koodin lukemista ja muokkaamista, koska kaikissa simulaattoreissa logiikan ohjeistus on sama.

6.3 IO-terminaalien ja servon lisääminen projektiin

Lineaarimoottorin servo-ohjain ja terminaaliblokit voidaan lisätä projektin EtherCAT-väylään kahdella tapaa. Helpoin tapa, jos laitteisto on jo olemassa, on käyttää TwinCAT-ohjelman väylän skannausta. Tämän avulla kaikki väylässä olevat laitteet löytyvät. Skannauksen jälkeen ohjelma lisää löydetyt laitteet projektiin omaan kansioon, josta laitteiden linkitykset GVL-muuttujiin voidaan tehdä. Fyysiset laitteet voidaan myös lisätä samaan kansioon ilman skannausta, jos on tiedossa fyysisten laitteiden määrä ja tyyppi, kuten kuviossa 15. Ilman skannausta tehtävä lisääminen helpottaa varsinkin suunnitteluvaiheessa, koska TwinCAT-ohjelma osaa kertoa IO-terminaaliblokkien virran kulutuksen ja jäljellä olevan virran syötön. Tämä käy ilmi kuviosta 16 E-Bus- sarakkeen kohdalta. Ohjelma huomauttaa, jos virransyöttö ei riitä laitteiden paljouden takia. Tällöin voidaan varautua tarvittaviin laitteistohankintoihin ja joko lisätä IO-terminaaliblokkien väliin virransyöttöä varten tarkoitettu lisäkortti tai ylimääräiset IO-terminaaliblokit voidaan hajauttaa fyysisesti muualle rakennettuun järjestelmään käyttäen EtherCAT-väylää ja toista ulkoista virransyöttöä.



Kuvio 15. Laitteiden lisääminen väylään ilman skannausta.



Kuvio 16. Projektiin liitettyjen laitteiden näkymä

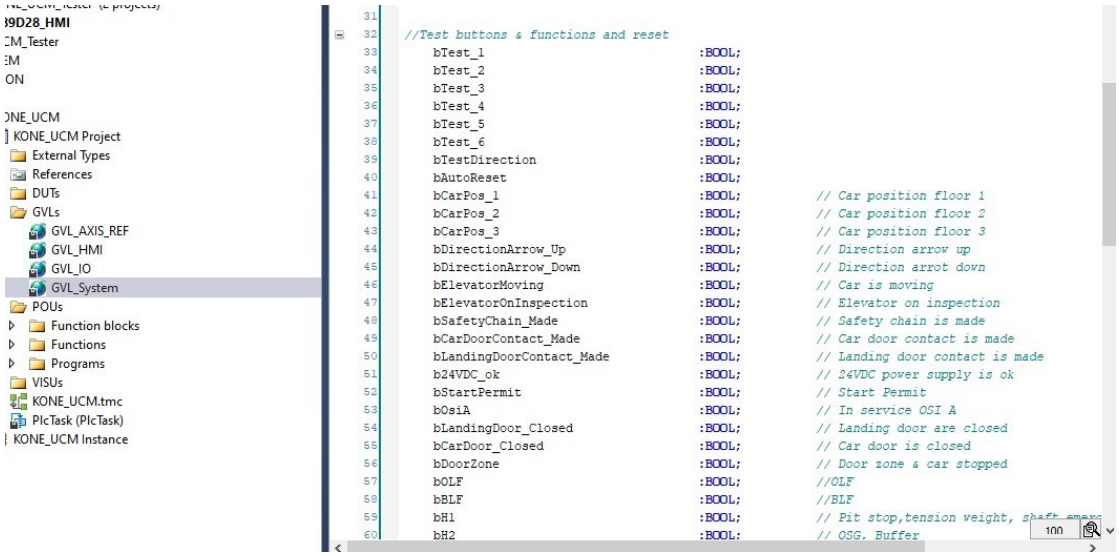
6.4 Muuttujien määrittely

Projekti aloitettiin muuttujien määrittelyillä. Projektin muuttujat määritellään GVL-listoille,joita voidaan lukea ja kirjoittaa. Muuttujia määritettäessä kirjoitetaan muuttujan nimi sekä muuttujan tyyppi. Muuttujan tyyppi on tässä työssä joko Bool-tyyppinen eli ”true” tai ”false” tai jokin numeraalinen lukuarvo.

Ohjelmassa on kolme eri ensisijaista GVL-listaa sisältävää tiedostoa. Ensimmäinen GVL-lista on ”GVL_IO”, joka on tarkoitettu fyysisten terminaalien ohjaukseen ja lukemiseen. Toinen on ”GVL_HMI”, joka on tarkoitettu HMI-järjestelmän tilojen lukemiseen ja kirjoittamiseen. HMI:n tarkoitus on ohjata tai lukea tehtyä ohjelmaa tai IO-tiloja rakennetun graafisen käyttöliittymän kautta hiiren tai kosketusnäytön avustuksella. Viimeisenä muuttujalistana on ”GVL_System”, johon kaikki edellisen muuttujalistojen muuttujat myös tallennetaan. Tämän listan muuttujia luetaan tai kirjoitetaan

muualta ohjelmistosta ja muutokset kirjoitetaan tai luetaan ”GVL_HMI” tai ”GVL_IO” listoille. Kuviosta 17 käy ilmi osa ”GVL_System” muuttujalistan muuttujista.

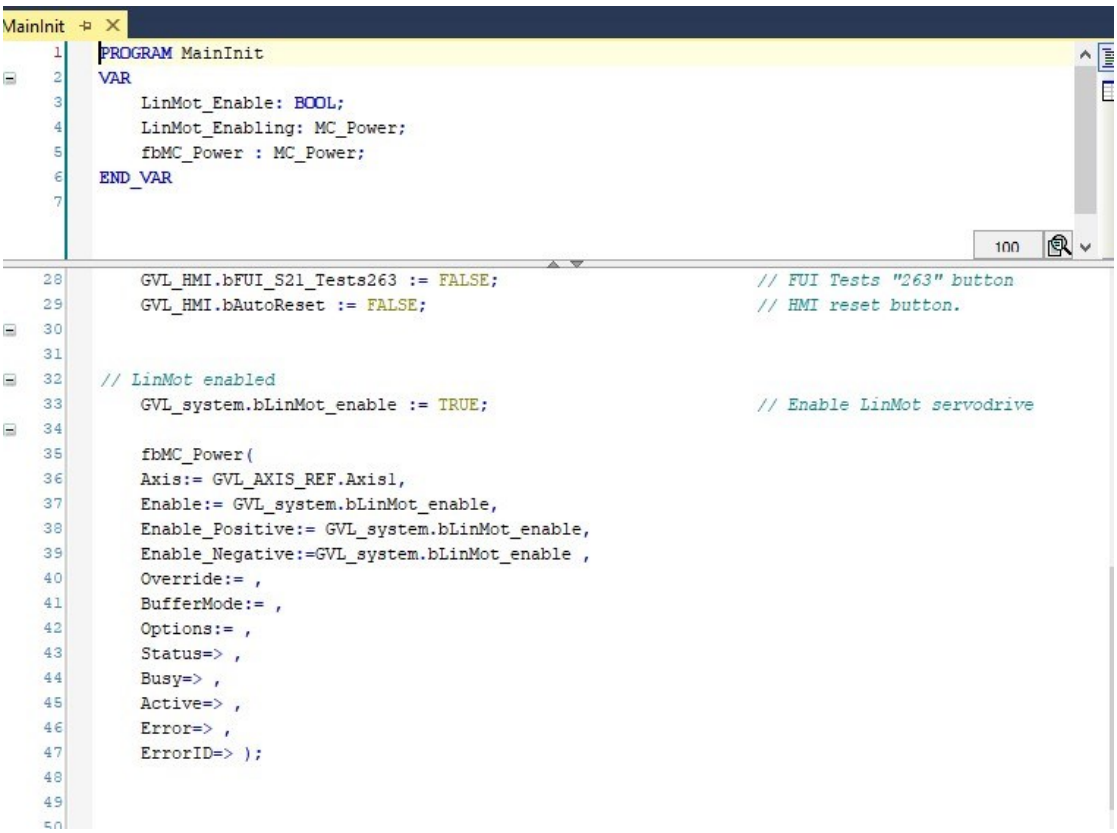
Kaikki projektissa käytettävät muuttujat ovat ”Bool”-tyyppisiä eli arvot ovat joko ”true” tai ”false” eli toisin sanoen joko 1 tai 0.



Kuvio 17. GVL_System muuttujalista ilman alkuarvoa.

6.5 Muuttujien alustaminen käynnistyksen yhteydessä

Kun simulaattori käynnistetään, täytyy tietyt muuttujat alustaa tiettyyn tilaan heti käynnistyksen yhteydessä. Alustaminen tapahtuu ”MainInit”-ohjelmalohkon sisällä, jossa halutuille muuttujille kerrotaan haluttu arvo käynnistyksen yhteydessä. Tässä tapauksessa arvo on joko ”true” tai ”false”. Esimerkiksi kuviossa 18 lineaarimoottorin ohjaukseen on luotu muuttuja nimeltään ”LinMot_Enable”, jonka tarkoituksena on sallia lineaarimoottorin toiminta. Jos muuttujan arvo on ”false”, ei lineaarimoottori suorita haluttuja liikeratoja ohjelmia suorittaessa.

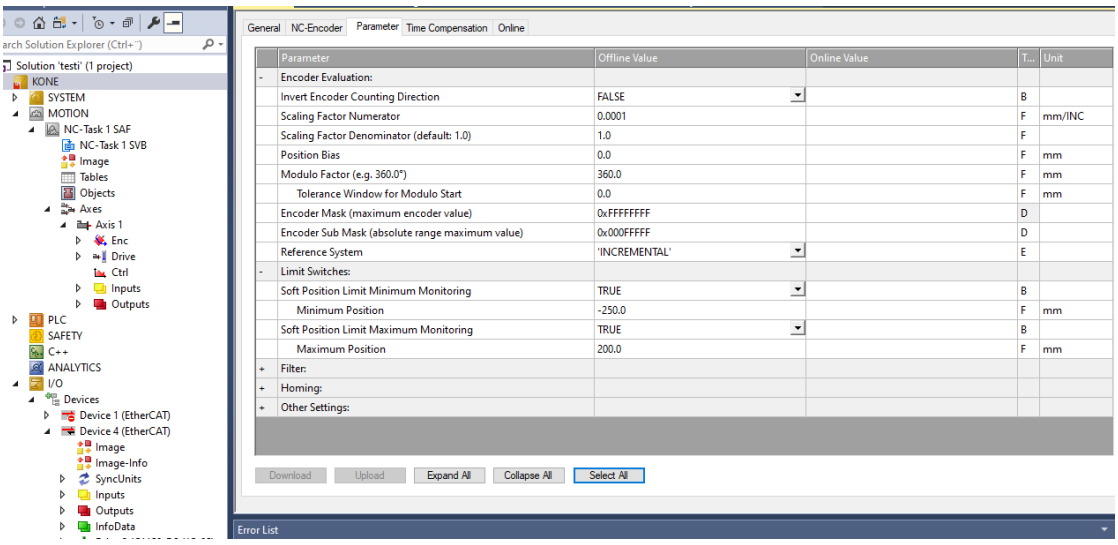


Kuvio 18. Lineaarimoottorin toiminnan salliminen. bLinMot_enable muuttuja alustetaan päälle logiikan käynnistyksen yhteydessä.

6.6 Lineaarimoottorin servo-ohjaimen käyttöönotto.

Kun lineaarimoottorin ohjain on lisätty väylään, täytyy ”Motion” kansioon lisätä uusi ”Task” ja tämän alle ohjattava akseli esimerkiksi nimellä ”Axis1”. Tämän avulla parametroidaan lineaarimoottorin tarvittavat parametrit, kuten liikeradan pituus ja sen päätyrajat, kiihtyvyydet ja muut olennaiset arvot mitä näkyy kuviossa 19. ”Axis1” valikoista voidaan tarvittaessa ajaa lineaarimoottoria manuaalisesti käsin väylää pitkin ja näin ollen testata moottorin toimintoja ja parametreihin aseteltuja arvoja. Esimerkiksi päätyrajojen testaus on tärkeä toimenpide, jotta ohjelma osaa pysäyttää moottorin akselin liikkeen ennen päätyä. Näin ollen moottoria ei saa ajettua kovalla vauhdilla päätyyn virheellisen käskyn takia.

Lineaarimoottorin ohjaukseen täytyy myös luoda ”Axis1” varten oma GVL-muuttujalista. Tämän avulla TwinCAT osaa yhdistää Motion-kansiossa olevan ”Axis1” liikkeenohjauksen ohjelmakoodista löytyvään kutsun liikuttaa lineaarimoottoria.

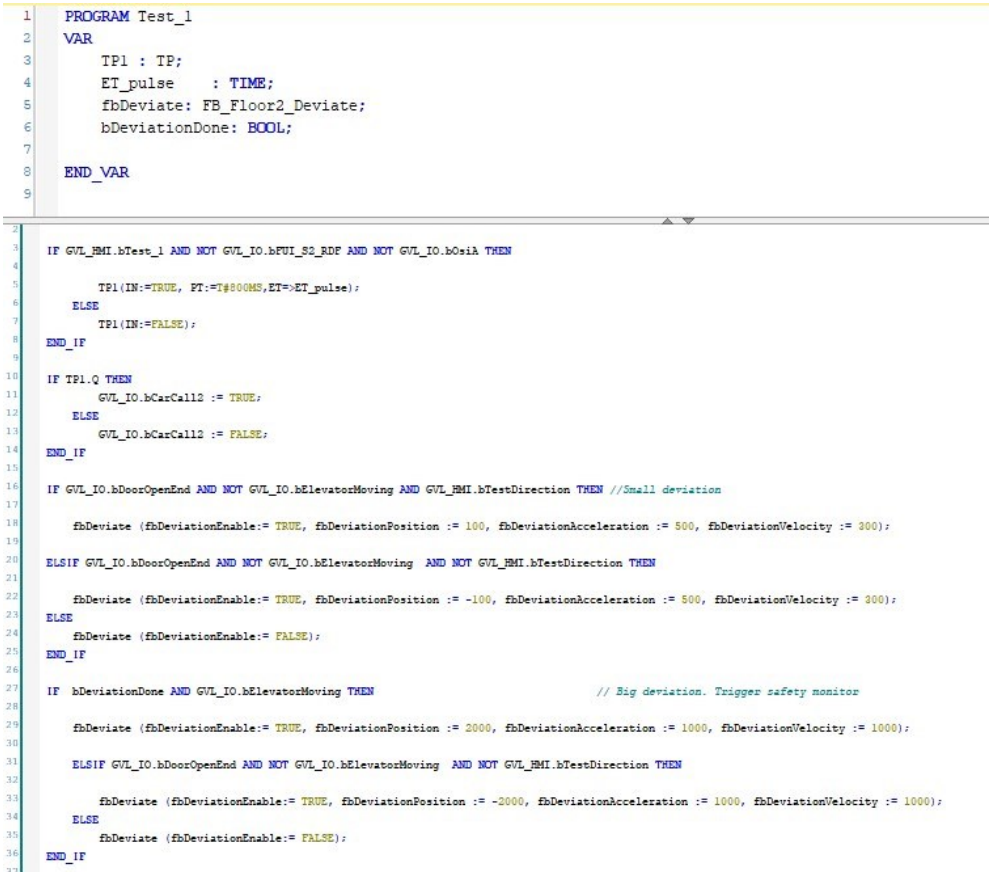


Kuvio 19. Servo-ohjaimen parametrointi.

6.7 Testiohjelman kirjoittaminen

Testiohjelmien tekemisessä keskityttiin yhden testiohjelman kirjoittamiseen, mikä käy ilmi kuviosta 20. Testiohjelmaa pystytään kopioimaan tarvittava määrä ja muokkaamaan tarvittavia arvoja, kuten lineaarimoottorin liikkeen nopeutta ja etäisyyttä. Kaikkien testien toimintaperiaate on sama, joten tästä syystä testiohjelman kopiointi nopeuttaa ohjelman kirjoittamista huomattavasti.

Yleisellä tasolla testi alkaa siitä, että logiikka antaa hissille kutsun 1-kerrokseen. Kun hissi on saapunut ensimmäiseen kerrokseen tai on jo valmiina ollut siellä niin ohjelma antaa hissille kutsun 2-kerrokseen. Kun hissi saapuu 2-kerrokseen ja ovet ovat auki tai avautuvat, saa logiikka tästä tiedon. Tämän jälkeen antaa logiikka lineaarimoottorille käskyn liikuttaa kerrosmagneettia hieman. Tämän jälkeen hissi yrittää hitaasti tasata hissikorin suhteessa magneettiin ja tästä liikkeestä logiikka saa tiedon hissin ohjausjärjestelmältä. Kun hissi yrittää tasata hissikoria seuraten magneettia, liikutetaan magneetti tietyllä nopeudella pois kerrosalueelta. Tällöin hissin turvaohjelmisto reagoi tapahtumaan ja tekee tarvittavat toimenpiteet. Testien onnistuminen ja hissin ohjausjärjestelmän haluttu lopputulos varmistetaan lukemalla hissin ohjausjärjestelmän antamien vikakoodien avulla sekä ulkoisella mittalaitteella, joka mittaa tarvittavien kohteiden reaktioaikoja.

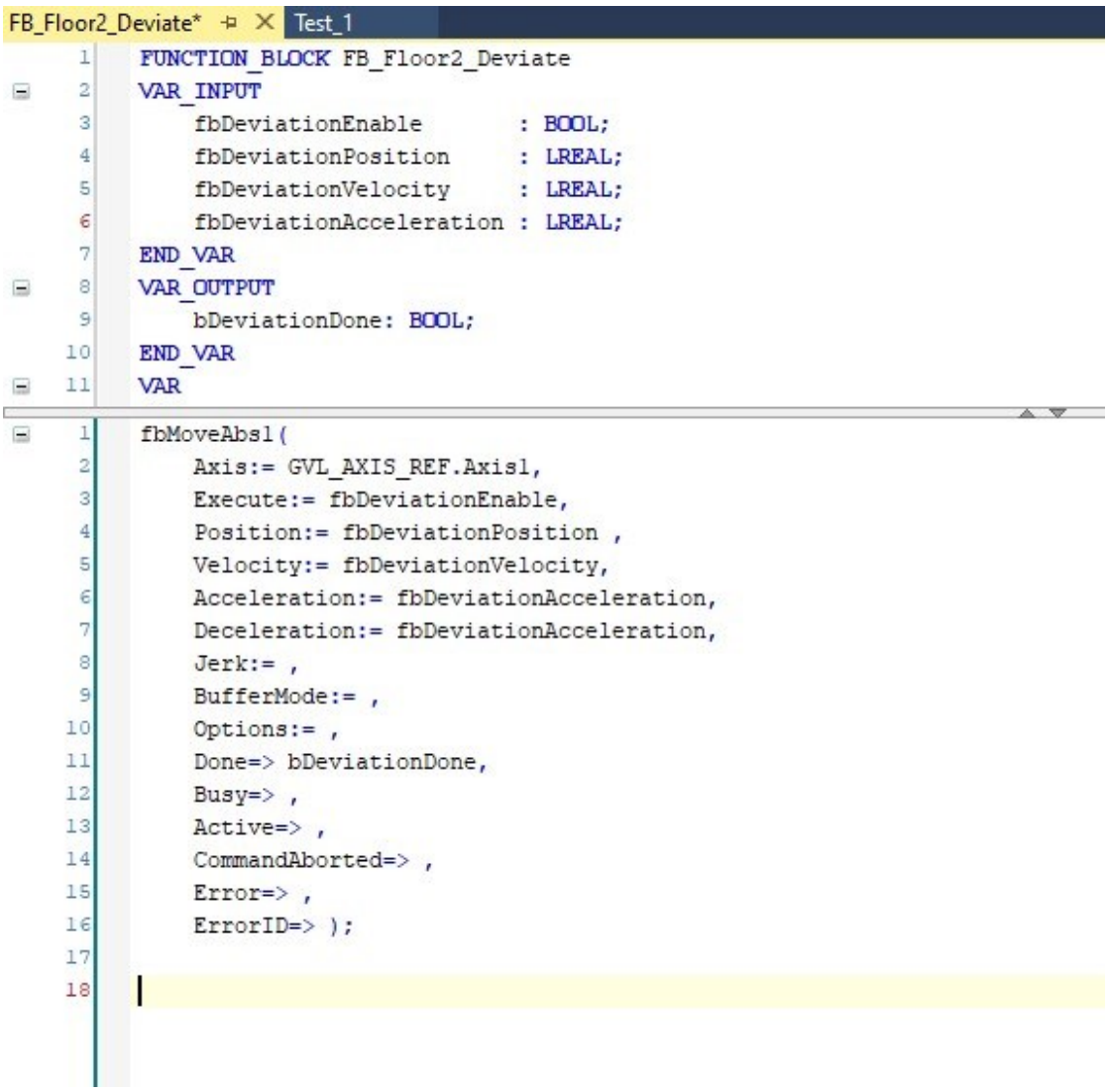


```
PROGRAM Test_1
VAR
    TP1 : TP;
    ET_pulse    : TIME;
    fbDeviate: FB_Floor2_Deviate;
    bDeviationDone: BOOL;

END_VAR
```

```
IF GVL_HMI.bTest_1 AND NOT GVL_IO.bFUI_S2_RDF AND NOT GVL_IO.bOsiA THEN

    TP1(IN:=TRUE, PT:=T#800MS,ET=>ET_pulse);
ELSE
    TP1(IN:=FALSE);
END_IF

IF TP1.Q THEN
    GVL_IO.bCarCall2 := TRUE;
ELSE
    GVL_IO.bCarCall2 := FALSE;
END_IF

IF GVL_IO.bDoorOpenEnd AND NOT GVL_IO.bElevatorMoving AND GVL_HMI.bTestDirection THEN //Small deviation

    fbDeviate (fbDeviationEnable:= TRUE, fbDeviationPosition := 100, fbDeviationAcceleration := 500, fbDeviationVelocity := 300);

ELSIF GVL_IO.bDoorOpenEnd AND NOT GVL_IO.bElevatorMoving  AND NOT GVL_HMI.bTestDirection THEN

    fbDeviate (fbDeviationEnable:= TRUE, fbDeviationPosition := -100, fbDeviationAcceleration := 500, fbDeviationVelocity := 300);
ELSE
    fbDeviate (fbDeviationEnable:= FALSE);
END_IF

IF  bDeviationDone AND GVL_IO.bElevatorMoving THEN                              // Big deviation. Trigger safety monitor

    fbDeviate (fbDeviationEnable:= TRUE, fbDeviationPosition := 2000, fbDeviationAcceleration := 1000, fbDeviationVelocity := 1000);

    ELSIF GVL_IO.bDoorOpenEnd AND NOT GVL_IO.bElevatorMoving  AND NOT GVL_HMI.bTestDirection THEN

        fbDeviate (fbDeviationEnable:= TRUE, fbDeviationPosition := -2000, fbDeviationAcceleration := 1000, fbDeviationVelocity := 1000);
    ELSE
        fbDeviate (fbDeviationEnable:= FALSE);
END_IF
```

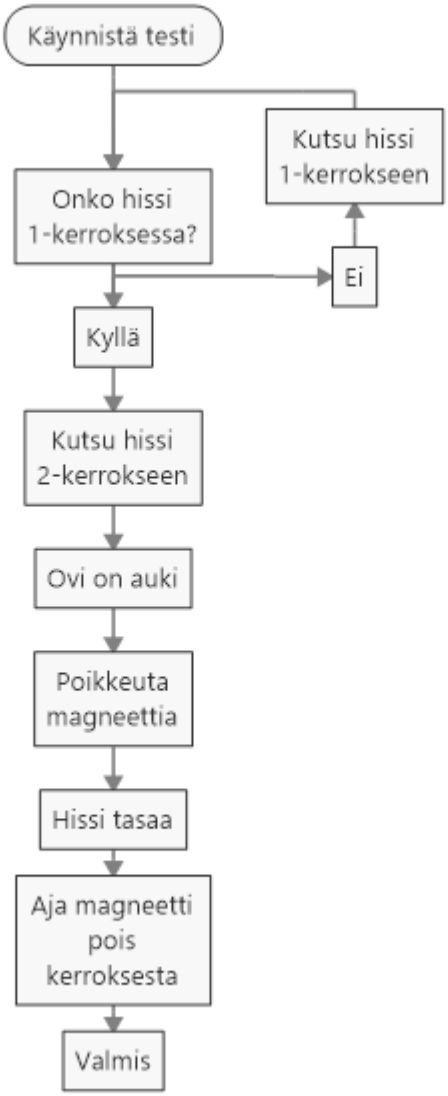
Kuvio 20. Testiohjelman koodi.

Jokaisessa testiohjelmassa kutsutaan erillistä ”FB_Floor2_Deviate” lohkoa, kun halutaan liikuttaa lineaarimoottoria. Lineaarimoottorin ohjaukseen käytetyssä lohkossa on tarvittavat muuttujat, joilla moottoria voidaan liikuttaa. Jokaisessa testiohjelmassa kerrotaan minkä arvon kyseinen muuttuja saa. Halutut arvot, joilla moottoria halutaan ohjata ovat lohkossa sisääntulona nimellä ”VAR_INPUT” ja haluttu lopputulos, kun moottori on suorittanut halutun liikkeen, on lähtönä nimellä ”VAR_OUTPUT”. Tämä säästää aikaa, työtä ja laitteen muistia, kun lineaarimoottorin ohjauksen voi tehdä yhteen lohkoon. Testiohjelmat sitten kutsuvat tarvittaessa ohjauslohkoa kertomalla sille halutun ajosuunnan, nopeuden ja muut tarvittavat arvot. Ohjelmalohkon sisältö on kuvattu ST-kielellä kuviossa 21.



Kuvio 21. Lineaarimoottorin ohjaukseen käytetty lohko.

Logiikan koodin eteneminen on suoraan verrannollista sekvenssiin. Sekvenssillä tarkoitetaan sitä, että halutaan toteuttaa jokin sarja askel kerrallaan, kuten kuviossa 22 on havainnollistettu. Kun sarjan ensimmäinen askel on toteutunut, toteutetaan sarjan seuraava askel, kunnes kaikki askeleet on käyty läpi.

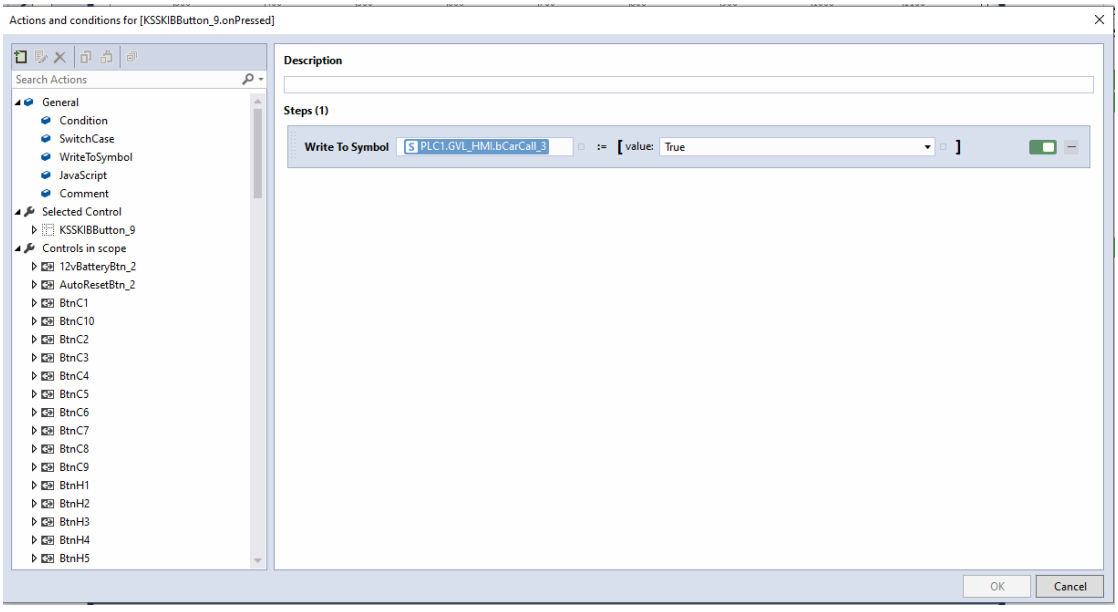


Kuvio 22. Testiohjelman havainnointi

6.8 Ohjelman visualisointi

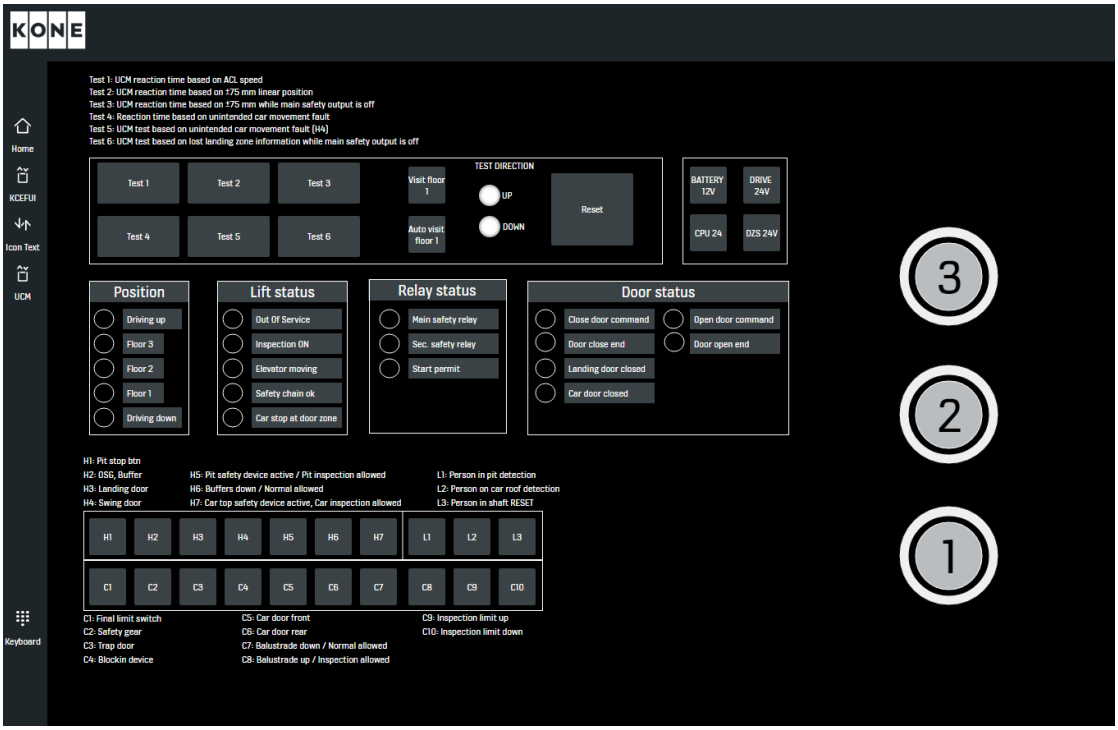
Visualisointi tehdään Beckhoff TwinCAT omalla ”HMI” työkalulla. Työkaluja on kaksi, joista toinen on kevyempään visualisointiin tarkoitettu. Toisella työkalulla voidaan tehdä raskaampia ja näyttävämpiä visualisointeja, mutta vaatii lisenssin. Visualisoinnin tarkoitus on välittää testaajalle hissin tilatietoja sekä antaa testaajalle mahdollisuuden käynnistää testejä sekä ohjata hissin perustoimintoja kuten antaa hissikutsuja.

Visualisointi aloitetaan pohtimalla mitä tietoja halutaan testaajalle näyttää sekä mitä toimintoja halutaan ohjata. Tämän jälkeen rakennetaan haluttu visualisointi käyttämällä valmiita komponentteja kuten painonappeja, mittareita ja tekstikenttiä. Jos ohjelmasta ei löydy halutun näköistä komponenttia, voidaan komponentti itse tehdä. Kun halutun näköinen graafinen ympäristö on rakennettu, täytyy lisätyt indikaattorit ja napit linkityttää ”GVL_HMI” muuttujalistaan, kuten kuviossa 23. Tällöin haluttu indikaattori esimerkiksi muuttuu vihreäksi, kun jokin tulo aktivoituu tai lisätyn napin painallus aktivoi halutun lähdön päälle.



Kuvio 23. Hissin kutsunapin linkitys "GVL_HMI" muuttujalistaan.

Toimeksiannossa käytettiin valmista visualisointipohjaa, missä on valmiina tiettyjä itse tehtyjä toimeksiantajaan viittaavia asioita kuten yrityksen logo tai toimeksiantajan hissin kutsunapin näköinen painike. Valmiiseen pohjaan lisätään sivuja ja sivuille rakennetaan itse haluttu visualisointi. Kuviossa 24 on visualisointi mitä tullaan käyttämään testauksessa. Toimeksiannossa tärkein asia oli rakentaa painikkeet halutuille testeille. Painikkeiden tarkoituksena on saada logiikka suorittamaan haluttu testi. Indikaattoreiksi lisättiin tärkeimmät tulotiedot, jotta varsinkin ensimmäisiä kertoja testejä tehdessä uudella kokoonpanolla olisi mahdollisimman helppo seurata testin edistymistä ja mahdollisissa ongelmatilanteissa ongelman selvittäminen.



Kuvio 24. Käyttöliittymäympäristö

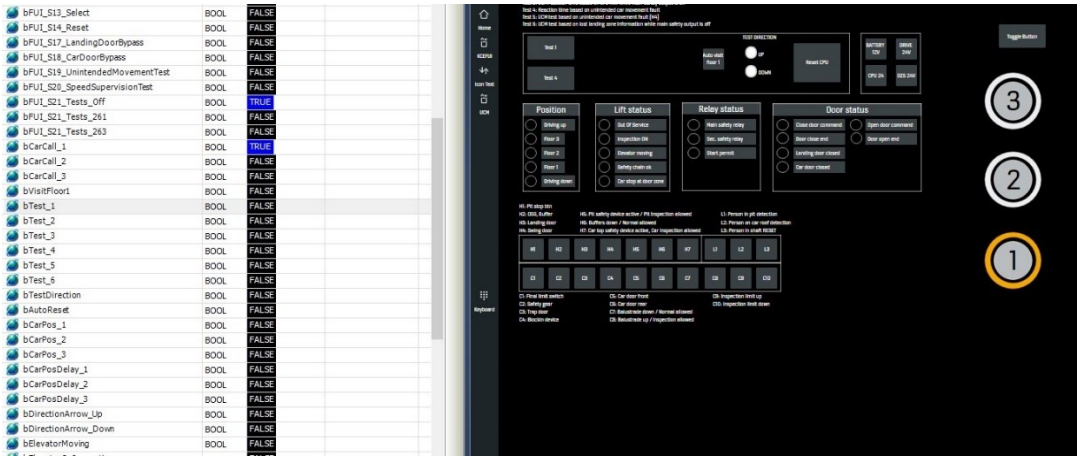
6.9 Testaus

Kun ohjelmat ja visualisointi on saatu tehty, täytyy toiminnallisuudet testata simuloimalla ja laitteistoa testaamalla. Simulointi on tärkeää varsinkin isoimmissa kokoonpanoissa. Tämän tarkoituksena on käydä ohjelmaa läpi ja tutkia toimiiko ohjelma ollenkaan halutulla tavalla ennen kuin varsinaista laitteistoa testataan. Varsinkin suojalaitteiden testaus on erittäin tärkeää, jotta suojalaitteet toimisivat halutulla tavalla. Yleensä ohjelmaa kirjoitettaessa tulee jatkuvasti simuloitua ohjelman toimintaa, jotta virheet saadaan korjattua eikä mahdolliset virheet kertaannu myöhemmissä ohjelman vaiheissa.

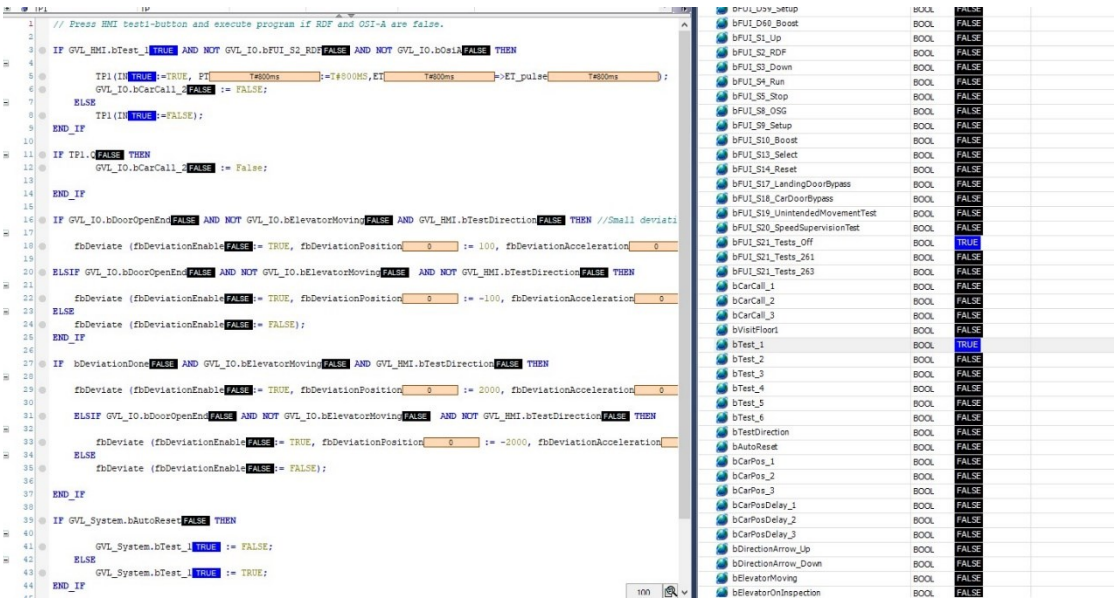
Toimeksiantajan automatisoinnin testaus suoritettiin pelkästään tietokoneella suoritettavan simuloinnin avulla, koska fyysistä automatisointia ei ollut vielä tehty, mikä tarkoitti, että itse laitteiston testaaminen oli mahdotonta. Simulointi suoritetaan halutulle ohjelmalle antamalla muuttujille haluttuja arvoja ja seuraamalla, että ohjelma etenee halutussa järjestyksessä. Tässä vaiheessa ilmenee viimeistään virheitä ohjelman suorittamisessa, mitä ei ole ymmärtänyt ottaa huomioon ja mikä saattaa johtaa myös isoihin korjausmuutoksiin.

Simuloinnissa oli kaksi testattavaa kohdetta. Ensimmäinen on HMI eli käyttöliittymän testaaminen. Käyttöliittymän testauksessa tarkoituksena on kokeilla, että painikkeiden toiminnallisuudet toimivat eli muuttavat haluttujen muuttujien arvoja sekä muuttujien arvot indikoituvat oikein käyttöliittymälle. Esimerkiksi hissikutsun painaminen muuttaa hissikutsulle luodun muuttujan arvoa, kuten kuviossa 25 ja sytyttää samalla kutsunappiin valon, jolla tarkoitus indikoida kutsun olevan aktiivinen. Käyttöliittymän testaus suoritettiin ensimmäiseksi, koska se helpottaa ohjelmiston testausta. Tarvittavien muuttujien kytkeminen päälle tai pois tapahtuu käyttöliittymän kautta helpommin ja samalla indikaattorit indikoivat ohjelman tilaa. Muuttujalistasta haluttujen muuttujien arvojen muuttaminen on työläämpää, koska muuttujat täytyy

etsiä muiden muuttujien joukosta ja arvon muuttaminen vaatii ylimääräisiä painalluksia, kuten kuviossa 26.



Kuvio 25. Käyttöliittymän toimintojen testaaminen.



Kuvio 26. testiohjelman simulointi.

7 Pohdinta

Työn tavoitteena oli suunnitella hissisimulaattorin automatisointia varten tarvittava laitteisto ja tehdä laitteistolle ohjelmisto. Työn tuloksena saatiin hankittua hissisimulaattorille tarvittava laitteisto ja osittain valmiiksi tehty ohjelma. Valmis fyysinen laitteisto ei ehtinyt aikataulusyistä rakentua.

Laitteiston määrittäminen oli melko yksinkertainen prosessi. Piti tietää signaalin toiminnallisuus ja valita tälle sopiva terminaali. Pääyksikön eli CPU:n valinnassa joutui tukeutumaan kuitenkin laitetoimittajaan, koska valinnanvaraa oli paljon eri ominaisuuksilla. Simulaattorin ohjelma saadaan täysin valmiiksi vasta, kun itse simulaattoriin on fyysisesti asennettu hankittu laitteisto. Tämän jälkeen saadaan vielä tehtyä pieniä muutoksia ja korjattua asiat, joihin ei osattu varautua. Automatisoinnin myötä pitäisi ohjelmistotestauksen helpottua huomattavasti, koska logiikka huolehtii suurimman osan asioista ilman että käyttäjän tarvitsee koskea mihinkään. Terminaaleista löytyy myös ylimääräisiä kytkentäpaikkoja, joten laajennusvaraa löytyy tulevaisuutta ajatellen. Lineaarimoottoria voidaan nyt myös ohjata kymmenillä eri parametreilla, kun vanhassa kokoonpanossa ohjausta rajoittivat vipukytkimet ja näille asetetut toiminnot.

Logiikkaohjelmointia tehtäessä monet asiat olisi voinut varmasti tehdä helpommin tai järkevämmällä tavalla. Erityisesti käyttöliittymän kohdalla voisi pohtia voisiko siitä tehdä yksinkertaisemman ja helpommin luettavan. Käyttöliittymää voi kuitenkin helposti muokata tarvittaessa, kun simulaattoria pääsee testaamaan kunnolla. Logiikkaohjelmaa tehtäessä tuli myös monesti tilanteissa, jossa jokin asia ei toiminut niin miten olisi halunnut. Yleensä syy löytyi itse tekijästä, mutta myös laiteteknisiä ongelmia tuli vastaan ja tämän takia muutama asia jäi hieman hämärän peittoon.

Tulevaisuutta ajatellen tämä projekti oli hyvä alku omalle oppimiselle ja alusta simulaattorin jatkokehittämistä varten. Ensimmäinen niin sanottu kehityskohde on tehdä valmiille testerille toimenpiteet käyttöönottoa varten. Kaikki toiminnallisuudet tarkistetaan. Yksinkertaisimmillaan tämä on hissin kutsunapin toiminnan testaamista ja varmistamista, että hissi ajaa haluttuun kerrokseen. Myös kaikki testiohjelmat on

käytävä läpi ja testattava ennen varsinaista testaustoimintaa, jotta tarvittavat vikakoodit ja reaktioajat täsmäävät jokaisen testitapahtuman kohdalla.

Toinen kehittämisen kohde olisi tehdä testiohjelma toimimaan laskurin avulla täysin automaattisesti ja tekemään tarvittava määrä testisuoritteita ilman jatkuvaa valvontaa. Tämä parantaisi tehokkuutta, koska tällöin ei tarvitse sitoa työntekijää simulaattorin luokse. Myös analogisten mittausterminaalien hankinta tulevaisuudessa voi olla ajankohtainen. Analogisella signaalilla voidaan ulkoisesti mitata esimerkiksi kelkan kulkemaa matkaa tai nopeutta.

Lineaarimoottorin ohjaaminen kenttäväylän kautta ja siitä saatava kokemus saattaa myös tuoda uusia sovellutuksia tämän projektin ulkopuolelta, joissa lineaarimoottorin käyttämisestä saattaisi olla hyötyä.

Lähteet

Beckhoff Automation. 2021. Beckhoff Automation tietosivu. Viitattu 25.1.2021. https://www.beckhoff.com/fi-fi/company/

Beckhoff CX5130. 2021. Beckhoff Automation tuotesivu. Viitattu 17.3.2021. https://www.beckhoff.com/fi-fi/products/ipc/embedded-pcs/cx5100-intel-atom/cx5130.html

Beckhoff EK1100. 2021. Beckhoff Automation tuotesivu. Viitattu 17.3.2021. https://www.beckhoff.com/en-us/products/i-o/ethercat-terminals/ek1xxx-bk1xx0-ethercat-coupler/ek1100.html

Beckhoff EK1110. 2021. Beckhoff Automation tuotesivu. Viitattu 17.3.2021. https://www.beckhoff.com/fi-fi/products/i-o/ethercat-terminals/ek1xxx-bk1xx0-ethercat-coupler/ek1110.html

Beckhoff EL1722. 2021. Beckhoff Automation tuotesivu. Viitattu 17.3.2021. https://www.beckhoff.com/en-us/products/i-o/ethercat-terminals/el1xxx-digital-input/el1722.html

Beckhoff EL1809. 2021. Beckhoff Automation tuotesivu. Viitattu 17.3.2021. https://www.beckhoff.com/fi-fi/products/i-o/ethercat-terminals/el1xxx-digital-input/el1809.html

Beckhoff EL2008. 2021. Beckhoff Automation tuotesivu. Viitattu 17.3.2021. https://www.beckhoff.com/en-us/products/i-o/ethercat-terminals/el2xxx-digital-output/el2008.html

Beckhoff EL2624. 2021. Beckhoff Automation tuotesivu. Viitattu 17.3.2021. https://www.beckhoff.com/en-us/products/i-o/ethercat-terminals/el2xxx-digital-output/el2624.html

Beckhoff EL2652. 2021. Beckhoff Automation tuotesivu. Viitattu 17.3.2021. https://www.beckhoff.com/en-us/products/i-o/ethercat-terminals/el2xxx-digital-output/el2652.html

Beckhoff programming languages. N.d. Beckhoff information nettisivu. https://infosys.beckhoff.com/english.php?content=../content/1033/tcplccontrol/html/tcplcctrl_languages.htm&id=529798189732973665

Elevator. 2019. Verkkodokumentti Britannica verkkosivuilla. Viitattu 12.2.2021. https://www.britannica.com/technology/elevator-vertical-transport

EtherCAT – the Ethernet Fieldbus. 2021. Beckhoff Automation tuotesivu. Viitattu 25.1.2021. https://www.beckhoff.com/en-us/products/i-o/ethercat/

Gonzales, C. 2015. Engineering Essentials: What Is a Programmable Logic Controller? Verkkoartikkeli. Viitattu 12.2.2021. https://www.machinedesign.com/learning-resources/engineering-essentials/article/21834250/engineering-essentials-what-is-a-programmable-logic-controller

Tutorial – Introduction to Fieldbus. 2021. Verkkodokumentti. Viitattu 23.8.2021. http://verwertraining.com/tutorals/tutorial-introduction-to-fieldbus-and-profibus/

Historia. N.d. Verkkodokumentti KONE Oyj:n kotisivulla. Viitattu 23.12.2020. https://www.kone.com/fi/yhtio/historia/

KONE Oyj:n vuosikatsaus. 2020. PDF-dokumentti KONE Oyj:n kotisivulla. Viitattu 16.7.2021. KONE_2020_Vuosikatsaus_Painettu raportti_tcm18-101942.pdf

KONE yrityksenä. N.d. Verkkodokumentti KONE Oyj:n kotisivulla. Viitattu 23.12.2020. https://www.kone.com/en/company/ viitattu

KONEen hissilaboratoriot. N.d. Verkkodokumentti KONE Oyj:n verkkosivuilla. Viitattu 23.12.2020. https://www.kone.com/fi/yhtio/innovaatiomme/koneen-hissilaboratoriot/

Linear motors. 2021. LinMot tuotesivu. Viitattu 25.1.2021 https://linmot.com/products/linear-motors/

Referenssit ja tarinat. N.d. Verkkodokumentti KONE Oyj:n kotisivulla. Viitattu 23.12.2020. https://www.kone.fi/referenssit-ja-tarinat/artikkelit/kaiken-takana-on-koysi.aspx

Servo Drives. 2021. LinMot tietosivu. Viitattu 25.1.2021. https://linmot.com/products/servo-drives/

TwinCAT automation software. 2021. Beckhoff Automation tuotesivu. Viitattu 26.01.2021. https://www.beckhoff.com/en-us/products/automation/TwinCAT/