



# Traktorin ohjausmoduulin HiL-simulaatorratkaisu

Joni Nieminen

OPINNÄYTETYÖ  
Toukokuu 2019

Sähkö- ja automaatiotekniikka  
Älykkäät koneet

## TIIVISTELMÄ

Tampereen ammattikorkeakoulu  
Sähkö- ja automaatiotekniikka  
Älykkäät koneet

NIEMINEN JONI:

Traktorin ohjausmoduulin HiL-simulaatorratkaisu

Opinnäytetyö 78 sivua, joista liitteitä 20 sivua  
Toukokuu 2019

---

Eri teollisuudenalojen järjestelmien testaaminen voi olla haastavaa kustannussyistä sekä ne ovat usein aikaa vieviä. Simulaatiotestaus on yksi ratkaisu näihin ongelmiin ja tähän pyrkii myös ENABLE-S3 -tutkimushanke, jossa opinnäytetyön teettävä yritys Creanex Oy on myös mukana. Yhtenä tutkimusaiheena on maatalouden yhteissimulaatio ja siinä mukana oleva maataloustraktori. Opinnäytetyön tarkoituksena oli valmistaa tutkimushanketta tukeva ratkaisu, jossa traktorin oikealla ohjausmoduulilla ohjataan simuloitua traktoria.

Opinnäytetyön tavoitteena oli toteuttaa järjestelmä niin sanottuna HiL-simulaatorratkaisuna, jossa traktorin oikea ohjausmoduuli toimii osana simulaatiota. Tämä mahdollistaa oikealle traktorille tarkoitetun ohjausjärjestelmän ja ohjauksen testaamisen simulaatiomaailmassa. Tavoitteena oli myös toteuttaa kyseisen järjestelmän kompaktiksi kaiken tarvittavan sisältäväksi paketiksi, jota on helppo kuljettaa.

Työssä tutkittiin, mitä simulaatiolla ja HiL-simulaatiolla tarkoitetaan. Järjestelmän suunnittelussa ja toteutuksessa tutkittiin niin sen mekaanista ratkaisua kuin sähköistä kytkentää sekä ohjelmiston kehitystä. Lopuksi tutustuttiin myös järjestelmän testausmenetelmiin. Työ pyrittiin dokumentoimaan kaikilta osin mahdollisimman tarkasti järjestelmän jatkokehityksen ja -tutkimuksen kannalta.

Järjestelmä saatiin valmistettua suunnitellusti muutamaa poikkeusta lukuun ottamatta. Järjestelmän ohjelma pyrittiin tekemään mahdollisimman selkokieliseksi. Sähkösuunnittelusta sekä mekaanisesta suunnittelusta tehtiin selkeät dokumentit, kokoonpanosta otettiin valokuvia ja kokonaisuus tästä kerättiin yrityksen verkkosivuille järjestelmän jatkokehitystä varten.

## **ABSTRACT**

Tampereen ammattikorkeakoulu  
Tampere University of Applied Sciences  
Degree Programme in Electrical and Automation Engineering  
Intelligent Machines

NIEMINEN JONI:  
HiL-Simulation Solution for a Tractor Control Unit

Bachelor's thesis 78 pages, appendices 20 pages  
May 2019

---

This thesis was made for Creanex Oy which is part of the ENABLE-S3 research project. This research project focuses on finding different ways to test industrial systems and solving time- and cost-related problems in the systems by testing them in a simulation environment. One of the research subjects is to create a farming co-simulation which includes a farming tractor. The purpose of this thesis was to plan and manufacture a system for the farming tractor which can be tested in the simulation environment.

The objective of this thesis was to implement the system as an HiL-simulation solution where the tractor's real control unit is part of the simulation. This enables testing of the tractor's controls and control system in the simulation environment. The objective was also to make the system as compact as possible which will be easy to transport from place to place.

The theoretical section explores the simulation and the HiL-simulation. The planning and the implementation phases of the system include mechanical, electrical and software sections. The testing section introduces a way to test the system which was produced in this thesis.

The system was manufactured as planned with some exceptions. All the documentation of this system was designed to be as clear as possible to support further research. The documentation concerning mechanical and electrical plans, software code, and photographs of the system was collected together into one location for the company.

---

Key words: simulation, hardware-in-the-loop, farming, tractor, enable-s3

## SISÄLLYS

1	JOHDANTO .....	7
2	SIMULAATION KÄYTTÖ TUOTEKEHITYKSESSÄ.....	8
2.1	HiL -simulaatiotestaus.....	9
2.2	Creanex AgSim.....	10
3	SIMULAATIOJÄRJESTELMÄN ESITTELY .....	12
3.1	HY-TTC 580 -ohjausmoduuli.....	12
3.2	MultIO24s -liityntäkortti.....	15
4	CODESYS -OHJELMOINTIYMPÄRISTÖ .....	17
5	CAN-VÄYLÄ JA CANOPEN -PROTOKOLLA .....	18
6	JÄRJESTELMÄN SUUNNITTELU .....	21
6.1	Mekaaninen suunnittelu .....	22
6.2	Sähköinen suunnittelu .....	28
6.3	Ohjelmiston suunnittelu.....	30
7	JÄRJESTELMÄN TOTEUTUS.....	32
7.1	Kokoonpano ja sähköinen kytkentä.....	32
7.2	Ohjausmoduulin ohjelma ja sen ohjelmointi .....	37
7.3	Simulaattorin konfigurointi.....	41
8	JÄRJESTELMÄN TESTAUS .....	46
8.1	Manuaaliohjauksen testaus.....	47
8.2	Ulkaisen ohjauksen testaus ja testitapaukset.....	51
9	POHDINTA .....	55
	LÄHTEET .....	56
	LIITTEET .....	59
	Liite 1. Alustava komponenttisijoittelukuva.....	59
	Liite 2. Liitinpaneeli .....	60
	Liite 3. Käyttäjäohjainpaneelin tekninen piirustus .....	61
	Liite 4. Tukijalat, tekninen piirustus .....	62
	Liite 5. Kantokahvan tekninen piirustus.....	63
	Liite 6. Järjestelmän sähkökytkentäkuva.....	64
	Liite 7. Järjestelmän ohjelma: Global Variable List .....	67
	Liite 8. Järjestelmän ohjelma: PLC_PRG.....	69
	Liite 9. Järjestelmän ohjelma: CANComm .....	70
	Liite 10. Järjestelmän ohjelma: CNTRLS_PRG .....	72
	Liite 11. Järjestelmän ohjelma: Safety_CHK.....	77
	Liite 12. Järjestelmän ohjelma: PID .....	78



## LYHENTEET JA TERMIT

AgSim	Creanex Oy:n kehittämä maataloussimulaatio.
Ajoalgoritmi	Tässä työssä tarkoitetaan traktorin autonomiseen ajoon liittyvää komentosarjaa.
Bootloader	Suomeksi : alkulatausohjelma. Järjestelmän prosessorille asennettu ohjelma, joka käynnistyessään käynnistää tarvittavat ohjelmakomponentit.
CAN	Controller Area Network. Teollisuudessa käytetty väylätekniikka.
CANopen	Väyläprotokolla, joka toimii CAN-väylän avulla.
Co-simulation	ks. Yhteissimulaatio
DIN-kisko	Sähköasennuksissa standardisoitua kiinnitysmenetelmää varten oleva kisko.
D-Sub, D-liitin	Elektroniikassa käytetty liitintyyppi.
ECU	Electronic Control Unit; yleisesti ajoneuvoissa käytetty elektroniikkajärjestelmien ohjaamiseen tarkoitettu elektroninen ohjausyksikkö.
ENABLE-S3	Eurooppalainen tutkimushanke, jossa keskitytään eri alojen autonomisten järjestelmien testauksen kehittämiseen.
FBD	Function Block Diagram; Standardin mukainen ohjelmointikieli.
HiL	Hardware-in-the-Loop; simulaatiotestaustapa, jossa simulaatioon yhdistetään tutkittavan järjestelmän osa.
I/O	Input / Output; Sisään- ja ulostulo
ini-tiedosto	Windows alustalla yleisesti käytetty konfigurointitiedosto.
Instruction List, IL	Instruction List; Standardin mukainen ohjelmointikieli.
LIN	Local Interconnect Network. Ajoneuvoteollisuudessa käytetty sarjamuotoinen väyläprotokolla.
LD	Ladder; Standardin mukainen ohjelmointikieli.
Muunnostaulu	Tässä työssä: Muuntaa signaalit simulaation tai liityntäkortin tunnistamaan muotoon.

Node	CAN-väylään kytkeytynyt laite, jolla on mahdollisuus lähettää tai vastaanottaa viestejä.
Objektikirjasto	CANopen protokollassa käytetty taulu, johon sisään- ja ulostulot sekä kommunikointiin liittyvät asetukset kirjataan.
Ohjausmoduuli	ks. ECU
Ohjelmointiympäristö	Ohjelma, jota käytetään ohjelmiston suunnitteluun ja toteutukseen.
OSI-malli	Kansainvälisen standardisoimisjärjestön ISO:n kehittämä malli, joka kuvaa tiedonsiirtoprotokollien yhdistelmää.
PID-säädin	Proportional-Integral-Derivative -säädin. Säättötekniikassa käytetty säädintyyppi, joka laskee ulostuloarvon mitatun ja halutun arvon virheen mukaan.
PWM	Pulse Width Modulation, Pulssinleveys modulaatio. Digitaalinen signaali, jossa pulssisuhteen avulla muutetaan signaalin arvoa.
SFC	Sequential Function Chart; Standardin mukainen ohjelmointikieli.
ST	Structured Text; Standardin mukainen ohjelmointikieli.
TestRunner	Creanex Oy:n kehittämä ohjelmisto simulaatiojärjestelmien testausta varten.
Yhteissimulaatio	Simulaatio, jossa samassa simulaatiomaailmassa käytetään useampaa simuloitua laitetta.

## 1 JOHDANTO

Opinnäytetyö liittyy työn teettäneen yrityksen Creanex Oy:n osuuteen eurooppalaisessa ENABLE-S3 -tutkimushankkeessa, jossa isona tavoitteena on luoda menetelmät ja niitä tukevat simulointiin perustuvat työkalut autonomisten ohjausjärjestelmien toiminnan ja luotettavuuden kustannustehokkaaseen testaamiseen. Tutkimushankkeen yhtenä osa-alueena rakennetaan testausympäristö maataloussimulaatiolle, jossa voidaan testata usean laitteen autonomista yhteistoimintaa ja ohjausta viljapellon puinnissa.

Opinnäytetyön tarkoituksena on valmistaa tutkimushanketta tukeva ratkaisu, jossa traktorin oikealla ohjausmoduulilla ohjataan simuloitua traktoria. Tämän kaltaisen menetelmä on niin sanottu HiL-, eli Hardware in the Loop -ratkaisu. Tämä mahdollistaa oikean traktorin ohjaukseen tarkoitetun ohjelmiston ja ohjauslaitteiston testauksen ja kehityksen simuloitussa ympäristössä. Simulointiratkaisua varten rakennetaan ohjausmoduulin ympärille järjestelmä, joka voidaan kytkeä simulointia ajavaan tietokoneeseen. Tämä mahdollistaa ohjausmoduulin ohjauksen järjestelmässä. Järjestelmää lopuksi vielä testataan eri testitapauksilla, joilla voidaan testata esimerkiksi järjestelmän suojaominaisuuksia. Tarkoituksena ei ole pyrkiä ratkaisemaan traktorin autonomista ohjausta, vaan luoda sille alusta jatkokehitystä varten.

Opinnäytetyön tavoitteena on luoda järjestelmä ja sille esimerkkisovellus, jota käyttämällä ENABLE-S3 -tutkimushankkeessa kehitettyjä työkaluja ja menetelmiä voidaan edelleen kehittää ja demonstroida. Tavoitteena on saada järjestelmästä mahdollisimman kompakti ja mahdollisesti myös ulkoisesti näyttävä, koska sitä on tarkoitus esitellä tutkimushankkeen viimeisessä tapahtumassa.

Työn teettävä yritys Creanex Oy on tamperelainen osakeyhtiö, joka kehittää simulaatoriratkaisuja teollisuuteen, koulutussimulaatioympäristöjä oppilaitoksille sekä tarjoaa myös tuotekehityspalveluita teollisuuden asiakkailleen. Yhtiö myös valmistaa Tampereen toimipisteessään simulaattorit asiakkailleen. Yhtiö työllistää henkilöstöä niin ohjelmisto-, mekaniikka- ja sähkösuunnittelun parissa, kuin myös sähkö- ja mekaniikka kokoonpanonkin puolella.

## 2 SIMULAATION KÄYTTÖ TUOTEKEHITYKSESSÄ

Simulaation ideana on luoda oikean maailman järjestelmästä mahdollisimman todennukainen jäljitelmä esimerkiksi virtuaalimaailmaan tietokoneavusteisesti. Simulaatiota käytetään, kun pyritään selvittämään jonkin järjestelmän käyttäytymistä eri tilanteissa. Monet järjestelmän osat pystytään mallintamaan matemaattisilla yhtälöillä, joita voidaan käyttää virtuaalimaailmassa jäljittelemään oikean maailman käyttäytymistä osilla. Simulaatiota varten voidaan myös osien matemaattisia malleja yksinkertaistaa simulaation tehokkuutta varten, ottamalla huomioon vain seikkoja, jotka ovat tärkeitä tutkimuskohteessa. (Technopedia 2019.)

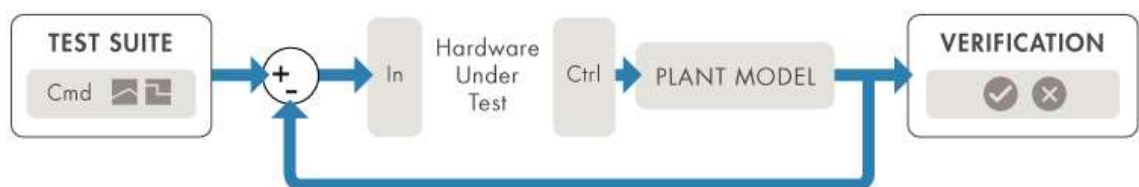
Simulaatiota voidaan käyttää esimerkiksi yrityksen uuden tehtaan rakentamisen suunnittelussa. Tällaiseen simulaatioon luotaisiin uuden tehtaan ympäristö laitteineen ja työntekijöineen virtuaalisina matemaattisina malleina. Näiden mallien avulla yritys voi luoda ympäristön, jossa voidaan testata ja kehittää uuden tehtaan tuottavuutta, valmistuskapasiteettia ja uuden tehtaan kokonaiskustannuksia. Tällaisessa tapauksessa simulaatio on erittäin hyödyllinen työkalu, koska uuden tehtaan toimintaa voidaan kustannustehokkaasti kehittää jo ennen kuin sitä on edes rakennettu. (Smith 1998.)

Simulaatioissa huonona puolena on se, että se voi olla vain niin tarkka, kuin määritetyt matemaattiset mallit. Matemaattisia malleja joudutaan usein yksinkertaistamaan tietokoneilla ajettuihin simulaatioihin laskentatehon ja ylipäättään niiden monimutkaisuuksien takia. Usein myös järjestelmän oletettua toimintaa ei saada täysin muutettua matemaattisiksi malleiksi. (BusinessDictionary n.d.)

## 2.1 HiL -simulaatiotestaus

Tähän työhön keskeisesti liittyvä HiL-simulaatio, eli Hardware-In-the-Loop -simulaatio on simulaatio, jossa simulaatiomallit testataan oikean ohjauslaitteiston avulla. HiL-simulaatio tai HiL-simulaatiotestaus tehdään reaaliajassa toimivaksi, toisin kuin muita simulaatiomalleja voidaan esimerkiksi nopeuttaa tai hidastaa tarkastelua varten. HiL-simulaatiotestauksen avulla laitteistosta pystytään tarkastelemaan esimerkiksi laitteiston vasteaikoja siihen osoitettujen ohjauksien mukaan. HiL-simulaatiossa virtuaalinen malli luodaan tietokoneelle, josta simulaatiosta saadut arvot voidaan syöttää simulaatioon kytkettyyn laitteistoon tai järjestelmään. (What Is Hardware-In-The-Loop Simulation? n.d.)

Tässä työssä HiL-simulaatiossa käytetään TTControlin valmistamaa ECU:a, eli ohjausmoduulia. Simulaatiomalli ajetaan tietokoneella Creanexin kehittämän simulaatiosovelluksen AgSim:in avulla. Ohjausmoduulille ohjelmoidaan traktorin käyttölaitteiden, antureiden ja väyläjärjestelmien toimintaa jäljittelevät toiminnot, joita pystytään käsittelemään simulaatiomallissa.



KUVIO 1. HiL-simulaatiotestauksen periaate (Hardware-in-the-Loop (HiL) Simulation n.d.)

Kuviossa 1 on Mathworksin esittämä tapa HiL-simulaatiotestaus järjestelmästä (Hardware-in-the-Loop (HiL) Simulation n.d.). Tähän työhön liittyen kuvioon voitaisiin kuvitella "Test suite" kohdalle esimerkiksi traktorin käyttölaitteet. TTControlin ohjausmoduuli sijoitettaisiin "Hardware under test" kohtaan ja simulaatiomalli voitaisiin sijoittaa "Plant model" kohtaan. Työn lopuksi tehdyt testit järjestelmälle olisi kohdassa "Verification".

## 2.2 Creanex AgSim

AgSim on kehitetty aikaisemmin ENABLE-S3 -tutkimushankkeen puitteissa. AgSim soveltuu niin HiL-simulaatiotestaukseen, kuin myös muihinkin simulaatiotestaus ratkaisuihin. AgSim on simulaatioympäristö, jossa pystytään käyttämään maataloudessa esimerkiksi viljan puimisessa käytettyjä työlaitteita niiden ominaisissa ympäristöissä. Simulaatiossa voidaan käyttää traktoria ja leikkuupuimuria. Traktoriin voidaan liittää myös erinäisiä työkaluja, joita simulaatiossa voidaan käyttää. Näitä ovat esimerkiksi viljakärri, heinäpaalipihdit, jousipiikkiäes, kylvölannoitin ja etukauha.

Hankkeessa käytössä olevassa AgSim versiossa on myös mahdollisuus nelikopterin, eli dronen ohjaamiseen. Nelikopteria käytetään simulaatiossa siihen, että siihen liitettyllä simuloidulla hyperspektrikameralla voidaan kuvata viljapelto. Kuva joukosta, joka saadaan hyperspektrikameralta, voidaan luoda eräänlainen kartta, jolla voidaan tutkia esimerkiksi viljan kypsyysaste ja kasvuala. Nelikopteri toimii myös autonomisesti toimivan leikkuupuimurin konenäkönä puintivaiheessa.



KUVA 1. AgSim -simulaation visuaali yhteissimulaatiosta (Creanex Oy)

AgSim on kehitetty hanketta varten erityisesti toimimaan niin sanotussa yhteissimulaatiossa (co-simulation). Kuvassa 1 on esitetty yhteissimulaation toiminta. Hankkeessa tarkoituksena on saada kolme laitetta, puimuri, traktori ja nelikopteri, toimimaan autonomisesti yhteistyössä saman simulaation sisällä. Nelikopterin

tehtävänä on kuvata hyperspektrikameralla ja toimia puimurin näköelimenä. Helikopteri kuvaa puimurin etualaa puinti vaiheessa ja ilmoittaa puimurille eteen tulevista esteistä kuten esimerkiksi kivistä tai liikkuvista eläimistä. Pellon kuvantamisvaihe, kuten myös puimurin näköelimenä toimiminen on tarkoituksena tapahtua autonomisesti. Puimurin tehtävänä on puida kypsä vilja pellolta. Puimurille on jo kehitetty hanketta varten ajoalgoritmi, hankkeessa mukana olevan toisen yrityksen toimesta, jonka avulla tämä on mahdollista. Traktorin tehtävänä on seurata puimuria siten, että puimuri voi oman säiliönsä täytyttyä tyhjentää sen traktorin vetämään viljakärryyn. Traktorin autonominen osuus kehitetään vasta myöhemmin hankkeen ulkopuolella.

### 3 SIMULAATIOJÄRJESTELMÄN ESITTELY

Työssä käytetyn simulaatiojärjestelmän tärkeimmät osat ovat TTControlin valmistama HY-TTC 580 -ohjausmoduuli ja Creanexin valmistama MultilO24s -liityntäkortti. Näiden lisäksi käytetään simulaation manuaaliohjaukseen käyttäjäohjainpaneelia, jossa on vipuohjaimia ja kytkimiä ohjausta varten. Nämä järjestelmän osat muodostavat yhdessä HiL-simulaation laitteisto-osuuden. Näiden lisäksi järjestelmään kuuluu myös edellisessä kappaleessa esitelty AgSim -simulaatio-ohjelmisto. Tässä kappaleessa keskitytään kuitenkin järjestelmän laitteistopuoleen, pois lukien erikseen suunniteltavaan käyttäjäohjainpaneeliin. Käyttäjäohjainpaneelin esittely ja suunnittelu löytyy kappaleesta 6.

Järjestelmää varten tutkimushankkeessa mukana oleva TTControl toimitti heidän valmistaman HY-TTC 580 -ohjausmoduulin. Heiltä oli mahdollisuus vuokrata kyseinen laite ja kehittää sen avulla työkaluja ja menetelmiä hankkeen puitteissa. Tästä syystä järjestelmä päätettiin toteuttaa kyseisellä ohjausmoduulilla. Ohjausmoduulin yhdistäminen simulaatioon toteutettiin Creanex MultilO24s -liityntäkortilla. Kyseinen kortti valittiin käytettäväksi työhön, koska Creanex käyttää sitä myös muissa heidän valmistamissaan vastaavissa järjestelmissä. Laitteiston manuaaliohjausta varten valitut käyttäjäohjainpaneelin ohjaimet valittiin yrityksen muista projekteista ylijääneistä käyttämättömistä osista.

#### 3.1 HY-TTC 580 -ohjausmoduuli

Opinnäytetyössä käytetty ohjausmoduuli kuuluu TTControlin HY-TTC 500 -tuoteperheeseen. Tuoteperheessä on viisi eri ominaisuuksia sisältävää vaativaan käyttöön tarkoitettua turva ohjausmoduulia, joista yksi on työssä käytetty HY-TTC 580. Kuvassa 2 on esitetty kyseinen malli. (HY-TTC500 Family n.d.)





KUVA 2. TTControl HY-TTC 580 -ohjausmoduuli (HY-TTC500 Family n.d.)

Tuoteperheen kaikissa ohjausmoduuleissa on käytössä 32-bittinen 180 MHz taajuudella toimiva ARM Cortex-R4 -mikroprosessori. Valmistaja tukee ohjausmoduulien ohjelmointiin C-kieltä ja CoDeSys V3.5 SP10 -ohjelmointiympäristöä. Tuoteperheen ohjausmoduulit täyttävät useita eri standardeja ja vaatimuksia, esimerkiksi IEC 61508, EN ISO 13849 ja ISO 25119, jotka ovat koneturvallisuuteen ja työkonoiden turvallisuuteen liittyviä standardeja. (HY-TTC 500 Family flyer n.d.)

HY-TTC 580 -ohjausmoduuli on koteloitu hyvin kompaktiin alumiiniseen vaativia olosuhteita kestäväään koteloon. Kotelointi ja laitteen rakenne mahdollistaa sen käytön hyvin laajalla lämpötila-alueella. Ohjausmoduulin laaja käyttöjännite alue mahdollistaa eri jännitesyöttömenetelmät (HY-TTC500 Family n.d.). Taulukossa 1 esitetään ohjausmoduulin sähköiset ja fyysiset ominaisuudet.

TAULUKKO 1. Ohjausmoduulin sähköiset ja fyysiset ominaisuudet (High Performance Safety Controller... n.d.)

Ominaisuus				Yksikkö
Ulkomitat	231.3	x	204.9	x mm
Ulkomitat huomioiden	315.3	x	204.9	x mm
Paino	1200			g
Liitin	154			pinniä
Käyttölämpötila	-40 to +85			°C
Käyttökorkeus	0 - 4000			m
Käyttöjännite	8 - 32			V
Huippu käyttöjännite	45			Vmax

<b>Syötön virta 12 V / 24 V</b>	400/200	mAmax
<b>Valmiustilan virta</b>	<1	mAmax
<b>Virta täydellä kuormalla</b>	60	Amax

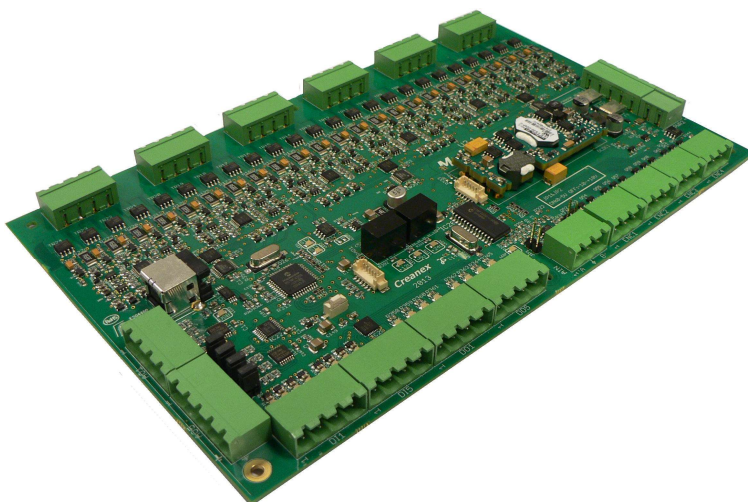
HY-TTC 580 -ohjausmoduulissa on Texas Instrumentsin TMS570 ARM -prosessori, kuten myös muissakin tuoteperheen laitteissa. Se sisältää useita liityntärajapintoja, niin väylä- ja sarjaliikenne sekä sisään-, ja ulostulokanavien muodossa. Näitä ovat esimerkiksi useat CAN-, Ethernet- ja LIN-väylät sekä yhteensä 60 kappaletta sisääntuloja ja 36 kappaletta ulostuloja. Sisään- ja ulostulokanavat voidaan myös sovelluksella asettaa toimimaan useilla eri toimintatavoilla. Taulukoon 2 on listattu HY-TTC 580 -ohjausmoduulin esitteen mukaan prosessoritiedot, liitännät sekä sisään-, ja ulostulojen tiedot.

TAULUKKO 2. HY-TTC 580 -ohjausmoduulin prosessoritiedot, liitännät sekä sisään- ja ulostulot (High Performance Safety Controller... n.d.)

<b>HY-TTC 580</b>	
<b>Prosessori</b>	32-bittinen TI TMS570, ARM cortex-R4F pohjainen, 180 MHz
<b>Muisti</b>	3 Mt / 8 Mt sis./ulk. Flash, 256 kt / 2 Mt sis./ulk. RAM, 64 kt ulk. EEPROM
<b>Liitännät</b>	7 kpl – CAN (50 kbit/s – 1 Mbit/s)
	1 kpl – Ethernet (10 Mbit/s)
	1 kpl – LIN
	1 kpl – RS232
<b>Ulostulot</b>	36 kpl – PWM / Digitaali, 4 A
	8 kpl – Digitaali, ylös vetävä, 4 A
	8 kpl – Digitaali, alas vetävä, 4 A
	8 kpl – Monikäyttö ulostuloa / analogista sisääntuloa
<b>Sisääntulot</b>	20 kpl – Analogista sisääntuloa, 12-bittiä
	12 kpl – Digitaalista laskuria (0,1 Hz – 20 kHz)
	K15 ja heräte
<b>Jänniteulostulo</b>	2 kpl – Anturijännite syöttöä, 5V / 500 mA
	1 kpl – Anturijännite syöttöä, 5-10 V / 2,5 W

### 3.2 MultilO24s -liityntäkortti

MultilO24s -liityntäkortti on Creanexin valmistama USB väylään liitettävä useita sisään- ja ulostuloja sisältävä kortti. Kortti voidaan yhdistää simulaatio-sovellukseen ja käyttää sisään- ja ulostuloja simulaatiossa. Kuvassa 3 on esitetty MultilO24s -liityntäkortti.



KUVA 3. Creanex MultilO24s -liityntäkortti (Creanex Oy)

I/O-korttia voidaan käyttää joko 5 V tai 24 V jännitteillä. Jos 5 V jännitesyöttöä ei käytetä, se voidaan muuttaa kortilla 5 V jännitelähteeksi, joka luodaan 24 V syöttöjännitteestä. Kortissa on yhteensä 34 kanavaa sisääntuloja ja 20 tai 28 kanavaa ulostuloja varten, riippuen siitä millaiseen käyttötarkoitukseen kanavat ovat otettu käyttöön. Taulukossa 3 on listattu kortin sähköiset ominaisuudet sekä fyysiset mitat ja taulukossa 4 sisään- ja ulostulokanavat.

TAULUKKO 3. MultilO24s -liityntäkortin sähköiset ja fyysiset ominaisuudet (Creanex MultilO24s V2.2 Specification V1.5 2016)

		Yksikkö
<b>Käyttöjännite</b>	5 (4,8 - 5,5)	V
	24 (18 - 30)	V
<b>Syötön virta</b>	0,5 (5 V)	A
	1,5 (24 V)	A
<b>Ulkomitat</b>	222,25 x 134,62	mm

TAULUKKO 4. MultiIO24s -liityntäkortin sisään-, ja ulostulokanavat (Creanex MultiIO24s V2.2 SpecificationV1.5 2016)

<b>Sisääntulot</b>	8 kpl – Digitaali sisääntuloa, 5 V - 70 V
	2 kpl – Analogista sisääntuloa, 10-bittia
	24 kpl – PWM pulssinleveysmodulaatio sisääntuloa, 1,1 A tai 24 kpl – Digitaali sisääntuloa, min. 15 V
<b>Ulostulot</b>	8 kpl – Digitaali ulostuloa, 80 mA
	12 kpl – Digitaali ulostuloa
	tai 4 kpl – Kulma-anturi (Quadrature encoder) lähtöä
	8 kpl – Analogista ulostuloa, 12- tai 16-bittinen

## 4 CODESYS -OHJELMOINTIYMPÄRISTÖ

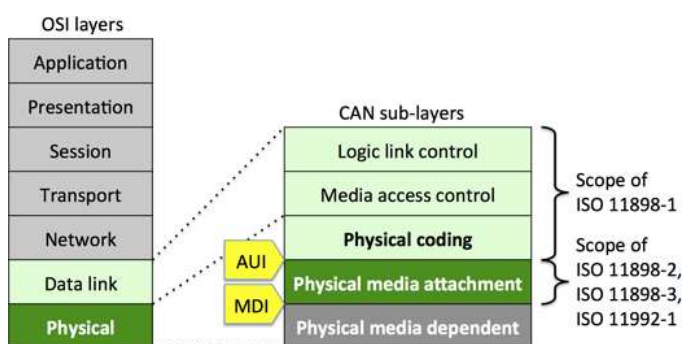
Opinnäytetyössä käytetään CoDeSys -ohjelmointiympäristöä ohjausmoduulin ohjelmointiin. CoDeSys -ohjelmointiympäristön on kehittänyt saksalainen ohjelmistoyritys 3S – Smart Software Solutions GmbH. CoDeSys on IEC 61131-3 standardin mukainen ohjelmointiympäristö (CoDeSys n.d.). Tämä tarkoittaa sitä, että ohjelmointiympäristö täyttää standardin mukaiset vaatimukset syntaksin, datatyypin, muuttujan, ohjelmalohkojen sekä ohjelmointikielten osalta. Standardin mukaan ohjelmointiympäristöstä löytyy ohjelmointikieliä yhteensä viisi erilaista, joista kaksi ovat tekstipohjaisia ohjelmointikieliä: IL, eli Instruction List ja ST, eli Structured Text ja kolme graafiseen ohjelmointiin perustuvaa ohjelmointikieliä: LD, eli Ladder; FBD, eli Function Block Diagram ja SFC, eli Sequential Function Chart. (International standard IEC 61131-3:2013)

Työssä käytetään CoDeSys 3.5 SP10, joka toimitettiin ohjausmoduulin mukana. Ohjelmointiympäristöön on lisätty TTControlin toimesta ohjelmakirjastoja muun muassa helpottamaan ohjausmoduulin sisään- ja ulostulojen käyttöä ja ohjelmointia. Ohjausmoduulin ohjelmointiin käytettiin pääasiassa tekstipohjaista Structured Text -ohjelmointikieltä.

TTControl on luonut CoDeSys ympäristöön HY-TTC 500 -laiteperheelleen ohjelmakirjaston, joka auttaa laitteen ohjelmointia ja sen käyttöönottoa ohjelmointiympäristössä. Kirjastossa on muun muassa lueteltuja tyyppejä, eli enumeraatio tyyppejä sisään- ja ulostulojen muuttujien hallintaan. Nämä tyypit sisältävät I/O-kanavalle sen ominaiset arvot ja kirjasto automaattisesti kohdistaa oikeat arvot näihin. Näiden avulla esimerkiksi analogisesta sisääntulosta kyetään tallentamaan arvo sitä vastaavaan muuttujaan luokassa (class) ja jatko käsitellä sitä tämän kautta. Kirjasto sisältää myös laajasti väylälaitteiden käyttöönottoon liittyviä funktioita, laitteen tehonsyötön kytkentään ja turvakytken toimintaan.

## 5 CAN-VÄYLÄ JA CANOPEN -PROTOKOLLA

CAN-väylä, eli Controller Area Network, on saksalaisen Robert Bosch GmbH:n kehittämä väyläratkaisu. Se on ensimmäisenä esitelty jo vuonna 1983. (History of CAN technology n.d.) CAN väylä on laajasti käytetty väylätekniologia ajoneuvotekniikassa, johon se on alun perin myös kehitetty, mutta nykyään myös esimerkiksi työkoneissa sekä erinäisissä teollisuuslaitteistoissa. Sitä käytetään myös lääketeollisuudessa ja sairaalalaitteistoissa. (Introduction to the... 2016, 2.) CAN-väylä toimii varsinaisesti OSI-mallin mukaisesti kerroksilla 1 ja 2, eli fyysinen kerros ja siirtokerros (kuvio 2). Kuviosta 2 nähdään myös, että fyysinen kerros jakautuu erikseen vielä kolmeen eri osa-alueeseen ja siirtokerros jakautuu kahteen osa-alueeseen. (CAN physical layer n.d.)

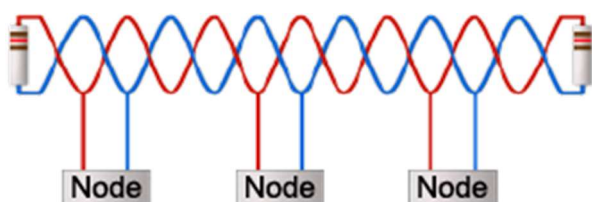


KUVIO 2. OSI-mallin mukainen esitys CAN-väylästä (CAN physical layer n.d.)

CAN -kommunikaatioprotokollasta on kahta eri versiota. Niin kutsuttu standardi versio, joka käyttää 11-bittistä viestitunnistetta CAN -viesteille, sekä laajennetun extended -version, joka käyttää 29-bittistä viestitunnistetta. Standardi 11-bittisen viestitunnisteen kanssa voidaan käyttää 2048 eri viestitunnistetta ja 29-bittisessä laajennetussa versiossa viestitunnisteiden määrä voi olla 537 miljoonaa kappaletta (Introduction to the... 2016, 3.). Tiedonsiirron nopeus väylässä määrittyy väylän johtimien pituuden mukaan. Mitä pidempi väylä on, sitä hitaampi tiedonsiirtonopeuden täytyy olla. Enimmäisnopeudellaan 1 Mbit/s väylän pituus saa olla enintään 40 metriä pitkä kun taas hitaimmalla nopeudella 10 kbit/s väylä voi olla jopa 6000 metriä pitkä. (Introduction to the... 2016, 7.) Tässä työssä käytetään simulaatiosovelluksen AgSimin vakionopeutta 500 kbit/s. Kyseisellä nopeudella väylän maksimipituus standardin mukaan on 100 metriä (What is the Maximum...

2019). Näin ollen väylän pituusrajoitus ei tule työssä vastaan kyseisellä nopeudella järjestelmän kompaktin rakenteensa vuoksi.

CAN-väylän rakenne, tai OSI -mallin mukaisesti fyysinen kerros, on usein toteutettu kierretyllä parikaapelilla. Liittimiksi on vakioitunut yleisesti käytetty D-sub 9-pinninen liitin, mutta sitä ei ole standardisoitu. (CAN physical layer n.d.) Verkon rakenne, eli verkkotopologia, voi olla joko linja-, rengas tai tähtityyppinen. Kaikissa verkkotopologioissa verkossa täytyy olla päätevastus/-vastukset. Päätevastukset vähentävät/poistavat väylään muodostuvat yliaallot, eli signaalin ”kimpoamiset” väylän päätteestä ja näin ollen signaalin kahdentamisen. Kuviossa 3 esitetään yksinkertainen malli linjatyyppisestä verkon rakenteesta, johon yhdistetty kolme toimilaitetta, eli nodea. (CAN lower- and higher-layer protocols n.d.)



KUVIO 3. CAN linjatyyppinen verkkotopologia, jossa kolme erillistä laitetta. Väylän päädyissä päätevastukset. (CAN lower- and higher-layer protocols n.d.)

Väylään voidaan liittää laitteita, mihin kohtaan väylää vain, huomioiden väyläjohtimen pituusrajoitukset. Väylään kytkettyjä laitteita kutsutaan nodeiksi, eli solmuiksi. Väylässä jokainen laite vastaanottaa jokaisen viestin. Väylään kytketyille laitteille määritetään erikseen, mitkä viestit kyseiselle laitteelle kuuluu. Tämä mahdollistaa myös sen, että väylään voidaan kytkeä myös jälkeempään lisää laitteita ilman, että ennestään kytkettyihin laitteisiin täytyisi tehdä muutoksia. Tästä hyvänä esimerkkinä on ajoneuvon CAN-väylän lukija, joka vastaanottaa väylän kaikki viestit ja esittää ne selkokielisenä käyttäjälle.

Viestitunniste, joka yleisesti on järjestelmän jokaisella laitteella uniikki, joka määrittää sen viestien prioriteetin väylässä. Prioriteetti määräytyy siten, että mitä pienempi numeroarvo, sitä suurempi prioriteetti sillä on väylään. Esimerkiksi laitteella, jonka viestitunniste on 20, on suurempi prioriteetti lähettää väylään viestiä kuin laitteella viestitunniste numerolla 24. (Introduction to the... 2016, 5.)

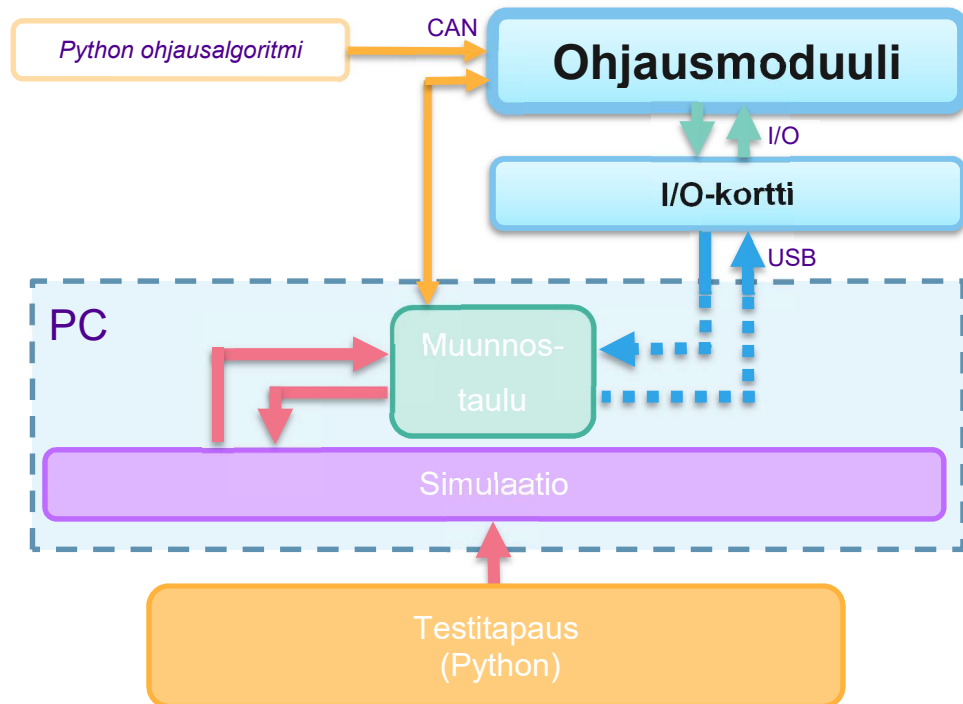
CANopen on avoin protokolla, jonka tarkoituksena on mahdollistaa yhteneväinen viestinvälitys CAN-väylässä eri laitevalmistajien välillä. Protokolla määrittää OSI-mallin mukaiset tasot 3-7. Protokolla toimii standardisoidun CAN-väylän välityksellä. (The Basics of CANopen 2019.)

Avoimen CANopen protokollan keskeisin lisäarvo CAN-väylätekniikkaan on sen objektikirjasto (Object Dictionary, OD). Se on eräänlainen taulukko, joka tallentaa toimilaitteen konfiguraatiodot sekä lähetettävän- ja vastaanotetun datan. Objektikirjaston osa tietueista ovat sellaisia, että ne ovat pakollisia toiminnankin kannalta olla lisättynä objektikirjastoon ja osa tietueista ovat valinnaisia riippuen laitteesta. Laitevalmistajalta, joka haluaa kaupata laitettaan CANopen yhteensopivana, objektikirjasto täytyy toimittaa sellaisena, että se sisältää vähintään vaaditut pakolliset tietueet. (The Basics of CANopen 2019.)



## 6 JÄRJESTELMÄN SUUNNITTELU

Järjestelmän suunnittelu aloitettiin jo hyvissä ajoin jo ennen projektin varsinaista aloitusta. Suunnittelussa hyödynnettiin laajasti aloituspalaverissa käytyjä asioita järjestelmän toiminnasta ja käyttötarkoituksesta. Suunnittelussa pyrittiin ottamaan huomioon myös se, että laitetta käytetään myös esittelyissä ja messuilla, joten sen täytyy olla helposti kuljetettava, mutta myös esteettisesti näyttävä. Suunnittelussa pyrittiin etenemään loogisessa ja selkeässä järjestyksessä, mutta usean eri osa-alueen vuoksi, eri osa-alueita jouduttiin suunnittelemaan rinnakkain ja myös tekemään jo tehtyihin suunnitelmiin muutoksia.



KUVIO 4. Järjestelmän alustavan suunnitelman lohkokaavio

Suunnittelun alkuvaiheessa tehtiin järjestelmän toiminnasta lohkokaavio (kuvio 4). Lohkokaaviosta nähdään järjestelmän toiminta hyvin yksinkertaistetulla tavalla. Lohkokaaviossa vaaleansinisellä on merkitty järjestelmän laitteisto osuus. Tähän kuuluvat TTControlin HY-TTC 580 -ohjausmoduuli, Creanex Oy:n MultiIO24s-I/O-kortti ja vaaleansinisellä katkoviivalla merkattu simulaatiosovellusta ajava tietokone. Laitteisto osuus on yhteydessä simulaatiota ajavaan tietokoneeseen USB -väylän ja CAN-väylä adapterin välityksellä. Simulaatiosovelluksessa

oleva muunnostaulu toimii rajapintana simulaation ja oikean laitteiston välillä. Se muuntaa I/O-kortin jännite- ja virtaviestit fysiikan yksiköihin, joita simulaatio voi käsitellä. Simulaatiosta päin tulevat fysiikan yksiköissä olevat viestit muunnetaan samaan tapaan laitteiston ymmärtämään muotoon. Simulaatioon voidaan myös ajaa niin kutsuttuja testitapauksia, joilla voidaan testata ja todentaa järjestelmän toimintaa. Python ohjelmointikielellä luodut testitapaukset voidaan ajaa joko simulaatiota ajavalla tietokoneella tai ulkoisesti esimerkiksi TCP/IP -yhteyden välityksellä. Lohkokaaviossa oleva keltaisella merkitty Python -ohjausalgoritmi viittaa järjestelmään ulkoisesti syötettyjä ohjauskomentoja, esimerkiksi autonomisuuden pyrkivää ajoa.

Järjestelmä suunniteltiin toteutettavaksi HiL-simulaationa, jossa oikea ohjausmoduuli ohjaa simuloitua virtuaalista traktorimallia. Järjestelmän tulisi kyetä toimimaan niin traktorimallin manuaalisessa ohjaamisessa, kuin myös vastaanottamaan ulkopuolisia ohjauskomentoja. Järjestelmään suunniteltiin myös eri turvatoimintoja, joita tutkimushankkeessa kehitetyillä työkaluilla ja menetelmillä voidaan testata. Näitä turvatoimintoja on esimerkiksi ovikytkin ja virheikäytön estäminen.

Suunnittelussa käytettiin laajasti eri suunnittelutyökaluja. Mekaanisen suunnittelun apuna käytössä oli avoimeen lähdekoodiin perustuva FreeCAD -ohjelmaa ja Autodesk Fusion 360 -3D-mallinnusohjelmaa. Sähköisen suunnittelun apuna käytettiin pääosin Autodeskin Eagle -ohjelmiston kytkentäkaavio-ominaisuutta. Edellä mainittua ohjelmaa käytettiin myös alustavan komponenttisijoittelukuvan luomiseen. Lohkokaaviokuvat tehtiin Microsoft Officen Word ja PowerPoint -ohjelmilla ja erinäiset I/O -kanavien alustavat kytkentäsuunnitelmat Excel -ohjelmalla. 3D-tulosteita varten käytettiin Dremel DigiLab 3D -slicer ohjelmaa.

## **6.1 Mekaaninen suunnittelu**

Järjestelmän suunnittelu aloitettiin mekaanisella suunnittelulla. Järjestelmä haluttiin saada kompaktiin tilaan ja kaikki järjestelmän osat (pois lukien simulaatio PC) samaan kotelointiratkaisuun. Koteloinnista haluttiin myös mahdollisesti läpinäkyvä tai edes yksi sivu läpinäkyväksi. Kotelon suunnittelu ja asennus tehtiin

yrityksessä hyviksi todettujen tapojen mukaisesti, eikä kotelointia varten tehty erityisiä lujuuslaskelmia tai komponenttien lämpenemiseen liittyviä laskelmia.

Kotelotyyppin valinnan kriteereinä oli kestävyys, kotelon läpinäkyvyys, helppo työstettävyys käsityökaluilla ja saatavuus olla hyvä. Kotelon täytyi olla myös kompaktin kokoinen, mutta otettava huomioon kaikkien komponenttien sopivuus ja ylimääräisen tilan tarve esimerkiksi käyttämättömille johdotuksille ohjausmoduulin johtosarjasta.

Kuten kappaleessa 3.1. esiteltiin HY-TTC 580 -ohjausmoduuli, sen asennusta varten huomioitavat ulkomitat ovat: 204,9 mm x 315 mm x 38,8 mm. Pituussuunnassa mitoissa on jo huomioitu ohjausmoduulin liittimien asennukseen vaadittava tilantarve. MultIO24s -liityntäkortti on ulkomitoiltaan 130 mm x 222 mm ja sen ympärille täytyy liittimien asennusta varten jättää tilaa 30-40 mm jokaisella sivulla, joten asennusta varten tarvittava tila on noin 210 mm x 302 mm.

Järjestelmän komponenttien asennuksessa oli kaksi vaihtoehtoa. Ensimmäinen vaihtoehto olisi asentaa kaikki komponentit vierekkäin kotelon sisään tarvittaviin johdotuksineen ja liittimineen. Tätä varten kotelon sisämittojen pitäisi olla vähintään noin 415 mm x 302 mm. Tämä jättäisi I/O-kortin viereen tilaa myös mahdollisille riviliittimille. Toisena vaihtoehtona oli asentaa ohjausmoduuli kotelon ulkopuolelle ja I/O-kortti kotelon sisälle, jossa tehdään myös kaikki tarvittavat kytkennät. Kotelon sisään varattaisiin tilaa sähkökytkentöjä varten riviliittimille. Tätä varten kotelon sisämittojen pituutta voitaisiin pienentää ja ohjausmoduulin liittimille ei tarvitse ottaa niiden asennusta varten tarvittavaa tilaa huomioon. Pienimmillään kotelo voisi olla tällä asennustyyllillä noin 302 mm x 280 mm. Leveys suunnassa mitaan on otettu huomioon tila mahdollisille riviliittimille (70 mm).

Ensimmäisenä vaihtoehtona kotelolle oli alumiinista tai teräksestä valmistettu 19" kehikkoasennukseen tarkoitettu, niin sanottu "räkki-kotelo". Tällaiseen koteloon olisi järjestelmän kaikki halutut osat voitu asentaa kotelon sisälle vierekkäin. Vaihtoehtona oli nVent Schroff MultiPacPRO 2U, kaksi yksikköä korkea kotelo kehikkoasennukseen (kuva 4). Kotelon kansi olisi voitu muuttaa esimerkiksi kirkkaasta muovista valmistettuun levyyn. Kotelomallista on eri syvyydellä olevia versioita,

mutta vertailuun valikoitiin suurin 340 mm syvä kotelo. Kotelon leveys on 447 mm ja korkeus 88,1 mm. (MultipacPRO, 19”... 2019.)



KUVA 4. nVent Schroff MultiPacPRO kehikkoasennettava kotelomalli (MultipacPRO, 19”... 2019)

Tämän kotelomallin hyvät ja huonot puolet:

- + Kestävä
- + Tilava
- + Hyvin saatavilla
  
- Läpinäkyvä kansi täytyisi itse tehdä
- Vaikeahko työstää käsityökaluilla
- Järjestelmää ei tulla koskaan asentamaan kehikkokaappiin

Toisena vaihtoehtona oli muovikotelo valmiilla läpinäkyvällä kannella. Muovinen kotelo olisi helposti työstettävissä ja kevyt liikutella. Vaihtoehdoksi valikoitui kotimaisen valmistajan Fibox EKPE 130 T läpinäkyvällä kannella oleva polykarbonaattikotelo (kuva 5). Kotelon sisämitat esitteen mukaan on 354 mm x 254 mm x 130 mm (Fibox EKPE 130 T 2017). Tähän koteloon asennuksen voi tehdä aikaisemmin mainitusti siten, että ohjausmoduuli asennettaisiin kotelon ulkopuolelle ja muut komponentit sisäpuolelle.



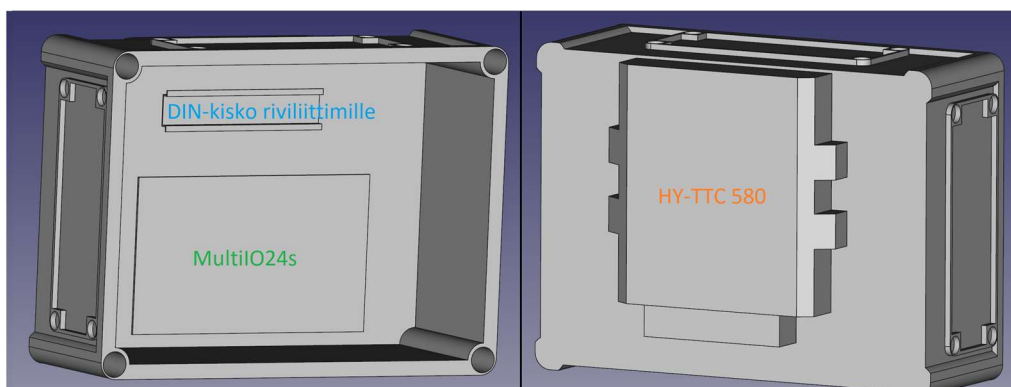
KUVA 5. Fibox EKPE 130 T (Fibox EKPE 130 T 2017)

Tämän kotelomallin hyvät ja huonot puolet:

- + Kevyt
- + Tilava
- + Hyvin saatavilla
- + Helposti työstettävä käsityökaluilla
  
- Kestävyys (verrattuna metalliseen)

Näistä vaihtoehtoista koteloksi valikoitui jälkimmäinen, eli Fibox EKPE 130 T. Valinta tehtiin kotelon näytävyyden ja helpon työstämisen vuoksi. Kun kotelotyyppi varmistui, suunnittelussa edettiin komponenttien sijoitteluun.

Alustava komponenttisijoittelukuva (liite 1) piirrettiin Autodesk Eagle -CAD ohjelmalla. Tämän alustavan kuvan perusteella piirrettiin 3D-malli FreeCAD -ohjelmalla koteloinnin syvyyden riittävyyden varmistamiseksi ja tilan tarpeen havainnollistamiseksi (kuva 6).



KUVA 6. 3D-malli kotelon alustavasta suunnitelmasta

Suunniteltuun 3D-malliin (kuva 6) ei piirretty kaapeleiden läpivientejä ajankäytön säästämiseksi koska alustavassa komponenttisijoittelukuvassa (liite 1) läpivientien paikalle oli jo määritelty riittävä tila.

Koteloon suunniteltiin yhdelle sivulle liitinpaneeli (liite 2). Liitinpaneelin suunnittelussa käytettiin Fusion 360 -ohjelmaa. Järjestelmään tarvitsi liittimet järjestelmän sähkönsyötölle ja moduulin sekä I/O-kortin tiedonsiirtoa varten. Näiden lisäksi sähkönsyöttöä varten samaan liitinpaneeliin suunniteltiin päävirtakytkin. Moduulissa on käytössä tiedonsiirtoväylinä (High Performance Safety Controller... n.d.):

- 7kpl CAN
- 1kpl Ethernet
- 1kpl LIN
- 1kpl RS232

Yllä olevista tiedonsiirtoväylistä haluttiin ottaa käyttöön Ethernet -liityntä ohjelmointia varten ja vähintään yksi CAN -väylä ulkoisten ohjauskomentojen vastaanottamiseen. Suunnitelmaan päädyttiin varaamaan paikat kahdelle CAN -väylälle, joiden liittiminä käytetään 9-napaisia D-liittimiä (D-Sub Solder Female... 2015). Ethernet -liityntää varten käytettiin Neutrikin valmistamaa NE8FDP -läpivientiliitintä (Neutrik NE8FDP n.d.). I/O-kortin tiedonsiirtoa varten tarvittiin USB -liityntä. Tätä varten käytettiin Neutrikin valmistamaa NAUSB-W-B USB A-B -läpivientiadapteriliitintä (Neutrik NAUSB-W-B n.d.). Sähkönsyöttöä varten suunniteltiin käytettäväksi sähkönsyöttömuuntajalle tarkoitettua Kycon KPJX-PM -liitintä (Kycon KPJX-PM n.d.). Päävirtakytkimeksi suunniteltiin käytettäväksi OTTO K1 ON-OFF -kytkintä (OTTO K1-Sealed Rocker n.d.).

Järjestelmään suunniteltu mahdollisuus manuaaliseen ohjaukseen ajateltiin aluksi toteuttaa USB -peliohjaimen avulla. Projektin edetessä päädyttiin kuitenkin suunnittelemaan tätä varten oma ohjain mahdollisesti koteloon kiinteästi asennettuna. Käyttjäohjainpaneeli suunniteltiin toteutettavaksi kahdella ohjainsauvalla, neljällä painonapilla ja yhdellä kolmiasentoisella vipukytkimellä. Ohjainsauvat valittiin OTTO valmistajalta heidän hall-ilmioon perustuvien ohjainsauvojen sarjasta. Ohjainsauvoja otettiin sekä yksi-, että kaksiakseliset versiot. Yksiakseliseksi ohjainsauvaksi valittiin Otto HTL2-511111AA12 (OTTO HTL2-2-Way,

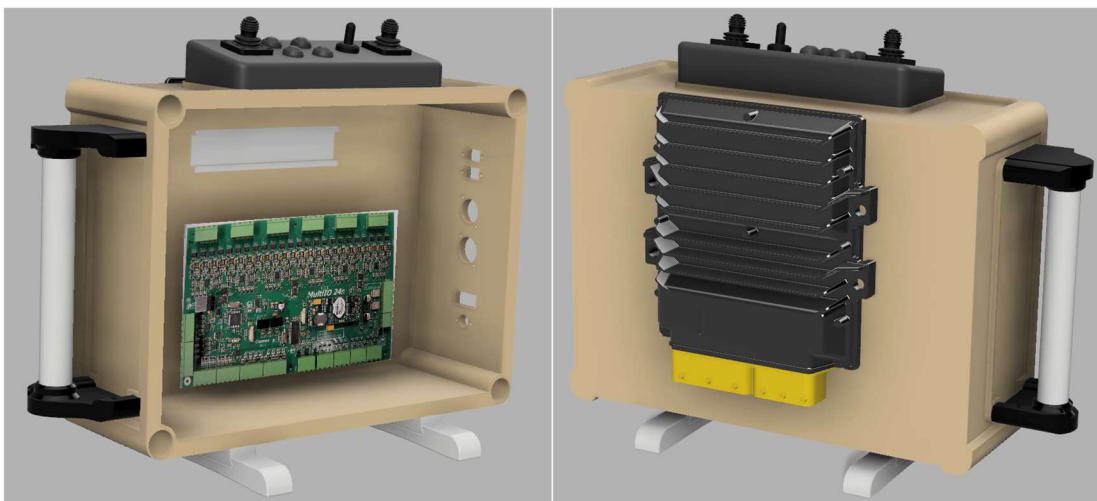
Single Axis... n.d.) ja kaksiakseliseksi Otto HTL4-511111AA12 (OTTO HTL4-4Way... n.d.). Painonapeiksi valittiin APEM IPR3FAD2L0G (Digi-Key APEM Inc. IPR3FAD2L0G n.d.), joka on normaalitilassa auki oleva, alas jäävä ja vihreällä merkkivalolla varustettu painonappi. Vipukyttimeksi valittiin kolmeen asentoon lukittuva kytkin APEM 637H/2 (Farnell 637H/2 - Toggle Switch... n.d.). Suunnitelman mukaan näiden avulla saataisiin manuaaliohjauksen tarvittavat toiminnot toteutettua.

Käyttäjäohjainpaneelia varten suunniteltiin kotelon yhdelle reunalle 3D-tulostettava kotelointi (kuva 7). Käyttäjäohjainpaneelin tekninen piirustus liitteessä 3. Mallinnus tehtiin käyttämällä Autodeskin Fusion 360 -ohjelmaa. Kotelointi mallinnettiin sopivaksi kotelossa valmiina olevan läpivientilaipan tilalle. Näin kyettiin hyödyntämään kotelon valmiita aukotuksia ja edelleen helpottamaan asennustyötä.



KUVA 7. Käyttäjäohjainpaneelin 3D-malli

Käyttäjäohjainpaneelin lisäksi koteloon suunniteltiin tukijalat, jotka voidaan helposti irrottaa kuljetusta varten (kuva 8), koska kotelo suunniteltiin pidettäväksi pystyasennossa. Kuljetusta helpottamiseksi koteloon suunniteltiin myös kantokahva (kuva 8). Molemmat näistä suunniteltiin myös kotelossa valmiina olevien aukotuksiin sopiviksi. Tukijalkojen sekä kahvan kiinnikeosan tekniset piirustukset liitteinä (liite 4 ja liite 5).



KUVA 8. Suunnitelman kokonaisuuden 3D-malli esitetty edestä ja takaa

Kuvassa 8 on esitetty suunnitelman kokonaisuus, jossa kaikki järjestelmän osat, pois lukien liittimet ja johdotukset. Mallista puuttuu myös kotelon kansi. Kokonaisuus on kasattu yhteen käyttäen Fusion 360 -ohjelmaa. Kokonaisuuteen lisätty ohjausmoduulin tarkempi 3D-malli on TTControlin tarjoama valmis 3D-malli laitteesta.

## 6.2 Sähköinen suunnittelu

Järjestelmän sähköinen suunnittelu aloitettiin tutustumalla ohjausmoduulin käyttöohjekirjaan. Käyttöohjekirjasta selviää suunnittelua varten jokaisen sisään- ja ulostulon tyyppi, sähköiset ominaisuudet ja mistä liittimen pinnistä mikäkin toiminto löytyy. Käyttöohjekirjasta löytyy myös kaikki ohjausmoduulin väyläkommunikointi ominaisuudet.

Creanex Oy:llä on ollut käytäntönä luoda projektissa oleville ohjausmoduuleille Microsoft Excel -ohjelmalla niin kutsuttu I/O-lista. Kyseinen lista tehtiin myös tässä projektissa. Esimerkki listasta näkyy taulukossa 5. Varsinainen lista sisältää laajasti informaatiota ohjausmoduulista, jota ei tämän työn yhteydessä voida julkaista. Listaan kirjataan ohjausmoduulin liityntämahdollisuudet, niin I/O-kanavat, tietoliikenneväylät kuin myös tehon syötön ja maadoituksien osalta. Tämän pohjalta voitaisiin tehdä myös automatisoidusti komentosarjojen avulla alustava tai jopa lopullinen kytkentäkaavio ohjausmoduulin ja yrityksen omien I/O-liityntä-



korttien välille (kuten esimerkiksi MultilO24s). Opinnäytetyötä varten tämä automatisoitu vaihe kuitenkin ohitettiin ja kytkentäkaavio luotiin manuaalisesti käyttäen silti apuna luotua I/O-listaa. Järjestelmän kytkentäkaavio tehtiin Autodesk Eagle -ohjelmalla. Järjestelmän kytkentäkaavio löytyy liitteestä 6.

TAULUKKO 5. Esimerkki suunnittelun apuna olevasta I/O-listasta

Liitin	Moduuli	Anturi / käyttö-tarkoitus	Pinni #	Signaalin tyyppi	Tarkennus	Simulaatio I/O	Kanava
X1	HY-TTC 580	Kaasupedaalin asento	100	PWM-HS	200 Hz	Analog In	VV-1
X1	HY-TTC 580	Jarrupedaalin asento	101	PWM-HS	200 Hz	Analog In	VV-2
X1	HY-TTC 580	Ohjauspyörän asento	102	PWM-HS	200 Hz	Analog In	VV-3
X1	HY-TTC 580	Ohjainpaneeli nappi 1	103	DO-HS	0 / 24 VDC	Digital In	DIN-1
X1	HY-TTC 580	Ohjainpaneeli nappi 2	104	DO-HS	0 / 24 VDC	Digital In	DIN-2

Mekaniikkasuunnittelu vaiheessa valittiin järjestelmään tulevia sähköisiä komponentteja. Näitä olivat eri liittimet, ohjainsauvat, painonapit ja vipukytkin. Näiden valinnat kuitenkin tehtiin ohjausmoduulin, suunnitellun komponentin ja I/O-kortin sähkötietoja sisältävien datalehtien perusteella.

Ohjausmoduuli toimii 8 – 32 VDC jännitteellä sekä I/O-liityntäkortti toimii 18 – 30 VDC jännitteellä (kappale 3.1 ja 3.2). Järjestelmään ei haluttu asentaa muuntajaa kotelon sisälle sähköturvallisuuden ja ylimääräisen lämmöntuoton takia. Näiden perusteella järjestelmän sähkönsyöttöä varten valittiin käytettäväksi ulkoista Nordic Power 24 VDC, 5 A muuntajaa. Sähkönsyöttöä varten koteloon valittiin liittimeksi muuntajan datalehden mukaisesti sille tarkoitettu 4-napainen Kycon KPJX-PM-4SS, niin kutsuttu DIN-liitin (Kycon KPJX-PM n.d.). MultilO24 I/O-korttia varten koteloon suunniteltiin USB -läpivientiliitin. Tähän valittiin käytettäväksi Neutrik NAUSB-W USB A-B -läpivientiliitintä (Neutrik NAUSB-W-B n.d.)). I/O-kortissa on USB -B-mallin naarasliitin juotettuna piirilevylle. Tästä suunniteltiin kytkettäväksi USB A-B -kaapeli kotelon sisällä Neutrikin USB A-B -läpivientiliittimeen. Ohjausmoduulin osalta liittimiä tarvitsee ohjelmointia ja CAN-väylä -kommunikointia varten. Ohjelmointia varten ohjausmoduulista otettiin käyttöön Ethernet ja tätä varten suunniteltiin käytettäväksi Neutrik NE8FDP -ethernet liitintä (Neutrik NE8FDP n.d.). CAN-väylää varten suunniteltiin käytettäväksi 9-napaista D-liitintä käytössä

olevan Kvaser USB-CAN -adapterin mukaisesti (Kvaser Leaf Light v2 User's Guide 2014, 11).

Kotelon sisään suunniteltiin myös riviliittimiä sähkönsyötön jakamista varten. Riviliittimistä suunniteltiin sähkönsyöttö jaettavaksi niin ohjausmoduulille, I/O-liityntäkortille kuin myös käyttäjäohjauspaneelin kytkimille ja merkkivaloille.

Käyttäjäohjauspaneelin ohjainsauvat toimivat 5 VDC jännitteellä (Digi-Key, APEM Inc. IPR3FAD2L0G n.d.). Näiden jännitteensyöttö suunniteltiin otettavaksi suoraan I/O-liityntäkortin sisäänrakennetun 5 VDC jännitelähdön kautta.

### 6.3 Ohjelmiston suunnittelu

Ohjelmiston suunnittelun pohjalla oli opinnäytetyön aloitus suunnitelmassa olevat määritykset järjestelmän toiminnasta. Myöhemmin järjestelmän rakennusvaiheessa määrityksiä tarkennettiin ja ominaisuuksia lisättiin. Alkuperäiset määritykset järjestelmän toimintaan sisällytettävistä toiminnoista olivat:

- Mahdollisuus manuaaliseen ajoon sekä laajennus etä- tai autonomiseen ohjaukseen
  - Manuaalisen ajon toiminnot (esimerkiksi ohjauspyörä, kaasu-/jarrupoljin) on mitattava sähköisesti, näin ollen autonominen ohjaus voi yliajaa nämä toiminnot
  - Ohjausjärjestelmä oltava suljettu säätöpiiri, eli toimintojen aiheuttamat tapahtumat on mitattava (esimerkiksi ohjauspyörän kääntö aiheuttaa renkaiden ohjauskulman muutoksen ja muutos on mitattava)
  - Autonomisessa ohjauksessa toimintoja säädetään PI-säätimellä
- Moottorin käynnistys ja sammutus
- Vakionopeuden säädin
- Turvatoiminnot:
  - Ovikytkin
  - Peräkoukun valvonta (täytyy olla täysin ylä- tai ala-asennossa)
  - Työlaitteen tai karrin kytkentä traktorin perään

Määrityksien perusteella järjestelmässä täytyi olla mahdollisuus manuaaliseen traktorin ohjaukseen ja siihen täytyi myös sisällyttää toiminto erillisten ohjauskomentojen vastaanottamista varten. Ohjausmoduulin ulkopuoliset ohjauskomennot suunniteltiin vastaanotettavaksi CAN-väylän kautta.

Manuaaliseen ohjaamiseen suunniteltiin kaksi vaihtoehtoa. Ensimmäisenä vaihtoehtona manuaalinen ajotoiminto suunniteltiin toteutettavaksi Logitech F310 USB -peliohjaimella. Toisena vaihtoehtona oli suunnitella erilliskomponenteista koottu käyttäjäohjainpaneeli traktorin haluttujen toimintojen perusteella. Valmis USB -peliohjain olisi toteutusvaiheen kannalta ollut helpompi toteuttaa, mutta erillinen käyttäjäohjainpaneeli mahdollistaa manuaalisen ajon toimintojen mittauksen ohjausmoduulilla suoraan. Suunnittelussa päädyttiin toteuttamaan manuaalinen ohjaus erillisellä käyttäjäohjainpaneelilla.

Alkuperäisessä määritelmässä ohjausjärjestelmä suunniteltiin toteutettavaksi suljettuna säätöpiirinä sekä autonomisen ohjauksen toiminnot säädettäisiin PI-säätimellä. Määritelmää tarkennettiin kuitenkin siten, että pidetään ohjausmoduulin ohjausjärjestelmä mahdollisimman yksinkertaisena ja esimerkiksi autonomisen ohjauksen toimintojen säätö tehtäisiin jo ennen niiden syöttöä järjestelmään. Näin ollen järjestelmä vain vastaanottaa käsitellyt ohjauskomennot.

Järjestelmään suunniteltiin toteutettavaksi myös simulaatioraktorin moottorin käynnistykseen ja sammutukseen sekä vakionopeudensäätimen käyttöön toiminnot. Näiden lisäksi suunniteltiin ovikytkimelle, peräkoukun valvonnalle ja työkalun kytkennälle turvatoiminnot.

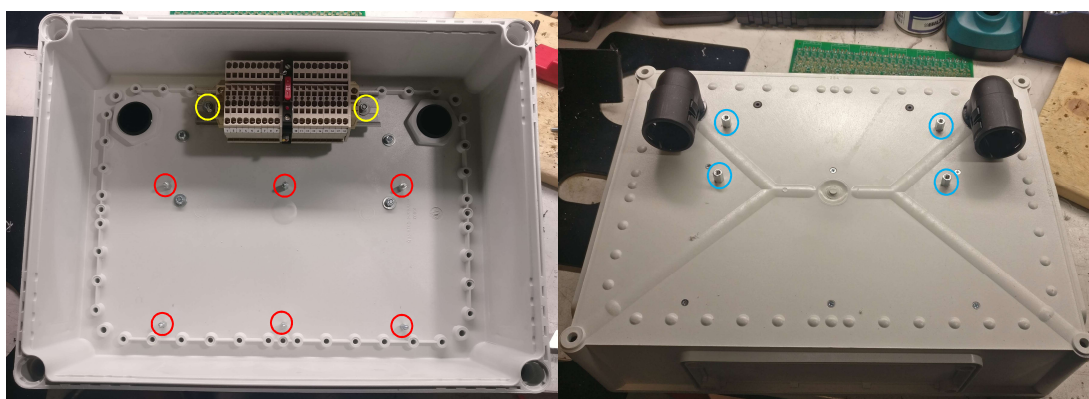
## 7 JÄRJESTELMÄN TOTEUTUS

Järjestelmän toteutusvaiheeseen kuului järjestelmän rakennus, eli mekaaninen kokoonpano ja sähköinen kytkentä. Järjestelmän rakennuksessa, niin mekaanisessa kokoonpanossa kuin myös sähköisessä kytkennöissä, käytettiin yrityksessä hyväksi todettuja käytäntöjä ja menetelmiä. Rakennettuun järjestelmään voitiin ryhtyä toteuttamaan ohjelmaa suunnitelman mukaisesti. Ohjelmointi tehtiin osa-alue kerrallaan ja toimintaa testattiin useita kertoja ennen lopullista versiota.

Järjestelmää varten tilattiin kotelo Fibox EKPE 130 T sekä muuntaja Nordic Power 24 VDC / 5 A Elfa Distrielec -elektroniikkatuotteiden jakelijalta. Järjestelmään tarvittavat muut osat löytyivät yritykseltä jo valmiina.

### 7.1 Kokoonpano ja sähköinen kytkentä

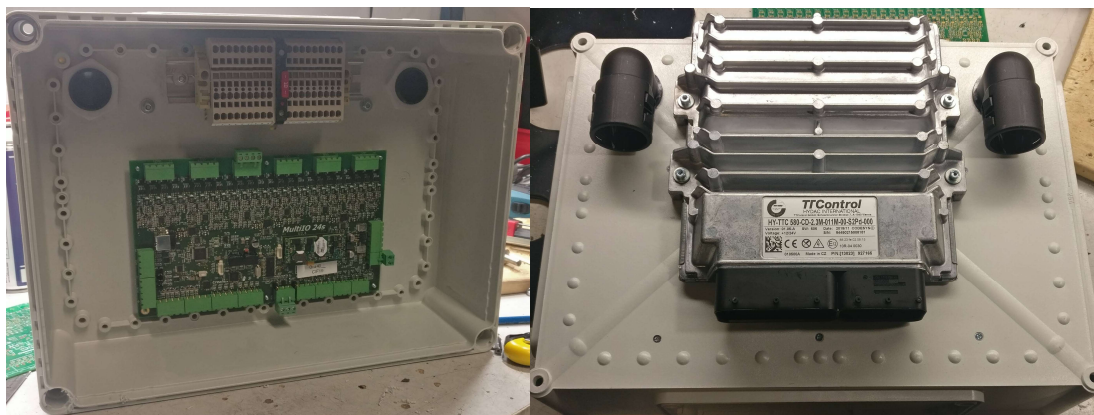
Järjestelmän kokoonpano aloitettiin luomalla kiinnityspaikat ohjausmoduulille, I/O-liityntäkortille ja DIN-kiskolle riviliittimiä varten. Näiden lisäksi ohjausmoduulin johtosarjoja varten koteloon tehtiin kaksi läpiviientiä (kuva 9). Kiinnityspaikat komponenteille tehtiin korotusruuvin avulla.



KUVA 9. Järjestelmän komponenttien kiinnityspaikat sekä läpiviennit

Kuvassa 9 punaisella on merkitty I/O-liityntäkortin kiinnityspaikka, sinisellä ohjausmoduulin kiinnityspaikka sekä keltaisella DIN-kiskon kiinnityspaikka, johon on jo asennettu DIN-kisko ja riviliittimet. I/O-liityntäkorttia varten käytettiin M3 -korotusruuveja ja ohjausmoduulia sekä DIN-kiskoa varten M4 -korotusruuveja.

Korotusruuvit kiinnitettiin kotelon sisäpuolelta muttereilla ja kotelon ulkopuolelta uppokantaisilla ruuveilla, jotta ne eivät häiritse ohjausmoduulin asennusta. Johtosarjoja varten koteloon asennettiin kaksi 90° kulmassa olevaa läpivientiä, jotka näkyvät oikealla puolella kuvassa 10, kotelon yläkulmissa. Läpiviennit asennettiin koteloon niille tarkoitetuilla muovisilla muttereilla.



KUVA 10. Järjestelmän komponentit kiinnitettynä koteloon

Järjestelmän komponentit asennettiin koteloon testatakseen, että ne mahtuvat ja esimerkiksi liittimien käyttö on vielä mahdollista (kuva 10). Moduulin johtosarja voitiin myös tässä vaiheessa syöttää läpivientien kautta kotelon sisälle ja leikata oikeaan mittaan (kuva 11).



KUVA 11. Ohjausmoduulin johtosarja asennettuna

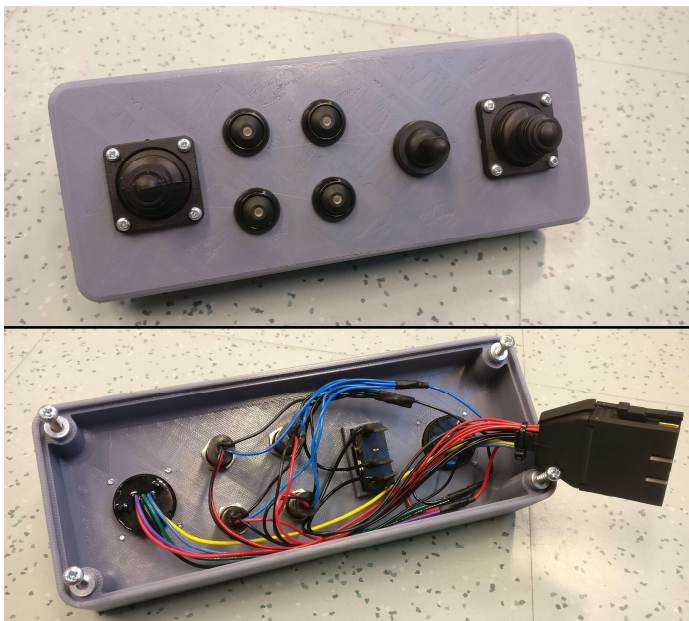


KUVA 12. Järjestelmän liitinpaneeli

Koteloon tehtiin aukotukset oikeaan kylkeen liittimiä ja kytkintä varten suunnitelman mukaisesti (kuva 12). Liitinpaneelin komponentit ylhäältä alkaen:

- 2 kpl CAN, 9-napainen naaras D-liitin
- 1 kpl Ethernet, Neutrik NE8FDP
- 1 kpl USB, Neutrik NAUSB-W
- 1 kpl Päävirtakytkin, OTTO K1
- 1 kpl Sähkönsyöttö, Kycon KPJX-PM-4SS

Aukotusta varten liitinpaneelin suunnitelma (liite 2) tulostettiin ja leikattiin sopivaksi ja tämän avulla määritettiin aukkojen tarkat paikat. Liittimet kiinnitettiin koteloon mukana tulleiden ruuvien avulla sekä käyttäen lukkiutuvia nyloc-mutteita.



KUVA 13. 3D-tulostettu käyttäjäohjainpaneeli

Käyttäjäohjainpaneeli 3D-tulostettiin suunnitelman mukaisesti (kuva 13). Käyttäjäohjainpaneeliin asennettiin suunnitellut kytkimet ja vivut. Kytkimien ja vipujen kytkentä tehtiin sähkösuunnitelman mukaisesti (liite 6, sivu 3).



KUVA 14. 3D-tulostettu kantokahva

Järjestelmän kantokahvaa varten 3D-tulostettiin kiinnityspalat ja käytettiin metallista 23 mm halkaisijaltaan olevaa putkea itse kahvana (kuva 14). Kiinnitys kote-

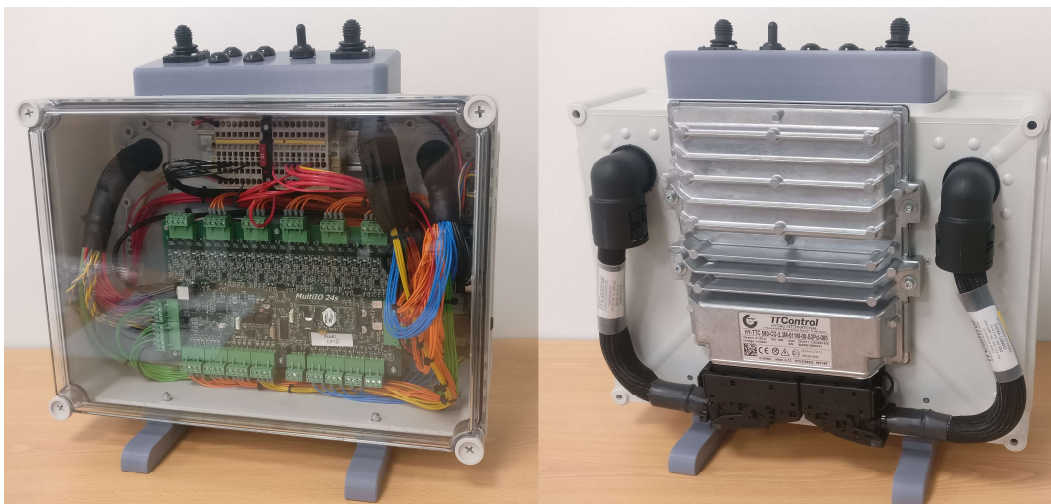


loon tehtiin kotelossa jo valmiina olevien aukkojen kohdalle M5 -ruuveilla ja muttereilla, jotka ovat upotettuna 3D-tulosteeseen. Koteloon 3D-tulostettiin myös pohjaan tukijalat, koska ohjausmoduuli siirtää kotelon painopistettä niin paljon, ettei se pysy pystyasennossa ilman tukea. Tukijalat näkyvät valmiin järjestelmän yleiskuvassa (kuva 15).

Järjestelmän sähkökytkentä tapahtui suunnitelman mukaisesti. Kytkennät tehtiin riviliittimillä sekä I/O-liityntäkortin liittimillä. Kytkennässä johtimien päihin asennettiin johdinholkit, jotka pitävät monisäikeisen johtimen säikeet yhdessä. Johdotus pyrittiin tekemään siististi kotelon reunoja pitkin siten, että kotelon sisällä oleva I/O-liityntäkortti näkyisi läpinäkyvän kannen läpi mahdollisimman hyvin.

Koska käytössä olevassa ohjausmoduulissa on suuri määrä järjestelmässä tarvitsemattomia johtimia, eikä näitä jatkokehityksen kannalta haluttu katkaista, ylimääräiset johtimet suojattiin sähköteipillä ja sijoitettiin ne I/O-liityntäkortin alle. Tämä mahdollistaa sen, että jos järjestelmää jatkokehitetään ja otetaan käyttöön ominaisuuksia, joita tähän työhön ei tarvittu, voidaan silti mahdollisesti hyödyntää samaa johtosarjaa.

Järjestelmän kokonaisuus on esitettyä kuvassa 15. Kotelon viimeistelyssä lisättiin ruuvilukitteet kaikkiin sellaisiin ruuveihin, jotka eivät olleet kiinnitettynä lukittavalla mutterilla. Ruuvien kiinnitys on varmistettava, koska järjestelmä on suunniteltu liikuteltavaksi. Järjestelmän kokonaispainoksi muodostui 5,3 kg, joten se on helposti liikuteltavissa.



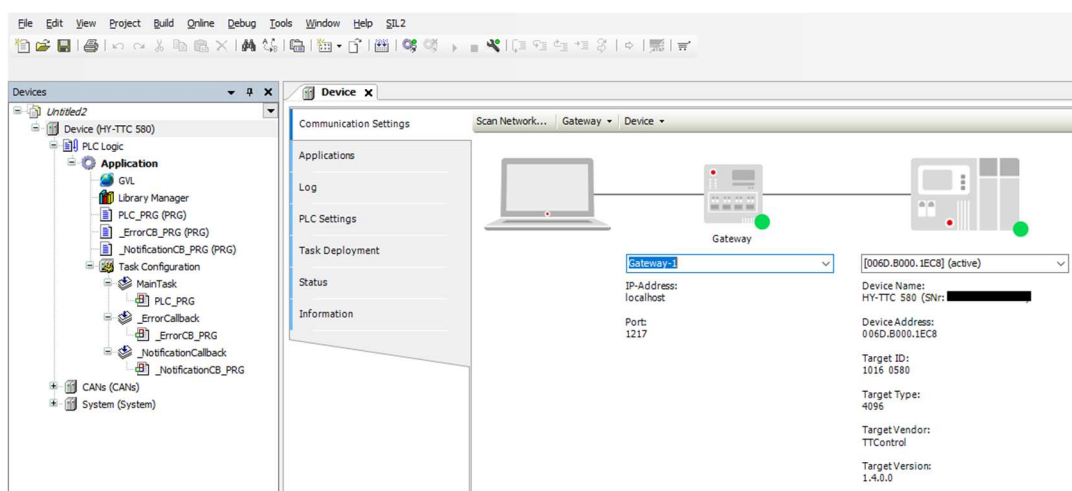
KUVA 15. Järjestelmän kokonaisuus



## 7.2 Ohjausmoduulin ohjelma ja sen ohjelmointi

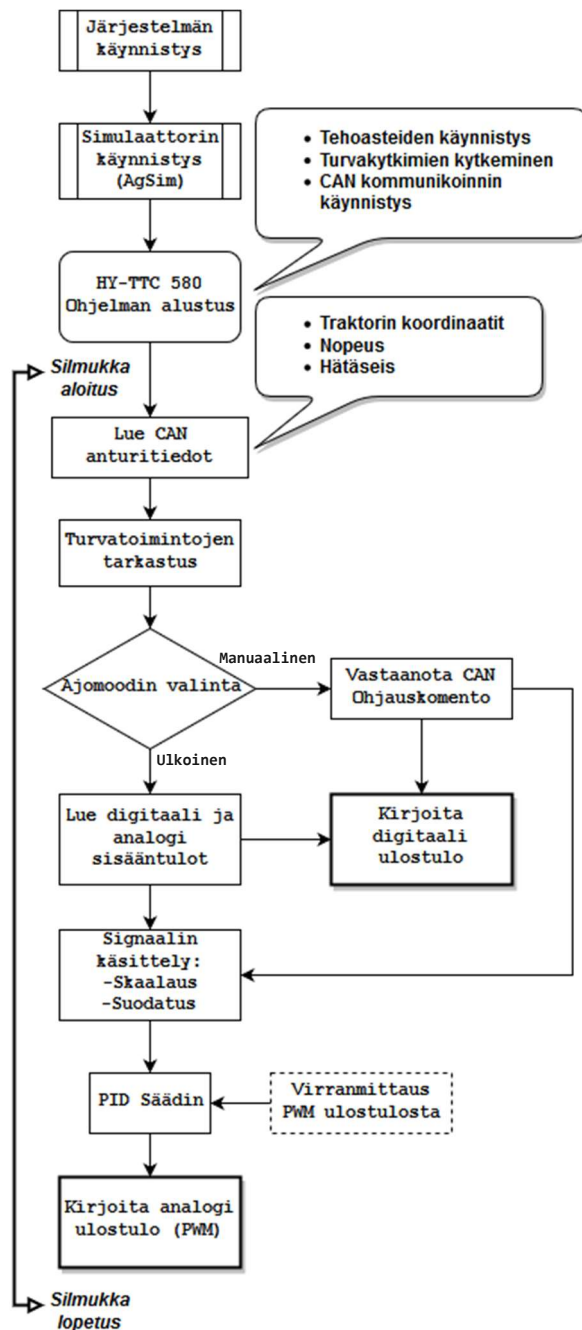
Järjestelmän ohjelmointia varten ohjausmoduuliin asennettiin TTControlin, eli valmistajan erikseen toimittama alkulatausohjelma, eli bootloader. Alkulatausohjelma käynnistää ja määrittelee ohjausmoduulin käynnistyksessä ladattavat toiminnot ja prosessit (Technopedia, Boot Loader n.d). Kun alkulatausohjelma on asennettu ohjausmoduulille, voidaan se yhdistää CoDeSys-ohjelmointiympäristöön, jossa ohjelma voidaan kirjoittaa.

Alkulatausohjelman asennus tapahtuu ohjausmoduulin valmistajan omalla ohjelmistolla, joka voidaan ladata valmistajan sivuilta. Ohjelmistosta ensin määritetään käytettävä tiedonsiirtomenetelmä CAN tai Ethernet. Tässä työssä tiedonsiirtomenetelmänä käytettiin Ethernet -liityntää. Laitteen yhdistyksen jälkeen ohjelmistosta valitaan ohjausmoduulille käytettävä alkulatausohjelma. Tämän jälkeen alkulatausohjelma voidaan ladata ohjausmoduulille ja ohjelmisto ilmoittaa tämän onnistumisesta.



KUVIO 5. Ohjausmoduulin yhdistäminen CoDeSys-ohjelmointiympäristöön

Ohjausmoduuli yhdistetään CoDeSys-ohjelmointiympäristöön käyttäen Ethernet -liityntää (kuvio 5). Yhdistämisen jälkeen CoDeSys-ohjelmistoympäristön avulla voidaan ohjausmoduuliin ladata ohjelma sekä tutkia ja testata, eli debugata sitä sen valmistus vaiheessa.



KUVIO 6. Ohjausmoduulin ohjelman lohkokaaviokuva

Kuviossa 6 on esitetty ohjausmoduulin ohjelman toiminta lohkokaavio muodossa. Kun järjestelmä käynnistetään, ohjelma jää odottamaan simulaatiosovelluksen käynnistämistä tietokoneelta. Simulaation käynnistyessä se aktivoi digitaalisen ulostulokanavan aktiiviseksi I/O-liityntäkortilta, jonka ohjausmoduuli lukee ja voi aloittaa oman alustus vaiheen.

Ohjausmoduulin ohjelma lukee sekä kirjoittaa jokaisen I/O-kanavan arvon globaali muuttujalistaan (Global variable list) (liite 7). Muuttujat voidaan kyseisestä listasta vastaamaan järjestelmän I/O-kanavien tilaa.

Ohjelman alustuksessa (liite 8) ohjelma tarkastaa onko alustusta jo tehty (rivi 1). Tämän jälkeen tarkistetaan edellä mainittu simulaattorista tulevan digitaalisen ulostulon tila (rivi 2). Jos tämä on aktiivinen, niin voidaan käynnistää ohjausmoduulin tehoaste I/O-kanaville (rivi 3), kytkeä tehoasteiden turvakytkimet (rivit 4-6) sekä käynnistää CAN-väylä -kommunikaatio (rivit 7-12). Lopuksi muutetaan alustuksen tila (rivi 13), ettei ohjelma tämän jälkeen palaa alustus vaiheeseen.

CAN-väylän anturitiedot luetaan (liite 9) ohjelman alustusvaiheen jälkeen ja silmukan jokaisella kierroksella. Ohjelma määrittää CAN-vastaanottokanavan, CAN-ID -tunnistenumeron, aikakatkaisun sekä tiedon vastaanottoon liittyvät arvot (rivit 2-11). Tämän jälkeen ohjelma tarkastaa onko uusia CAN-viestejä tullut edellä määriteltyn vastaanottokanavaan (rivi 13). Jos uusia viestejä on saapunut, ohjelma yhdistää viestissä olevat tavut (byte) sanoiksi (word) ja tallentaa kyseisen tiedon muuttujaan (rivit 14-15). Tämä toistetaan jokaisen halutun tavun kohdalla. Jos CAN-viestissä tapahtuu aikakatkaisu, niin ohjelma asettaa edellä määritetyt muuttujat arvoon 0. Digitaaliset, eli yhden bitin mittaiset viestit puretaan määritellystä tavusta ja puretut bitit voidaan tallentaa muuttujiin (rivit 30-62).

Seuraavassa vaiheessa ohjelma tarkastelee ohjausmoduulin turvatoimintojen tilan, ajomoodin valinnan tilan sekä lukee manuaalisen ja ulkoisen ohjauksen arvot (liite 10). Turvatoimintoihin liittyen ohjelma tarkastaa onko jokin turvatoiminto aktiivinen (rivi 1). Jos turvatoiminto on aktiivinen, koko ohjausarvojen luku ohitetaan ja ohjelma siirtyy turvatoiminnot määrittelevään ohjelmalohkoon (liite 11). Turvatoiminnoissa otettiin käyttöön traktorin virhekäyttöön liittyvä ominaisuus (rivit 1-3), joka tarkastaa onko kaasupoljin sekä jarrupoljin yhtä aikaa aktiivinen. Jos ehto toteutuu ohjelma asettaa kaasupolkimen ohjauksen pois (rivi 2). Ovikytkin (rivit 5-15), joka on järjestelmän käyttäjäohjainpaneelissa erillinen kytkin. Kytkimellä voidaan simuloida tilannetta, jossa traktorin ovi on auki tilassa. Jos ovi on auki, ohjelma asettaa kaasupolkimen ohjauksen pois (rivi 7), jarrupolkimen sekä käsijarrun ohjauksen päälle (rivit 8-9) sekä asettaa traktorin vaihdelaatikon asentoon P (rivit 10-12). Peräkoukun asennon tunnistaja (17-20) saa simulaatiomallista tiedon onko peräkoukun asento täysin ylä- tai ala-asennossa. Jos koukku on ylä- tai ala-asennon välissä, turvatoiminto asettaa kaasupolkimen ohjauksen pois ja ohjaa jarrupolkimen päälle.

Ajomoodiksi voidaan valita joko manuaalinen tai ulkonen-/autonominen ajomoodi. Ajomoodi määritellään CNTRLS\_PRG -ohjelmalohkossa liitteessä 10 rivillä 3. Jos ulkoinen/autonominen ohjaus ei ole aktiivisena, ohjelma lukee käyttäjäohjainpaneelin kytkimien ja ohjainsauvojen tilat (rivit 5-102). Esimerkkinä digitaalisten kytkimien lukeminen tapahtuu riveillä 6-9. Ohjelma lukee globaali muuttuja listaan tallennetun kytkimen asennon tilan ja kirjoittaa sen ulostuloksi määritelyyn globaaliin muuttujaan. Ohjainsauvojen tila ja sen ohjaus luetaan suoraan PID -funktioon (liite 12). Esimerkkinä vasemman puoleisen ohjainsauvan vaakasuoran akselin liike, jolla ohjataan simulaatioraktorin ohjauspyörän asentoa (rivit 20-28). PID -funktioille luetaan asetuspisteeksi ohjainsauvan tila (rivi 20) sekä ulostuloksi asetetun PWM -ulostulon virran arvo takaisinkytkentä arvoksi (rivi 21). Asetuspisteeksi luettu arvo sekä ulostuloksi kirjoitettu arvo skaalataan samalla funktiolla (rivit 22-27). PID -funktio laskee näiden arvojen, sekä ohjainmoduulille vakiona asetettujen P-, I-, D-arvojen avulla säädetyn PWM -ulostulon arvon, joka kirjoitetaan globaaliin muuttujaan datatyyppin muunnoksen jälkeen (rivi 20).

Ulkosen/autonomisen ohjauksen ollessa aktiivinen, ohjelma lukee ohjaukset CAN-väylän kautta. Esimerkkinä CAN-väylän kautta kaasupolkimen ohjaukseen on liitteessä 10 riveillä 107-114. Ohjaus tapahtuu kuten manuaalisen ohjauksenkin tapauksessa PID -funktion avulla. Ainoana erona on asetuspisteeksi luettu arvo, joka tulee globaalin muuttujalistaan tallennetusta CAN-väylästä otetusta arvosta (rivi 107) sekä skaalauksen arvojen osalta (rivit 109-114).

### 7.3 Simulaattorin konfigurointi

Simulaatiota varten täytyy tehdä erinäisiä konfiguraatietiedostojen muokkauksia ja lisäyksiä, jotta järjestelmä saadaan toimimaan simulaatiosovelluksen kanssa. Simulaatioon luodaan myös virtuaalinen CAN-väylä ja sitä varten CANopen -toimilaite, jonka avulla simuloidut CAN-väylän kautta toimivat toimilaitteet ja anturit voivat kommunikoida.

Konfiguraatietiedostot ovat niin kutsuttuja ini-tiedostoja. Ne ovat tiedostoja, joita voidaan käyttää yleisesti sisältämään ohjelman konfiguraatio ja asetustietoja (FileInfo, INI file extension 2019). Simulaatiosovelluksessa ini-tiedostoihin asetetaan esimerkiksi I/O-liityntäkortin toiminnallisuus, simulaatiosignaalien erinäiset reititykset, I/O-liityntäkortin kanavien sekä simulaatiomallin ohjaukseen liittyvät yhdistykset. Virtuaalista CANopen -nodea varten tehdään kappaleessa 5 esitelty objektikirjasto sekä ini-tiedosto, jossa toiminnallisuus otetaan käyttöön. I/O-liityntäkortti konfiguroidaan ja otetaan käyttöön *IOConfig.ini* tiedostossa.

```
[IOCard]
*IOCard = MultiIO24_1
```

Tiedoston ensimmäisessä kohdassa määritellään käytetyt I/O-liityntäkortit *[IOCard]*. Tämän alle määritellään käytössä oleva liityntäkortti, joka tässä työssä on MultiIO24. Liityntäkortin nimen lopussa oleva numero 1 on kortin identifioiva numero, eli jos samanlaisia kortteja järjestelmässä on useampi, tähän voidaan merkitä numeroita kasvavassa järjestyksessä.

```
[MultiIO24_1]
Type = CRNX_USB_MULTI_IO_24S
ProductID = 0xA400
UseRealHW = 1
DAC16Bit = 1 ; 12/16bit default 12bit
DAC10Volts = 1 ; 5 / 10 Volts default 5V
EnableAllVirtualValves = 1
ResetOutputsOnClose = 1
DOUT-1 = 1 ; used for the HY-TTC580 initialization
```

Tässä määritetään lisätyn liityntäkortin *[MultiIO24\_1]* ominaisuudet. Ensimmäisenä määritetään liityntäkortin kokonainen nimi sekä kortin ID-numero kohdissa *Type* ja *ProductID*. Simulaattori tunnistaa kortin näiden avulla USB-väylästä. Kohta *UseRealHW* asettaa liityntäkortin käyttöön. Seuraavana kohdissa

*DAC16Bit* ja *DAC10Volts* määritellään digitaali-/analogimuuntimen (DAC) käytetty tarkkuus sekä jännitetaso. Tässä työssä käytettiin 16-bittistä tarkkuutta sekä 10 V jännitetasoa. Kaikki PWM-sisääntulot otetaan käyttöön kohdassa *EnableAllVirtualValves* ja kortti asetetaan sammuttamaan kaikki ulostulot simulaation sammuttamisen yhteydessä kohdassa *ResetOutputsOnClose*. Kohdassa *DOUT-1* asetetaan liityntäkortin digitaalinen ulostulo järjestysnumerolla 1 jatkuvasti päällä olevaan tilaan. Tätä käytetään ohjausmoduulin ohjelmassa alustuskomentona.

```
[IOTable]
*AddChannel = MultiIO24_1 VV-1 | HandBrake | C_Joy1_Y- | ControlPanel Left Joystick
*AddChannel = MultiIO24_1 DIN-1 | Hitch | C_Btn_1 | ControlPanel Button 1
*AddChannel = MultiIO24_1 DOUT-1 | InitializePLC | INIT | Always on, initializes ECU
*AddChannel = MultiIO24_1 DOUT-5 | IndicatorLED 1 | C_IND_1 | Button 1 LED
```

I/O-liityntäkortin sisään- ja ulostulot otetaan käyttöön tiedoston kohdassa *[IOTable]*. Työ sisältää suuren määrän sisään- ja ulostuloja, joten yllä olevassa esimerkissä esitetään vain yksi jokaista käytettyä erityyppistä sisään- tai ulostuloa. Esimerkissä otetaan käyttöön PWM-sisääntulo *VV-1*, digitaalinen sisääntulo *DIN-1* sekä digitaaliset ulostulot *DOUT-1* ja *DOUT-5*. Ensimmäisenä kanavaa varten annetaan tieto kortin nimestä ja halutusta kanavasta. Pystyviivalla eroteltuna ensimmäiseksi annetaan kanavalle selkokielineen nimi, seuraavaksi ulkoiseen ohjaukseen tai simulaattorin sisällä käytetty muuttujanimi ja viimeiseksi kanavan lisätieto, esimerkiksi mikä ohjain tai kytkin on kyseessä.

Osa I/O-kanavista ja simulaatiosignaaleista täytyy reitittää simulaation sisällä. Reitityksiä tehdään kahdella eri tiedostolla: *IORouting.ini*, joka vastaa I/O-korttien kanavien reitityksestä sekä *SimRouting.ini*, joka vastaa simulaatiosignaalien reitityksestä.

```
[IORouting]
*Route = C_Btn_1 | C_IND_1
*Route = C_Btn_2 | C_IND_2
*Route = C_Btn_3 | C_IND_3
*Route = C_Btn_4 | C_IND_4
```

I/O-kanavien reititys tehdään *IORouting.ini* tiedoston kohdassa *[IORouting]*. Reititykset tehtiin I/O-kanaville annettuiden muuttujanimien avulla. Reititys tehtiin järjestelmän käyttäjäohjainpaneelin kytkimien merkkivaloja varten. Tässä ohjaus-

moduulilta tuleva kytkimen asentotieto I/O-liityntäkortille muuttuun *C\_Btn\_1* tallennettu yhdistetään digitaalisen ulostulon muuttuun *C\_IND\_1*, joka on kytketty vastaavan kytkimen merkkivaloon.

```
[IORouting]
*Route = FarmingTractorChannels AO 1 | TractorSignalsNode X_Coordinate
*Route = FarmingTractorChannels AO 2 | TractorSignalsNode Y_Coordinate
*Route = FarmingTractorChannels AO 4 | TractorSignalsNode Heading
*Route = SIO_FarmingTractor AO 1 | TractorSignalsNode Speed
*Route = SIO_FarmingTractor OUT_HitchEndStop | TractorSignalsNode HitchEnd
```

Simulaatiosignaalien reititys tehdään tiedostolla *SimRouting.ini* kohdassa *[IORouting]*. Tällä reititetään traktorin sijaintitiedot simulaatiomaailmasta, nopeus sekä peräkoukun ja rajakytkimen toiminta CANopen -noden objektiikirjastoon.

```
[IN_GasPedal]
Template = T_IN_JS_ANALOG_TRACTOR
SignalChannel = MultiIO24_1 VV-5
[IN_BrakePedal]
Template = T_IN_JS_ANALOG_TRACTOR
SignalChannel = MultiIO24_1 VV-4
[IN_SteeringWheelAngle]
Template = T_IN_JS_ANALOG_TRACTOR
SignalChannel = MultiIO24_1 VV-3
SignalModel = Integrator 1.0 -60 60 DynModel 0.05
UILimits = -100.0 100.0
X = 70 1900 2100 4075
Y = 0.9 0 0 -0.9
```

I/O-kanavista saadut signaalit voidaan yhdistää suoraan simulaatiosignaaleihin *SignalMap.ini* -tiedostossa. Tässä esimerkkinä *[IN\_GasPedal]* joka vastaa simulaatiosignaalin kaasupolkimen asentoa. Tässä siihen yhdistetään MultiIO24s -liityntäkortin kanava VV-5, joka vastaanottaa PWM -signaalin ohjausmoduulilta. Se määritetään lukemaan *T\_IN\_JS\_ANALOG\_TRACTOR* -mallia (Template), joka on esitelty alla. Jarrupoljin konfiguroidaan samaan tapaan kuin kaasupoljinkin. Ohjauspyörä *[IN\_SteeringWheelAngle]* yhdistetään ottamaan signaalin MultiIO24s kanavasta VV-3. Sille asetetaan myös integroiva toiminta (*Integrator 1.0*) ja lisätään siihen dynaamista viivettä (*DynModel 0.05*) oikean traktorin toimintaa vastaavasti. Ohjainsauvan arvo, joka otetaan MultiIO24s VV-3 kanavasta, skaalataan X ja Y-muuttujilla siten, että arvoilla 70 – 1900 ohjauspyörän kulman muutosnopeus on välillä 0,0 – 0,9 °/näyte ja arvoilla 2100 – 4075 kulman muutosnopeus on välillä -0,9 – 0,0 °/näyte. Simulaattori ottaa näytteitä 10 ms välein, joten suurin muutosnopeus ohjauspyörällä on -90–90 °/s.

```
[T_IN_JS_ANALOG_TRACTOR]
MappingType = Analog
UILimits = 0 100.0
UIResolution = 0.1
X = 0 75 4030
Y = 0 0 100
```

Konfiguraatiotiedostoon voidaan luoda malleja, joita voidaan käyttää useammalla eri signaalilla. Yllä esimerkki, jota käytetään määrittelemään signaalityypiksi analogisen signaalin (*MappingType*), simulaatiosignaalin raja-arvot sekä tarkkuuden (*UILimits & UIResolution*) ja viimeisillä *X* ja *Y* -arvoilla skaalataan signaali haluttuihin arvoihin. Tämä esimerkki skaalaa sisään tulevan arvon siten, että alle 75 arvolla luetut arvot ovat aina 0. Kun sisään tuleva signaali ylittää arvon 75 niin se kasvaa lineaarisesti 0 – 100 välissä aina signaalin arvon saavuttaessa 4030 arvon.

```
[CanConnection]
*CanConnection = CANBackBone
[CANBackBone]
BoardName = KVASER
SubType = 72 ; Leaf
BoardID = 1
Port = 1
Baudrate = 500
Mode = Standard
UseRealHW = 1
```

CAN-väylän toiminnallisuus otetaan käyttöön *CanConfig.ini* -tiedostossa. Kohdassa *CanConnection* määritetään haluttu nimi CAN-solmulle. Tämän jälkeen kyseinen solmu konfiguroidaan vastaamaan käytettyä CAN -adapteria. Tässä työssä on käytössä Kvaser nimiseltä valmistajalta Leaf Light HS v2, jonka tyyppinumero on 72. Nämä määritetään kohdissa *BoardName* ja *SubType*. Jos simulaatio tietokoneessa on useampia saman mallin adaptereita, *BoardID* -kohdassa voidaan määrittää mikä adapteri halutaan käyttöön. *Port* -arvolla määritetään käytetty porttinumero kyseisessä adapterissa. Laitteen tiedonsiirtonopeus määritetään *Baudrate* -arvolla. Tässä työssä adapteri konfiguroidaan käyttöön nopeudella 500 kbit/s, koska ohjausmoduuli toimii myös kyseisellä nopeudella. Konfigurointi arvo *Mode* määrittää käytetyn protokollan version. Tässä työssä käytetään normaalia 11 -bittiä pitkiä viestitunnisteita CAN -viesteille, joten tähän annetaan arvo *Standard*.

```
[CanNodes]
*CANNode = TractorSignalsNode
[TractorSignalsNode]
CanConnection = CANBackBone
DCF = FarmingTractor\OD-TractorSignals.dcf
NodeID = 2
AutoOperational = 1
```

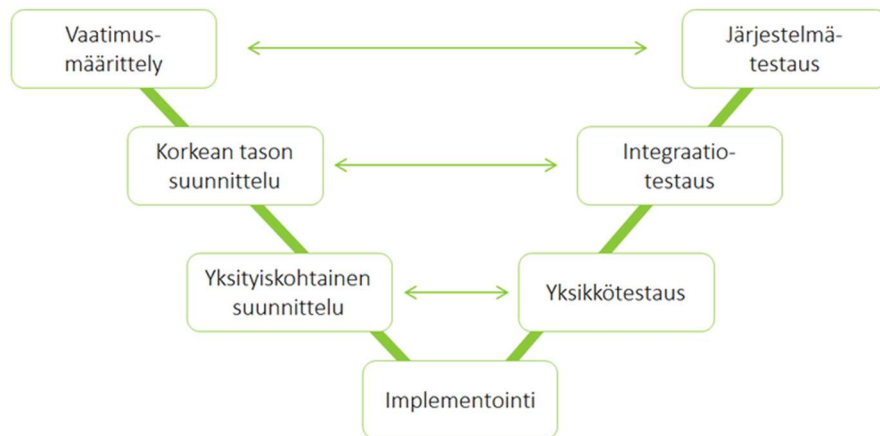


CANopen -solmu, eli node otetaan käyttöön konfiguraatitiedoston [*CanNodes*] kohdassa. Tässä määritellään haluttu nimi CANopen -solmulle, joka tähän työhön määritettiin *TractorSignalsNode*. Tälle solmulle määritetään mistä solmu lukee CAN -viestit (*CanConnection*), mitä konfiguraatitiedostoa se käyttää (*DCF*) sekä sen käytetty solmunumero (*NodeID*). *AutoOperational* -arvo määrittää solmun käynnistymään simulaation käynnistyessä.

Simulaatiosovellus luo edellä mainitun DCF-tiedoston perusteella CANopen -solmua varten tarvittun objektikirjaston. DCF-tiedosto sisältää esimerkiksi solmun konfiguraatioparametrit, solmun asetukset, jokaisen objektikirjaston merkinnän vakioarvot sekä pienimmät ja suurimmat arvot ja mitä objektikirjaston merkintöjä solmussa käytetään (Device Configuration Files n.d.).

## 8 JÄRJESTELMÄN TESTAUS

Järjestelmiä testataan, jotta voidaan varmistaa ja todentaa järjestelmän oikea ja haluttu toimintatapa (ISTQB Glossary. n.d.). Järjestelmän testaamisella tarkoitetaan järjestelmän kokonaisuuden testaamista sellaisessa ympäristössä kuin se on suunniteltu käytettäväksi (Kankaanranta & Mäkelä. n.d.).



KUVIO 7. Testaamisen V-malli (Kankaanranta & Mäkelä. n.d.)

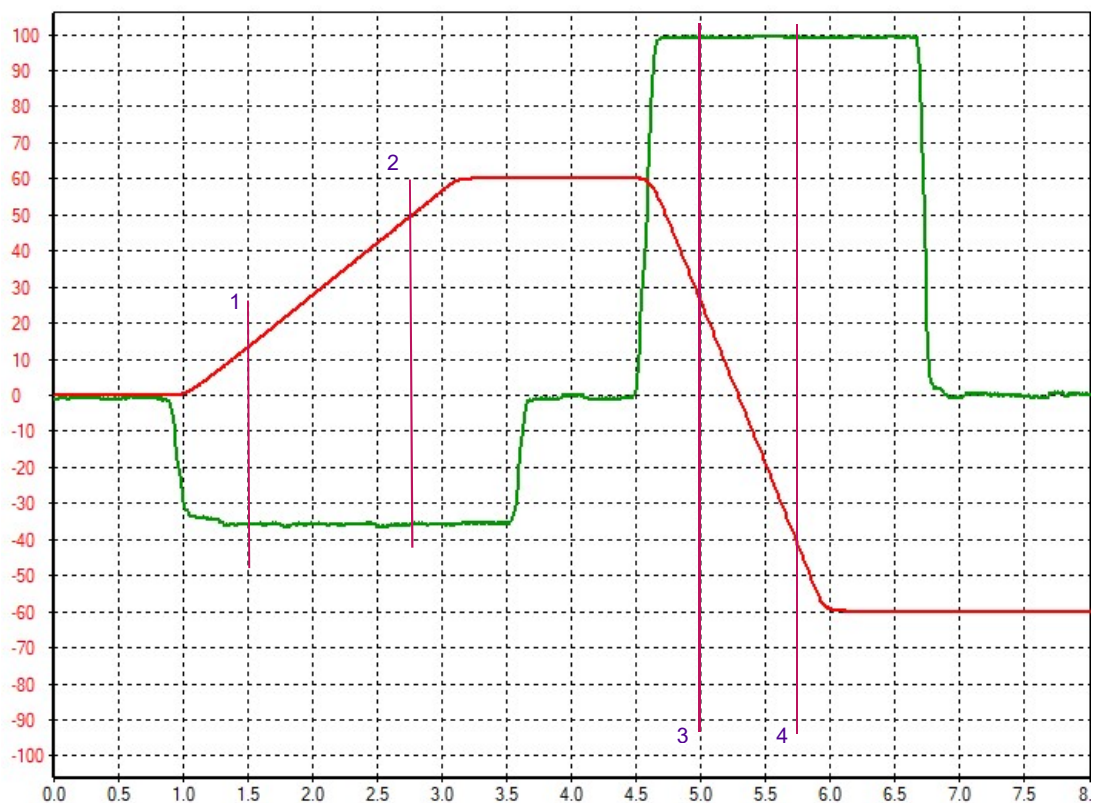
Vanha, jo 90-luvulla ohjelmistotestausta varten kehitetty, mutta osittain vielä käytetty V-malli on esitetty kuviossa 7. Se havainnollistaa suunnittelun, toteutuksen ja testivaiheiden suhdetta toisiinsa (Kankaanranta & Mäkelä. n.d.). Tässä luvussa keskitytään mallin ylimmällä tasolla olevaan ”Vaatimusten määrittely – Järjestelmätestaus”. Alempien tasojen testauksia tehtiin työn edetessä, mutta näitä ei julkisesti dokumentoida.

Järjestelmän testaamiseen käytettiin sekä simulaatiosovelluksen ominaisuuksia että yrityksen aikaisemmin tutkimushankkeen puitteissa kehitettyä TestRunner -ohjelmaa. TestRunner -ohjelma yhdistyy simulaatiosovellukseen palveluna, joka voi lukea sekä kirjoittaa simulaation signaaleja sekä I/O-kanavien arvoja. TestRunner -ohjelmalle testaamista varten kirjoitetut sovellukset, niin kutsutut testitapaukset, voidaan kirjoittaa käyttäen Python -ohjelmointikieltä. TestRunner -ohjelmaa varten yritys on kehittänyt ja luonut luokkia (class) Python ohjelmointia varten, jotka mahdollistavat esimerkiksi simulaatiosignaalien sekä I/O-kanavien

käytön. Simulaatiosovelluksella on mahdollista tarkkailla laajasti simulaatiosignaalien, I/O-kanavien sekä väyläliikenteen arvoja. Näistä voidaan tulostaa myös kuvaajat ajan suhteen.

### 8.1 Manuaaliohjauksen testaus

Manuaaliohjauksen testaamisessa käytettiin simulaatiosovelluksen työkaluja arvojen tarkkailuun ja ohjaukset tehtiin järjestelmän käyttäjäohjainpaneelin avulla. Tuloksista tehtiin simulaatiosovelluksen työkalulla kuvaajat, joita käydään läpi tässä kappaleessa. Testaukset tehtiin kaikille manuaaliohjaamiseen tarkoitettuille ohjauksille, mutta tässä dokumentissa käydään läpi vain oleelliset toiminnot testien toistavuuden luonteen vuoksi.

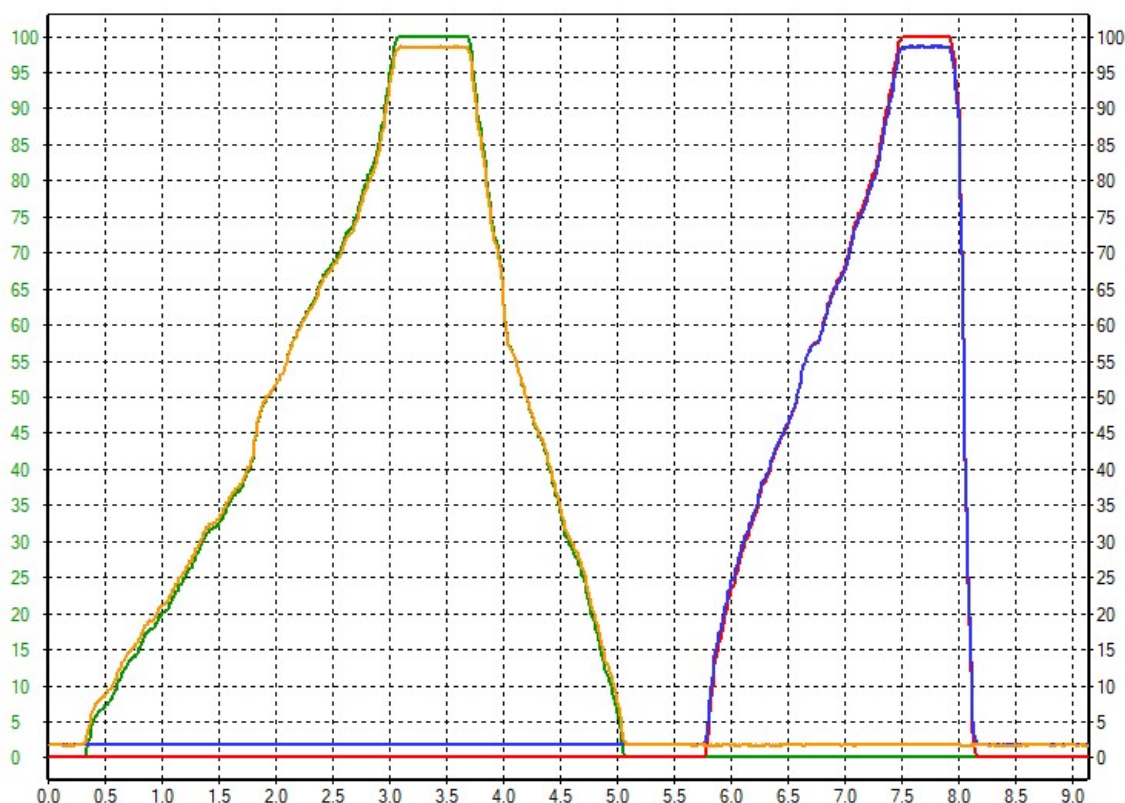


KUVIO 8. Testi 1: Simulaatiomallin ohjauspyörän asennon muutos verrattuna ohjainsauvan arvoon

Ensimmäisessä testissä testattiin ohjauspyörän toimintaa. Ohjauspyörän kulma pitäisi muuttua ajan suhteen nopeammin mitä suurempi ohjausarvo ohjainsauvalta annetaan (integraiva toiminta). Kuvion 8 pystyakseli kuvaa kulman arvoa

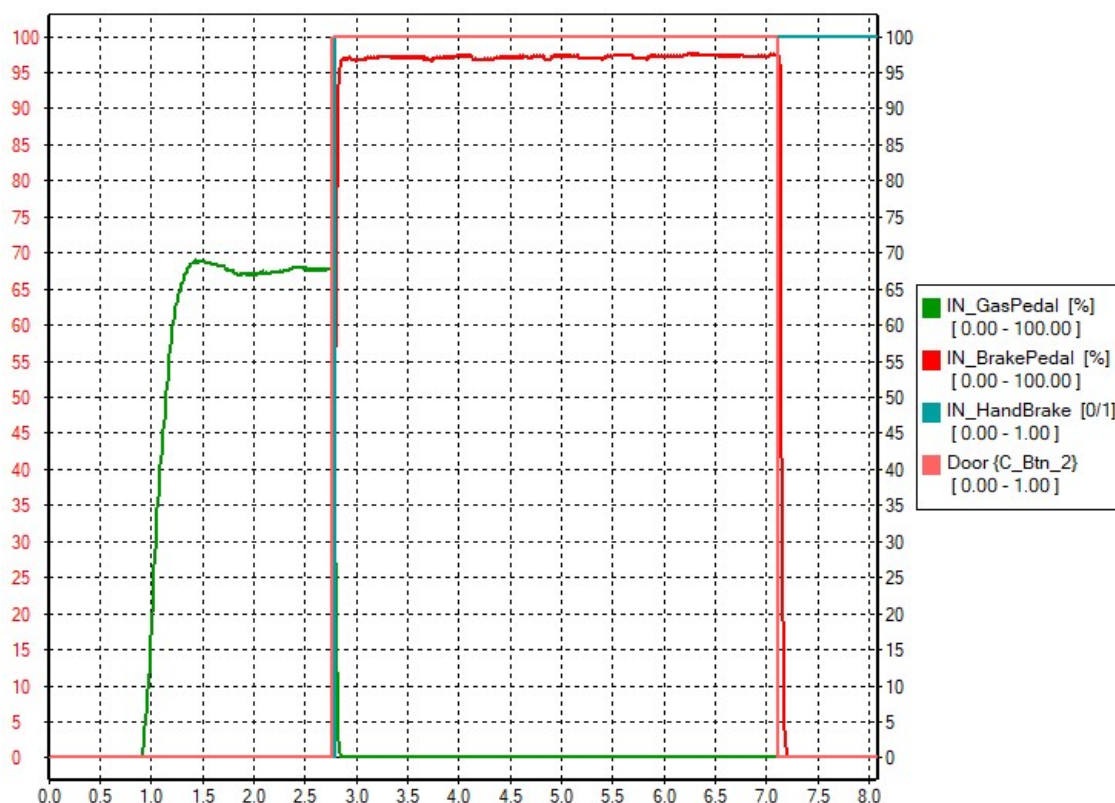
asteena sekä prosentuaalista ohjainsauvan arvoa. Vaaka-akseli esittää kuluneen ajan. Kuvaajan keskellä vaakatasossa molemmat arvot ovat 0. Punainen kuvaaja esittää simulaatiomallin ohjauspyörän asentoa ja vihreä kuvaaja ohjainsauvan arvoa. Tärkeimpänä havaintona voidaan kuvaajasta tehdä vertaamalla arvoja välillä 1-2 sekä 3-4. Kuvaajan kohdalla 1-2 ohjainsauvalla ohjataan noin 1300 yksikköä ohjausarvoa, joka vastaa noin 32 % ohjausta ja kohdalla 3-4 noin 4075 yksikköä, joka vastaa täyttä 100 % ohjausta. Simulaattorin konfiguraation mukaan ohjauspyörän muutosnopeus 100 % ohjauksella pitäisi olla 90,0 °/s ja näin ollen 32 % ohjauksella 28,8 °/s. Ohjauspyörän kulma ajan hetkellä 1,5 s (kohta 1) on 13,22 ° ja ajan hetkellä 2,75 s (kohta 2) kulma on 49,33 °. Näiden avulla voidaan laskea käyttämällä kaavaa 1 (Khan Academy n.d.) ohjauspyörän muutosnopeuden kulmakerroin. Kaavalla saadaan kulmakertoimeksi 28,89 °/s. Vastaavat arvot kohdassa 3 on 5,0 s ja 26,28 ° ja kohdassa 4 nämä ovat 5,75 s ja -41,19 °. Kohtien 3-4 aikana kulmakerroin on kaavan 1 mukaan 89,97 °/s. Saadut arvot vastaavat sitä, mitä simulaattorin konfiguraatioon on määritetty sekä muutosnopeus kasvaa ohjauksen arvon kasvaessa. Tämän mukaan voidaan todeta, että järjestelmän ohjauspyörä toimii halutulla tavalla. Muutosnopeuden suhteellista suuruutta voidaan muuttaa simulaattorin konfiguraatiotiedostoista.

$$k = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1} \quad (1)$$



KUVIO 9. Testi 2: Simulaatiomallin kaasu- ja jarrupolkimien arvon muutos verrattuna ohjainsauvan arvoon

Kuvio 9 esittää testiä, jossa testattiin kaasu- ja jarrupolkimien toimintaa. Testissä käytettiin manuaaliohjaukseen tarkoitettua ohjainsauvaa ääriasennoissaan, jotka vastaavat kaasu- sekä jarrupolkimia. Kuvaajassa simulaatiomallin kaasupolkimien signaalin arvoa esittää vihreä kuvaaja sekä ohjainsauvan arvoa keltainen kuvaaja. Simulaatiomallin jarrupolkimien kuvaaja on esitetty sinisellä sekä sitä vastaava ohjainsauvan arvo punaisella. Molempien kohdalla simulaatiomallin signaalin pitäisi vastata ohjainsauvan arvoa samassa suhteessa. Ohjainsauvan syöttämä arvo on kuitenkin simulaattorin konfiguraatiossa skaalattu siten, että säätöalue tapahtuu arvovälillä 75 – 4030 ja nämä vastaavat simulaatiosignaalin arvoja 0 – 100 %. Ohjaussignaalin arvoa tutkimalla kuvaajasta tämä havaitaan siten, että ohjausta näyttäisi olevan jatkuvasti noin 2 % ja täydellä ohjauksella se jäisi vain noin 98 % kohdalle. Molempia pedaaleita ohjataan samalla ohjainsauvalla mutta vastakkaisiin suuntiin. Ohjausmoduulin ohjelma muuttaa ohjainsauvan signaalin kahdeksi eri ulostuloksi, jotka ovat kaasu- ja jarrupoljin. Tämän testin mukaan kaasu- sekä jarrupolkimien ohjaus toimii halutulla tavalla.



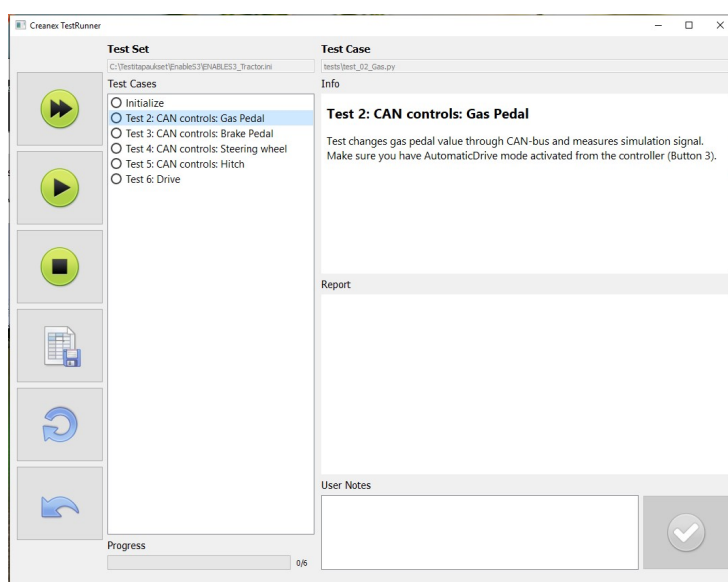
KUVIO 10. Testi 3: Ovikytkimen toiminta

Kolmannessa testissä testattiin ovikytkimen toiminta. Kun järjestelmä havaitsee ovikytkimen tilan muutoksen se pysäyttää kaasupolkimen ohjauksen, asettaa jarrupolkimen päälle asentoon, asettaa käsijarrun päälle sekä asettaa traktorin vaihdelaatikon asentoon P, eli park. Kuviossa 10 on esitetty kuvaaja testauksesta. Kuvaajassa vihreällä on esitetty kaasupolkimen asento, punaisella jarrupolkimen asento sekä vaalean sinisellä käsijarrun asento sekä vaaleanpunaisella ovikytkimen tila. Testi aloitetaan nostamalla kaasupolkimen asento noin 70 % kohdalle. Kohdassa 1 kytketään ovikytkin auki tilaan. Tämä aiheuttaa kaasupolkimen ohjauksen putoamisen, jarrupolkimen ohjauksen nousun 96 %:iin, sekä käsijarru ja P-vaihte aktivoituvat. Kohdassa 2 ovikytkin kytketään kiinni. Tämä pudottaa jarrupolkimen ohjauksen pois, mutta jättää käsijarrun sekä P-vaihteen aktiiviseksi. Tämä on haluttu toiminta, koska tämä estää traktorin liikkeelle lähdön heti oven sulkemisen jälkeen. Testin mukaan kyseinen turvaominaisuus toimii halutulla tavalla.

## 8.2 Ulkoisen ohjauksen testaus ja testitapaukset

Ulkoisen ohjauksen testaus toteutettiin niin kutsutuilla testitapauksilla. Testitapaukset kirjoitettiin käyttäen python-ohjelmointikieltä ja ne ajettiin simulaatioon käyttäen TestRunner -testiohjelmaa. Tulokset tallennettiin samaan tapaan simulaatiosovelluksen työkalulla kuvaajien muotoon. Ohjaukset tehtiin CAN-väylän kautta ohjausmoduulille. Testejä tehtiin lähes kaikkiin traktorin ominaisuuksiin liittyen, mutta tähän dokumenttiin käydään läpi vain oleelliset niistä.

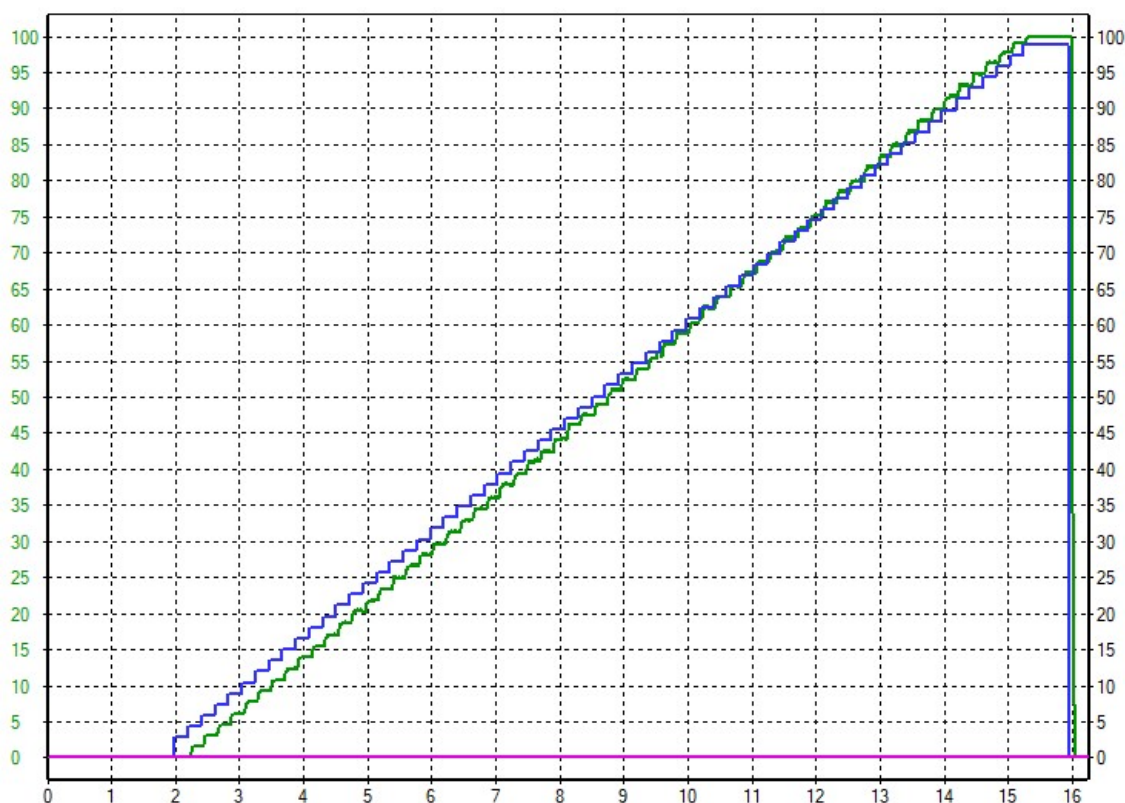
Testit ajettiin käyttäen TestRunner -ohjelmaa (kuvio 11). TestRunner -ohjelmaa varten testit kirjoitetaan käyttäen python-ohjelmointikieltä.



KUVIO 11. TestRunner -ohjelman perusnäky

TestRunner listaa testitapaukset sille määritetystä kansioista *Test Cases* -näky-mään. Sen oikealla puolella on varattu kenttä testitapaukselle kirjoitetulle informaatiolle. Informaatiokentän alapuolella on testitapauksen antamalle sanalliselle raportoinnille kenttä. Tämän alla on käyttäjän syöttämälle lisätiedolle tekstikenttä. Ohjelman vasemmassa reunassa on testitapauksien ajoon liittyvät toiminnot. Nappien toiminta järjestyksessä alkaen ylimmästä: ajaa kaikki testitapaukset listattuna testitapauksien listaan, ajaa valitun testin, pysäyttää käynnissä olevan testin, testien raportin kirjoitus levyille, lataa testit uudelleen kansioista ja sulje TestRunner -ohjelma.

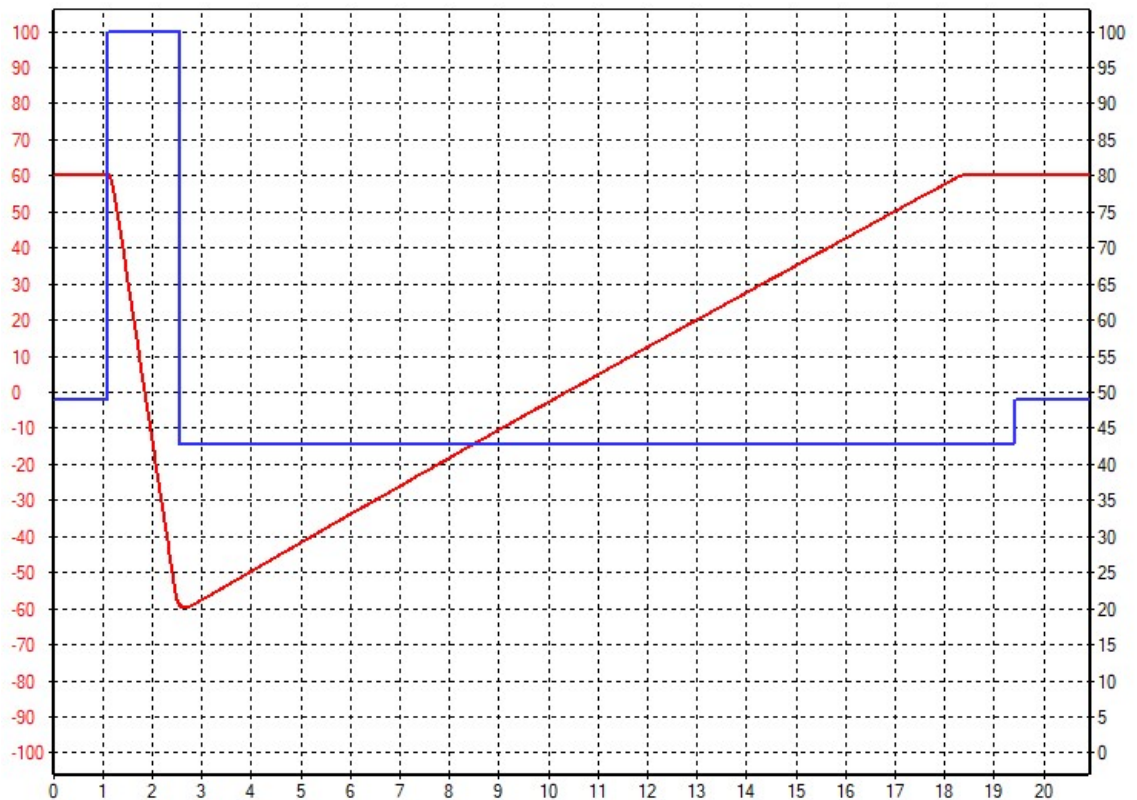




KUVIO 12. Testi 4: Simulaatiomallin kaasupolkimen arvon muutos verrattuna ulkoiseen ohjaukseen

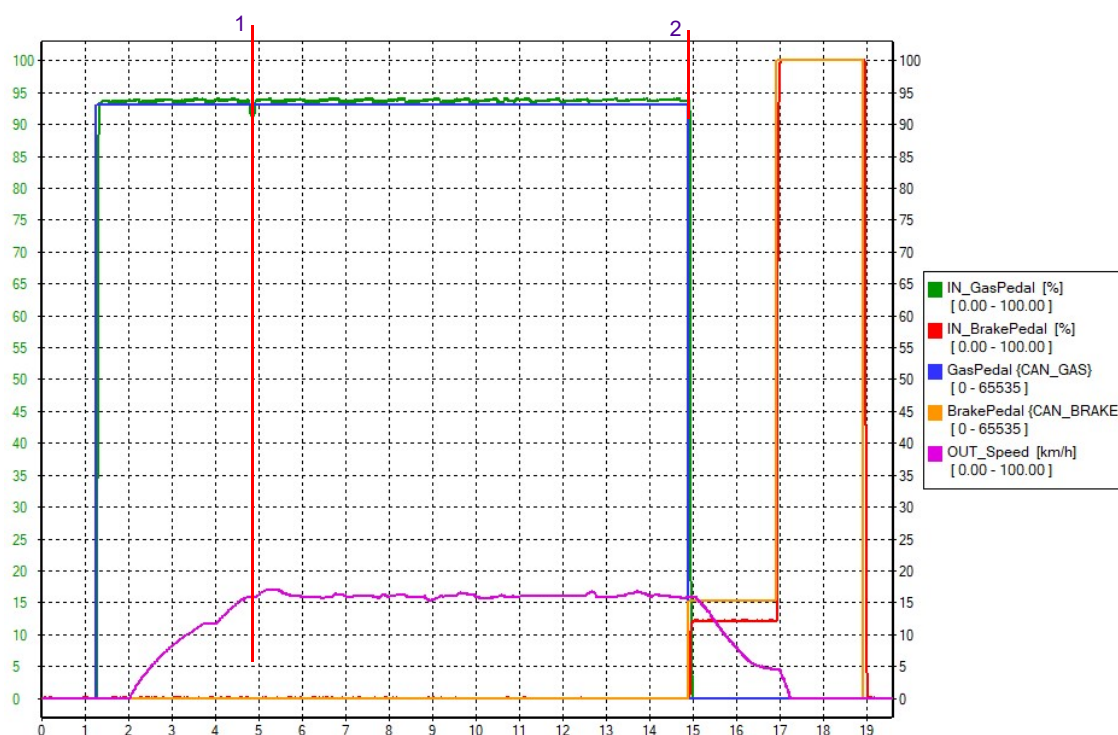
Ensimmäisessä ulkoisen ohjauksen testissä testattiin kaasupolkimen toimintaa (kuvio 12). Ohjaus syötettiin CAN-väylän kautta joka on kuvaajassa esitetty sini-sellä. Kaasupolkimen simulaatiosignaali on esitetty vihreällä. Ohjausarvo syötetään väliltä 0 – 65535, eli 16-bittisenä tietona. Ohjaussignaalin ja simulaatiosignaalin välille tuleva ero johtuu ohjausmoduulin ohjelmassa tehtyihin signaalinkäsittelyyn, jossa PWM -ulostulo skaalataan simulaatiosignaalia varten. Ulkoisen CAN-väylän kautta tehdyn ohjauksen arvot 1830 – 64000 vastaavat todellisuudessa simulaatiosignaalin 0 – 100 % arvoja. Kun tämä huomioidaan kuvaajassa, voidaan todeta, että kaasupolkimen ohjaus toimii halutulla tavalla.





KUVIO 13. Testi 5: Simulaatiomallin ohjauspyörän asennon muutos verrattuna ulkoisen ohjaukseen

Ohjauspyörän toimintaa testattiin ulkoisen ohjauksen kautta tässä testissä (kuvio 13). Kuvaajassa ulkoista ohjausta esitetään sinisellä ja punaisella simulaatiosignaalia, joka vastaa ohjauspyörän asentoa. Testissä ohjauspyörän asento aloitus-tilanteessa on  $60^\circ$ . Testissä ohjauspyörä ohjataan täydellä 100 % ohjauksella ensin täysin toiseen laitaan  $-60^\circ$  vastaavaan asentoon. Kun ohjauspyörä saavuttaa ääriasentonsa ohjauksen suunta käännetään ja annetaan noin -15 % ohjaus ohjauspyörää varten. Kun ohjauspyörä saavuttaa ääriasentonsa testi lopetetaan. Kuten manuaalisen ohjauksen testauksessa (kappale 8.1, Testi 1), tulkitaan kuvaajaa ohjausarvon suuruuden perusteella ja tarkkaillaan aiheutettua ohjauspyörän kulman muutosnopeutta. Täydellä ohjauksella ajan hetkellä 1,3 s ohjauspyörän asento vastaa  $50,48^\circ$  ja ajan hetkellä 2,3 s asento on  $-39,24^\circ$ . Pienemmällä -15 % ohjauksen aikana ajan hetkellä 3,5 s ohjauspyörän kulma on  $-53,67^\circ$  ja ajan hetkellä 17,5 s kulma on  $53,89^\circ$ . Näin ollen aluksi annetulla 100 % ohjauksella saavutetaan  $89,72^\circ/\text{s}$  kulman muutosnopeus ja -15 % ohjauksella itseisarvona noin  $7,68^\circ/\text{s}$  kulman muutosnopeus. Nämä vastaavat manuaaliohjauksessa havaittuja tuloksia, joten voidaan todeta ulkoisen ohjauksen toimivan halutulla tavalla.



KUVIO 14. Testi 6: Traktorin ajaminen eteenpäin suoralla tiellä

Ulkoisten ohjauksien viimeisessä testissä traktorin ominaisuuksia testattiin hie-  
man laajemmin. Traktorin oli tarkoitus ajaa eteenpäin, kytkeä vakionopeuden  
säädin noin 16 kilometrin tuntinopeudella ja lopuksi jarruttaa ja pysähtyä. Testin  
kuvaaja kuviossa 14. Kaasupolkimen ohjaus on esitetty sinisellä ja sen simulaa-  
tiosignaali vihreällä. Kaasupolkimen ohjaus on esitetty oranssilla sekä sen simu-  
laatiosignaali punaisella. Violetti kuvaaja esittää traktorin simulaatiomallin no-  
peutta. Kuvaajasta havaitaan traktorin nopeuden kasvavan kaasupolkimen oh-  
jauksen ansiosta. Nopeuden kasvaessa yli 16 km/h vakionopeuden säädin kyt-  
keytyy (kuvaajan kohta 1), eikä traktorin nopeus kasva suuremmaksi vaikka kaa-  
supolkimen ohjaus pysyy päällä. Kuvaajan kohdassa 2 traktorin kaasupolkimen  
ohjaus otetaan pois sekä heti sen jälkeen ohjataan jarrupolkimen asentoa noin  
15 %. Tämä aiheuttaa simulaatiomallin nopeuden hidastuksen noin 4 km/h no-  
peuteen. Tämän jälkeen jarrupolkimen ohjaus kasvatetaan täysin auki, joka ai-  
heuttaa traktorin simulaatiomallin pysähdyksen. Testin mukaan voidaan todeta  
ulkoisten ohjauksien toimivan halutulla tavalla.

## 9 POHDINTA

Opinnäytetyönä HiL-simulaatorratkaisun toteuttaminen traktorin ohjausmoduulille on osoittautunut hyvin laajaksi ja monialaista osaamista vaativaksi projektiksi. Projektin suhteellisen suppeat määritykset järjestelmän toiminnasta antoi suunnittelussa, toteutuksessa sekä ohjelmoinnissa hyvin vapaat kädet ja oman oppimansa hyödyntämisessä sekä omatoimisen oppimisen projektin eri osa-alueissa. Tietotaito järjestelmästä, käytetyistä suunnittelutyökaluista sekä järjestelmän testaamisesta karttui projektin edetessä niin eri lähdemateriaaleja itsenäisesti tutkiessa sekä työpaikalla vakituisena olevien suunnittelijoiden avustuksen kautta.

Opinnäytetyössä valmistettiin ENABLE-S3 -tutkimushanketta tukeva ratkaisu, joka mahdollistaa traktorin ohjausmoduulin käytön simulaatioympäristössä. Järjestelmässä on mahdollisuus käyttää traktorille kehitettyä esimerkkisovellusta ohjauslaitteiston ja ohjelmiston testaamiseen. Järjestelmää varten kehitettiin myös eri toimintoja varten testitapauksia, joilla testattiin ohjausjärjestelmän tärkeimmät ominaisuudet. Järjestelmässä on mahdollisuus testata täysin autonomisoituja ajoalgoritmeja ja tätä testattiin myös hyvin yksinkertaisella testitapauksella.

Järjestelmä saatiin valmistettua kompaktiksi vain yhden koteloinnin ympärille. Kotelointia varten valmistettiin myös erillinen kantokahva, joka mahdollistaa järjestelmän helpon liikuteltavuuden. Järjestelmän esteettisyyden toteutumisen osalta ei tehty testejä tai virallisia kyselyitä, mutta yksittäiset kommentit tutkimushankkeessa mukana olleilta yrityksiltä ovat olleet positiivisia.

Järjestelmä saatiin valmistettua annetussa aikamääreessä suunnitelluin osin muutamaa poikkeusta lukuun ottamatta. Järjestelmän ohjelma pyrittiin tekemään mahdollisimman selkokieliseksi sekä siihen lisättiin selkeyttäviä kommentteja. Sähkösuunnittelusta sekä mekaanisesta suunnittelusta tehtiin selkeät dokumentit, kokoonpanosta otettiin valokuvia ja kokonaisuus kerättiin verkkolevyille. Nämä avustavat järjestelmän jatkokehitystä ja kokonaisuudessaan uuden järjestelmän rakentamisenkin.

## LÄHTEET

Ars-Infromatica. n.d. Device Configuration Files. Luettu 2.5.2019. <http://www.ars-informatica.com/Root/Code/CANOPEN/DCF.aspx>

BusinessDictionary. n.d. Simulation. Luettu 24.3.2019. <http://www.businessdictionary.com/definition/simulation.html>

CiA. n.d. CAN lower- and higher-layer protocols. Luettu 10.4.2019. <https://www.can-cia.org/can-knowledge/>

CiA. n.d. CAN physical layer. Luettu 10.4.2019. <https://www.can-cia.org/can-knowledge/can/systemdesign-can-physicallayer/>

CiA. n.d. History of CAN technology. Luettu 10.4.2019. <https://www.can-cia.org/can-knowledge/can/can-history/>

CoDeSys. n.d. The System. Luettu 10.4.2019. <https://www.codesys.com/the-system.html>

Creanex Oy. 2016. MultilO24s V2.2 Specification V1.5. Käyttöohje.

Digi-Key. n.d. APEM Inc. IPR3FAD2L0G. Luettu 30.4.2019. <https://www.digi-key.com/product-detail/en/apem-inc/IPR3FAD2L0G/679-1144-ND/1280210>

Farnell. n.d. D-Sub Solder Female Stamp Pin. Luettu 30.4.2019. <http://www.farnell.com/datasheets/1899433.pdf>

Farnell. n.d. 637H/2 - Toggle Switch, (On)-Off-(On), SPDT, Non Illuminated, 600H Series, 10 A. Luettu 30.4.2019. <https://fi.farnell.com/apem/637h-2/switch-spdt-10a-250vac/dp/1607961?st=apem%20on%20off%20on>

Fibox. n.d. EKPE 130 T. Luettu 30.4.2019. [www.fibox.fi/catalog/product/1000/2538143\\_FIN1.pdf](http://www.fibox.fi/catalog/product/1000/2538143_FIN1.pdf)

FileInfo. 2019. INI file extension. Luettu 30.4.2019. <https://fileinfo.com/extension/ini>

International standard IEC 61131-3:2013.

ISTQB Glossary. n.d. Testing. Luettu 29.4.2019. <https://glossary.istqb.org/search/system%20testing>

Kankaanranta M. & Mäkelä T. n.d. Testaus lyhyesti. Luettu 29.4.2019. <http://smarteducation.jyu.fi/projekti/systech/Periaatteet/suunnittelun-periaatteet/testaus/testaus-lyhyesti>

Khan Academy. n.d. Slope formula. Luettu 2.5.2019. <https://www.khanacademy.org/math/cc-eighth-grade-math/cc-8th-linear-equations-functions/8th-slope/a/slope-formula>

Kvaser. 2014. Leaf Light v2 User's Guide. Luettu 27.4.2019. [https://www.kvaser.com/software/7330130980146/V1\\_2\\_189/kvaser\\_leaf\\_light\\_v2\\_usersguide.pdf](https://www.kvaser.com/software/7330130980146/V1_2_189/kvaser_leaf_light_v2_usersguide.pdf)

Kycon. n.d. KPJX-PM. Luettu 30.4.2019. <http://www.kycon.com/2013Catalogpage/DC%20Power/KPJX-PM.pdf>

National Instruments. 2019. The Basics of CANopen. Luettu 30.4.2019. <http://www.ni.com/fi-fi/innovations/white-papers/13/the-basics-of-canopen.html>

National Instruments. 2019. What is the Maximum Cable Length For a CAN Bus?. Luettu 3.6.2019. <https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z000000P7ikSAC>

Mathworks. n.d. Hardware-in-the-Loop (HIL) Simulation. Luettu 25.4.2019. <https://www.mathworks.com/discovery/hardware-in-the-loop-hil.html>

Mathworks. n.d. What Is Hardware-In-The-Loop Simulation? Luettu 6.4.2019. <https://it.mathworks.com/help/phymod/simscape/ug/what-is-hardware-in-the-loop-simulation.html>

Neutrik. n.d. NAUSB-W-B. Luettu 30.4.2019. <https://www.neutrik.com/en/product/nausb-w-b>

Neutrik. n.d. NE8FDP. Luettu 30.4.2019. <https://www.neutrik.com/en/product/ne8fdp>

nVent Schroff. 2018. MultipacPRO. [https://schroff.nvent.com/wcsstore/ExtendedSitesCatalogAssetStore/Attachment/SchroffAttachments/Documents/6\\_3\\_multipacPRO\\_e.pdf](https://schroff.nvent.com/wcsstore/ExtendedSitesCatalogAssetStore/Attachment/SchroffAttachments/Documents/6_3_multipacPRO_e.pdf)

OTTO. n.d. HTL2-2-Way. Luettu 30.4.2019. <https://www.otto-controls.com/htl2-2-way-single-axis-linear-hall-effect-finger-joystick-2>

OTTO. n.d. HTL4-4Way. Luettu 30.4.2019. <https://www.otto-controls.com/htl4-4-way-linear-hall-effect-finger-joystick-2>

Smith R.D. 1998. Simulation Article. <http://www.modelbenders.com/encyclopedia/encyclopedia.html>

Technopedia. n.d. Boot Loader. Luettu 29.4.2019. <https://www.techopedia.com/definition/3324/boot-loader>

Technopedia. n.d. Simulation. Luettu 24.4.2019. <https://www.techopedia.com/definition/5757/simulation>

Texas Instruments. 2016. Introduction to the Controller Area Network (CAN). <http://www.ti.com/lit/an/sloa101b/sloa101b.pdf>

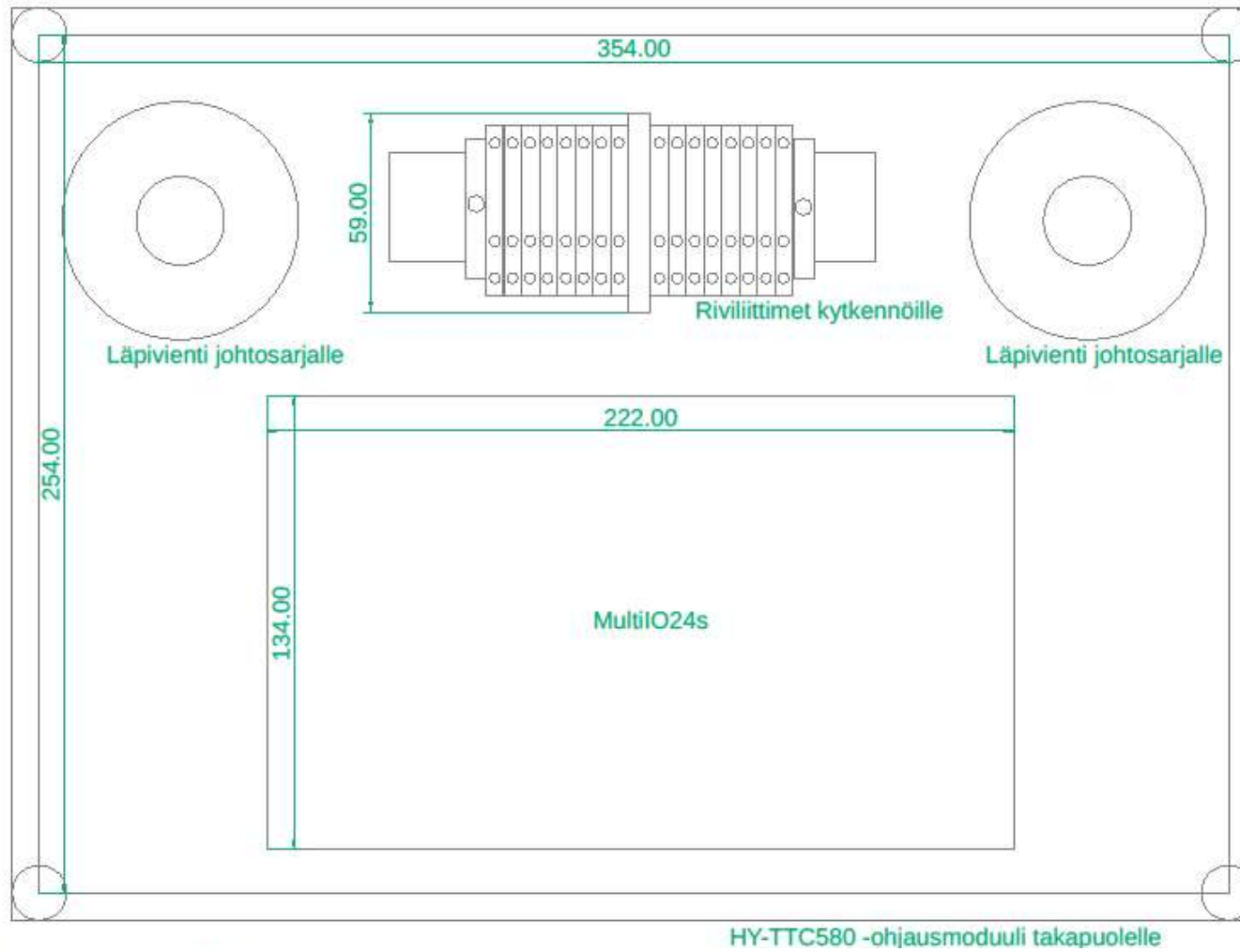
TTControl. n.d. High Performance Safety Controller – HY-TTC 580. Luettu 30.4.2019. [https://www.ttcontrol.com/wp-content/uploads/TTControl-HY-TTC\\_580-Datasheet.pdf](https://www.ttcontrol.com/wp-content/uploads/TTControl-HY-TTC_580-Datasheet.pdf)

TTControl. n.d. HY-TTC500 Family. Luettu 23.4.2019. <https://www.ttcontrol.com/products/electronic-control-units/safety-certified-controllers/hy-ttc-500-family/>

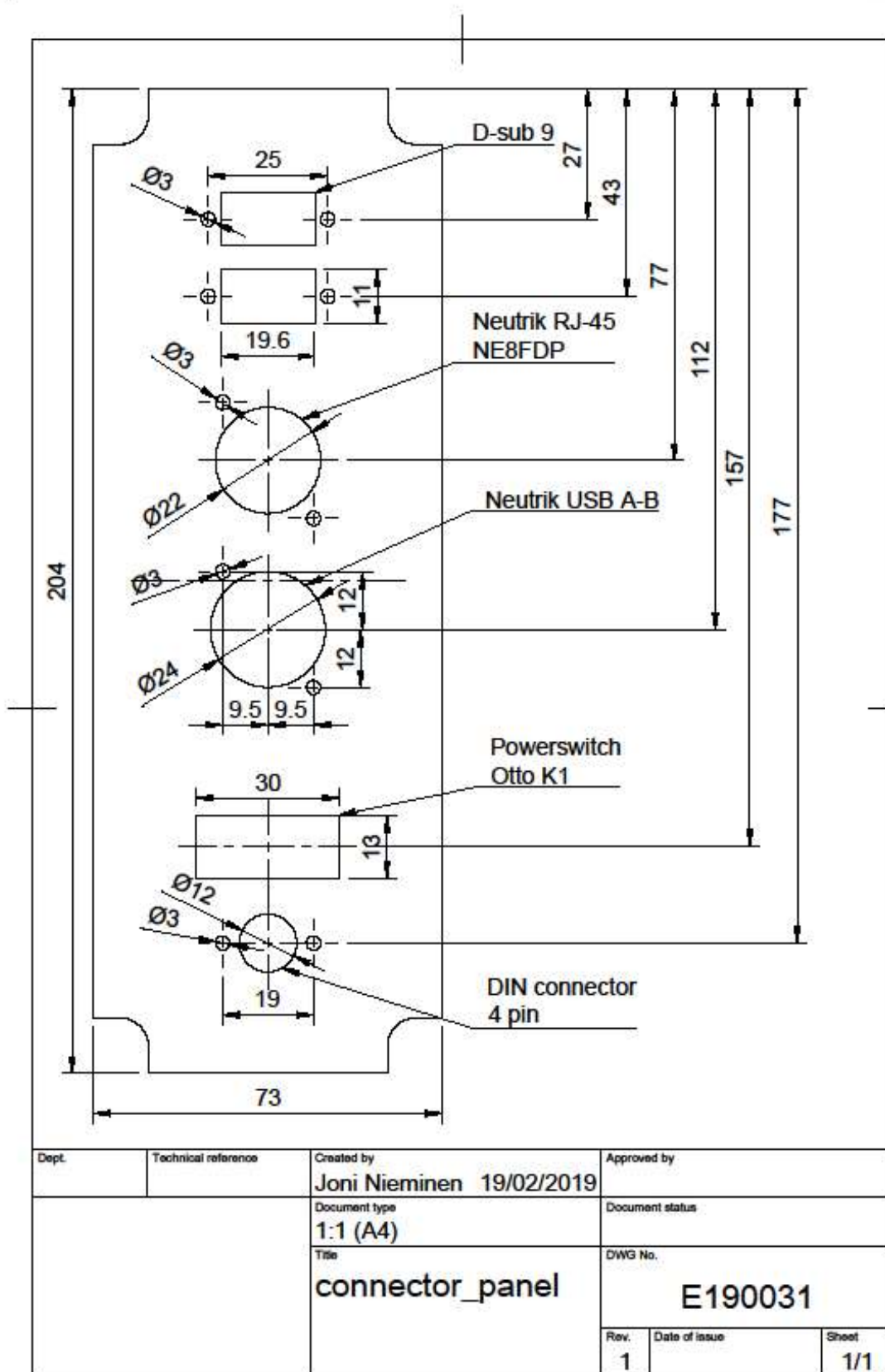
TTControl. n.d. HY-TTC 500 Family flyer. Luettu 23.4.2019. <https://www.ttcontrol.com/wp-content/uploads/TTControl-TTC-500-Family-Flyer.pdf>

**LIITTEET**

Liite 1. Alustava komponenttisijoittelukuva

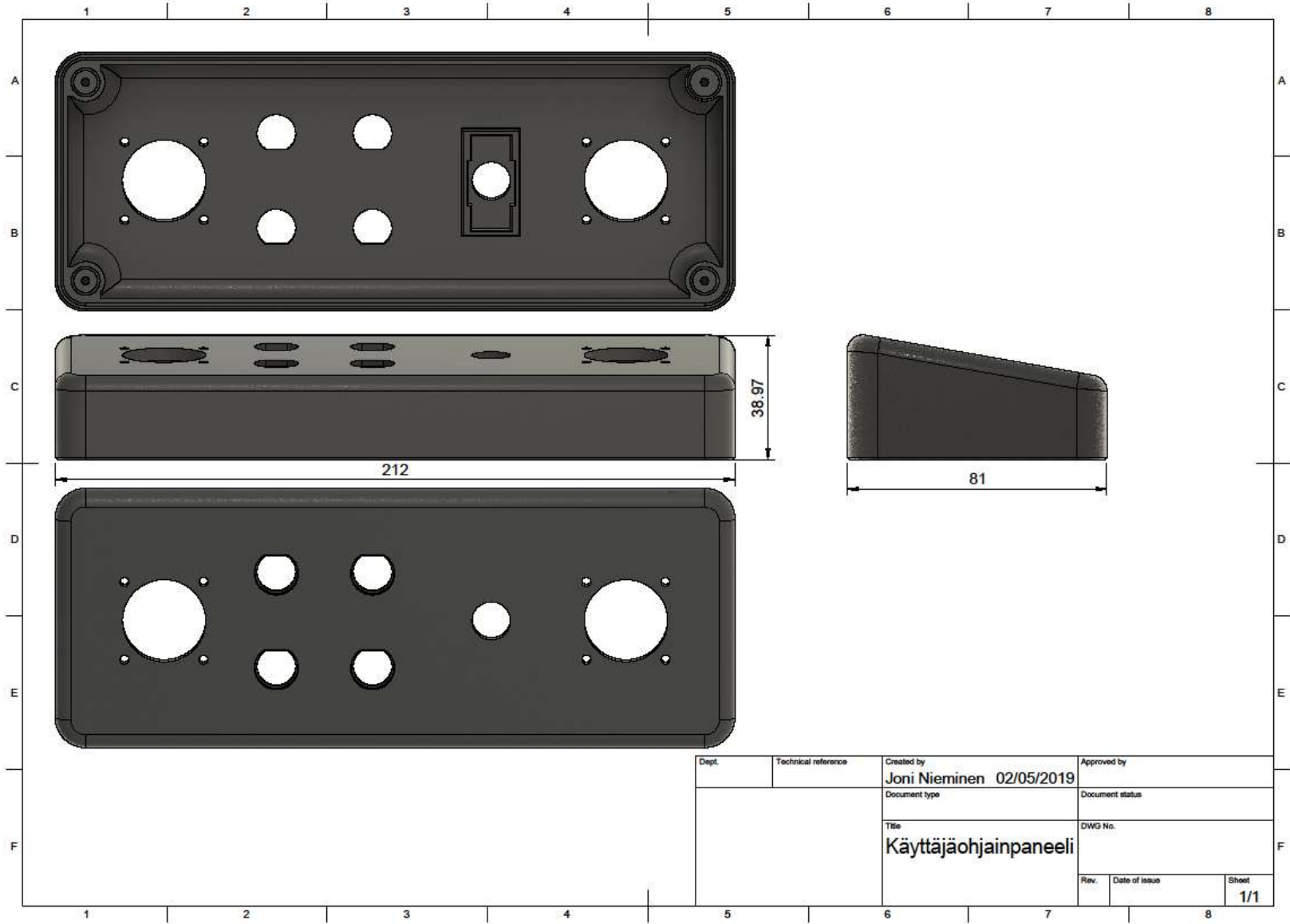


## Liite 2. Liitinpaneeli

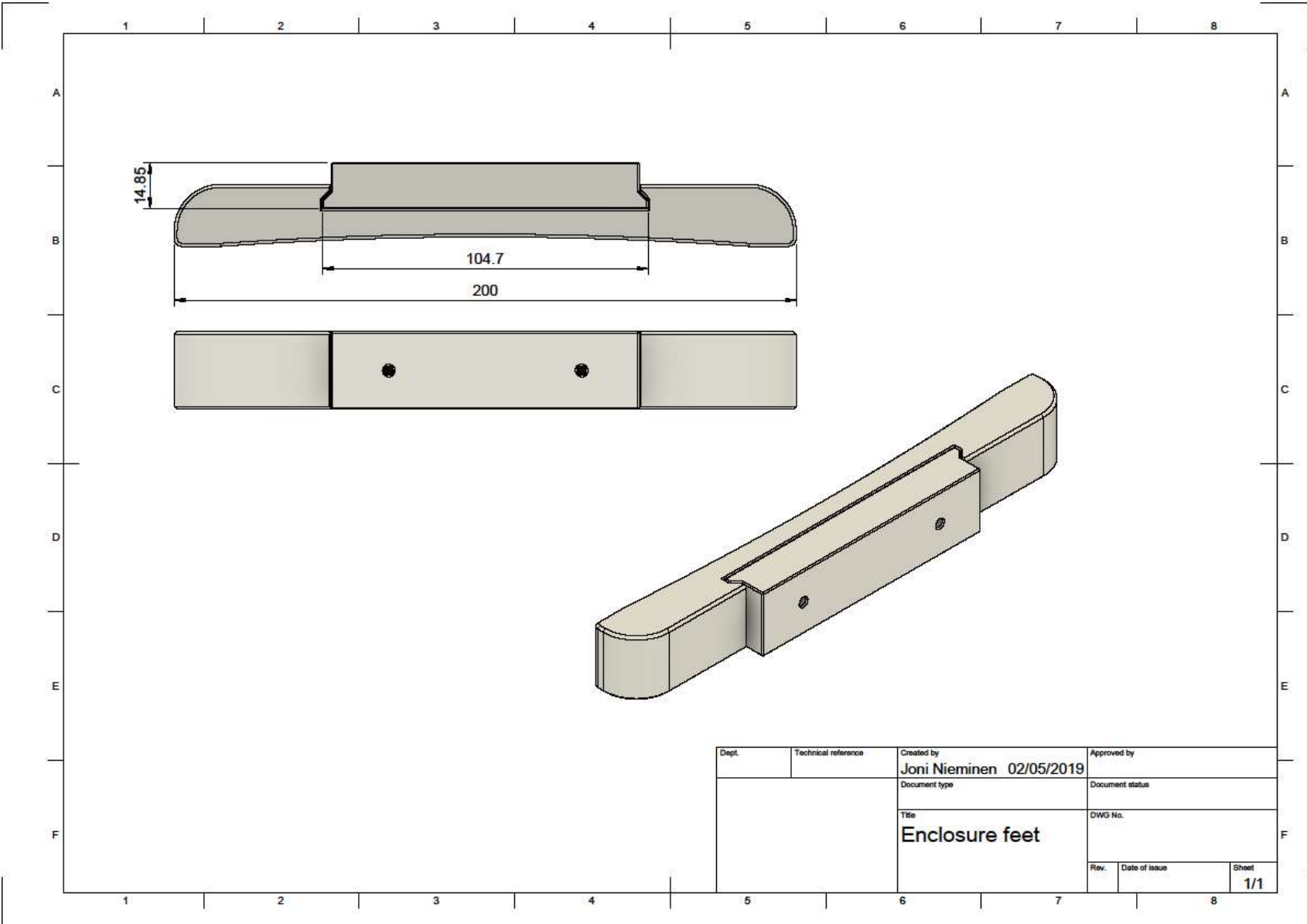




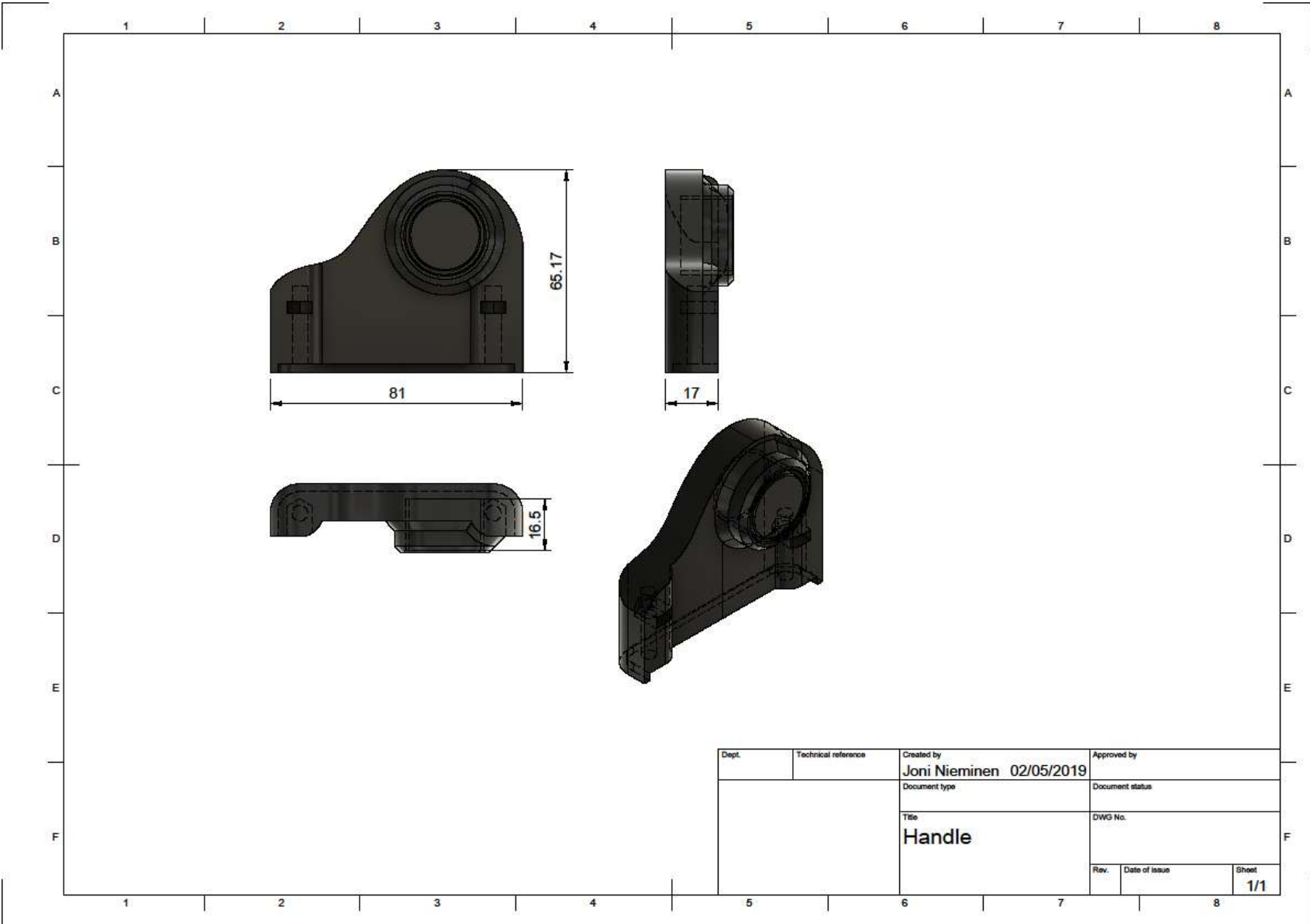
Liite 3. Käyttäjäohjainpaneelin tekninen piirustus



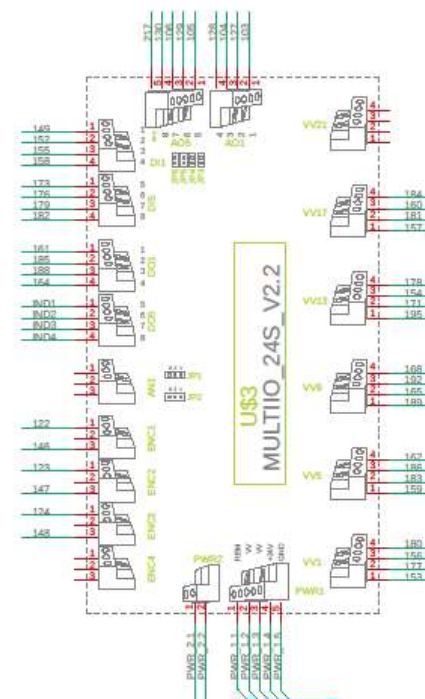
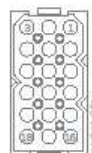
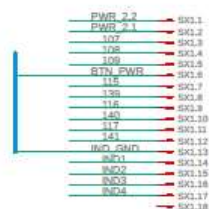
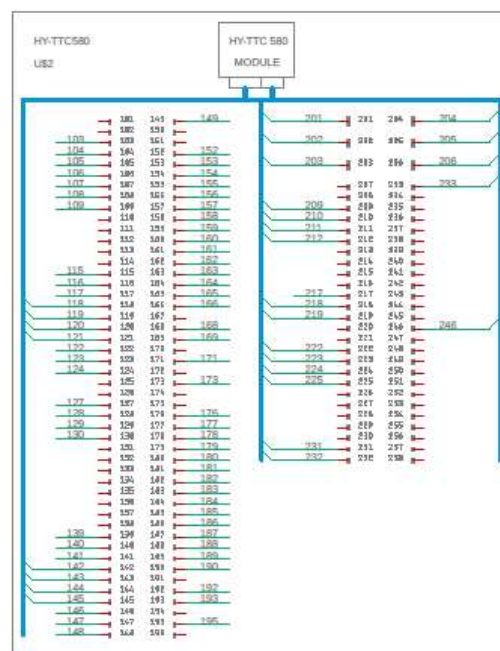
Liite 4. Tukijalat, tekninen piirustus



Liite 5. Kantokahvan tekninen piirustus







**Creanex oy**

TITLE: main\_wiring

Drawing Number: E190030

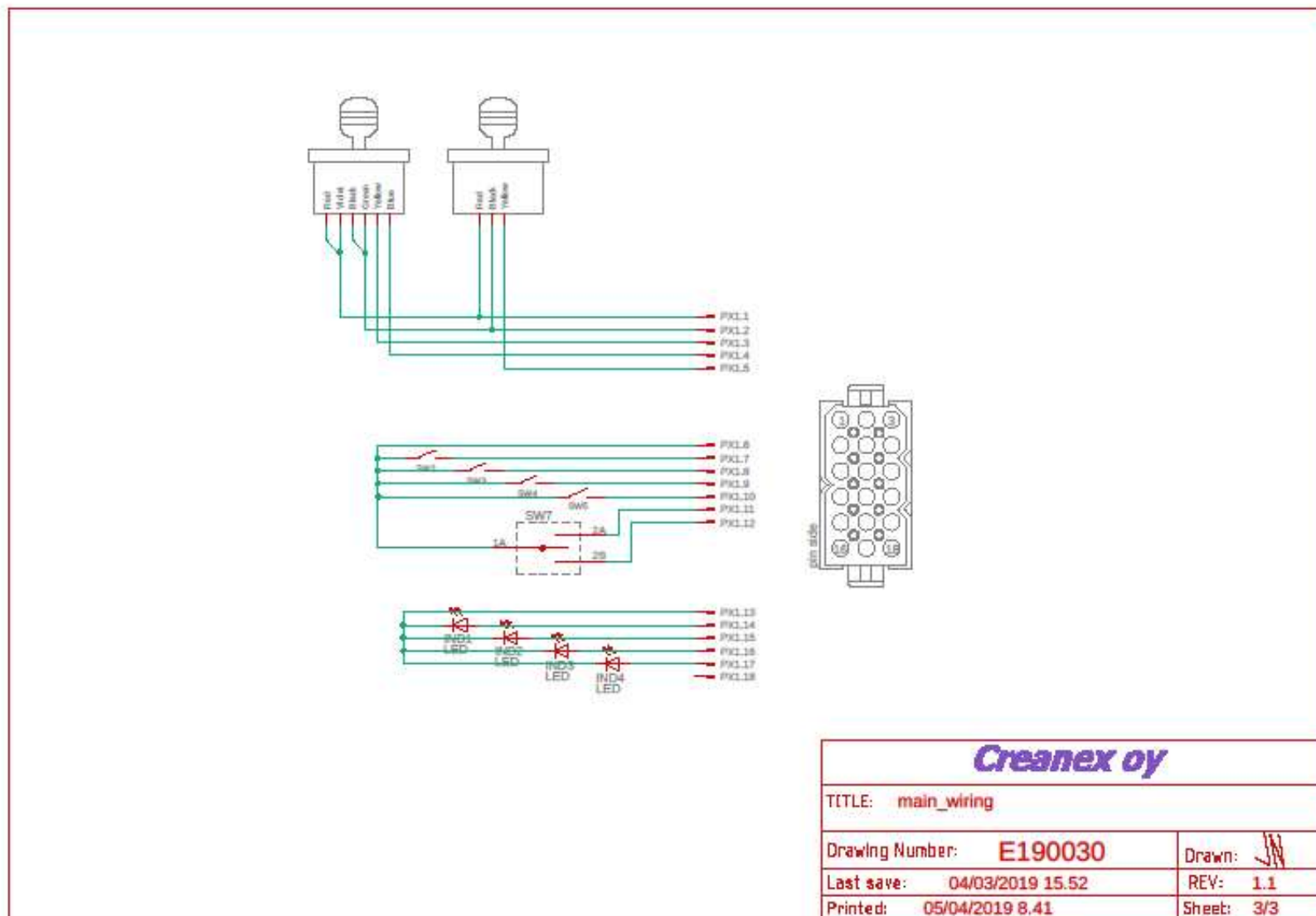
Drawn: *[Signature]*

Last save: 04/03/2019 15.52

REV: 1.1

Printed: 05/04/2019 8.41

Sheet: 2/3



## Liite 7. Järjestelmän ohjelma: Global Variable List

## Global Variable List: GVL

```

1      {attribute 'qualified_only'}
2      VAR_GLOBAL
3      (* Power stage and safety switch control variables for output control *)
4      gPowerStage ,
5      gSafetySwitch0 ,
6      gSafetySwitch1 ,
7      gSafetySwitch2      : lib500_types . IO_POWER_STATE := lib500_types .
IO_POWER_STATE . IO_POWER_OFF ;
8
9      (* CAN communication variables *)
10     gCan0      : tCan ;
11     CanGas ,
12     CanBrake ,
13     CanSteering      : WORD ;
14     CanSpeed      : REAL := 0.00 ;
15     CanEMGstop ,
16     CanButton1 ,
17     CanButton2 ,
18     CanButton3 ,
19     CanButton4 ,
20     CanSpeedControl ,
21     CanHandBrake ,
22     CanGearD ,
23     CanGearP ,
24     CanGearR ,
25     CanSpeedSlow ,
26     CanHitchEnd      : BOOL ;
27
28     (* Control panel joystick variables. *)
29     gLeftJoystick_X ,
30     gLeftJoystick_Y ,
31     gRightJoystick_Y      : lib500_types . IO_ADC_STATUS ;
32
33     (* Analog output variables. *)
34     gSteering ,
35     gHandBrake ,
36     gSpeedControl ,
37     gBrakePedal ,
38     gGasPedal      : WORD := 0 ;
39
40     (* Errors *)
41     gSteeringError ,
42     gHandBrakeError ,
43     gSpeedControlError ,
44     gBrakePedalError ,
45     gGasPedalError      : lib500_types . IO_ERRORTYPE := lib500_types .
IO_ERRORTYPE . IO_E_OK ;
46
47     (* Control panel buttons variables. Buttons are latching. Buttons and switch
are connected to BAT+.
48
Rocker switch
up : A, down : B, center : floating *)

```

Global Variable List: GVL

---

```

49      gButton1 ,
50      gButton2 ,
51      gButton3 ,
52      gButton4 ,
53      gRockerSwitchA ,
54      gRockerSwitchB      : lib500_types . IO_DI_STATUS ;
55
56      (* Digital output variables.*)
57      gHitch ,
58      gDoor ,
59      gAutManDrive ,
60      gEngineStarter ,
61      gEMGStop ,
62      gGearD ,
63      gGearP ,
64      gGearR              : BOOL := FALSE ;
65
66      (* Digital input variables *)
67      gEnableSSW ,          (* Simulator rises this bit when it
has initialized itself. After this Module enables powerstage and safetyswitches.
*)
68      gAutoDriveEnable     : lib500_types . IO_DI_STATUS ; (* Module takes commands
from CAN when this bit is 1. *)
69
70      (* Current measurement from PWM outputs *)
71      gHandBrakeCurrent ,
72      gSpeedControlCurrent ,
73      gBrakePedalCurrent ,
74      gGasPedalCurrent ,
75      gSteeringCurrent     : lib500_types . IO_PWM_CURRENT_QUEUE_E ;
76  END_VAR
77

```



## Liite 8. Järjestelmän ohjelma: PLC\_PRG

## POU: PLC\_PRG

---

```

1  PROGRAM PLC_PRG
2  VAR
3      InitModule : BOOL := TRUE;
4  END_VAR
5
6
7
8
9
10
11
12
13
14
15
16
17
18  CNTRLS_PRG ( );
19  Safety_CHK ( );
20  CANcomm ( );
21
22  GVL.gDoor := GVL.gButton2.pin_value;
23  GVL.gAutManDrive := GVL.gButton3.pin_value;
24
25  (* Error codes *)
26  GVL.gHandBrakeError;
27  GVL.gSpeedControlError;
28  GVL.gSteeringError;
29  GVL.gBrakePedalError;
30  GVL.gGasPedalError;
31
32  (* Current measurements of PWM outputs *)
33  GVL.gHandBrakeCurrent.values[0];
34  GVL.gSpeedControlCurrent.values[0];
35  GVL.gSteeringCurrent.values[0];
36  GVL.gBrakePedalCurrent.values[0];
37  GVL.gGasPedalCurrent.values[0];
38
39

```

---

## Liite 9. Järjestelmän ohjelma: CANComm

POU: CANcomm

---

```

1  PROGRAM CANcomm
2  VAR
3      fbCanRx1 ,
4      fbCanRx2   : CanRx ;
5      data1 ,
6      data2 ,
7      data3      : WORD ;
8      data4 ,
9      data5      : BYTE ;
10     dw : ARRAY [ 0 .. 3 ] OF BYTE ;
11     speedr1 : REAL ;
12     TxData : ARRAY [ 0 .. 7 ] OF BYTE ;
13     convdata : REAL ;
14 END_VAR
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33

```

---

```

1  (* For receiving CAN message from specific ID. *)
2  fbCanRx1 ( iCAN := ADR ( GVL.gCan0 ) ,
3             iCanId := 16#182 , (* CAN message ID *)
4             iCanIdExtended := FALSE , (* Uses standard ID lenght *)
5             iTimeoutLimit := T#500MS , (* After specified time function rises
oCanTimeout. *)
6             oByteCount => ,
7             oData => ,
8             oNewData => ,
9             oCanTimeout => ,
10            oReceiverId => ,
11            oTooManyReceivers => );
12
13 IF fbCanRx1.oNewData THEN
14     GVL.CanGas := cl_make_word ( byte0 := fbCanRx1.oData [ 0 ] ,
15                                byte1 := fbCanRx1.oData [ 1 ] );
16     GVL.CanBrake := cl_make_word ( byte0 := fbCanRx1.oData [ 2 ] ,
17                                  byte1 := fbCanRx1.oData [ 3 ] );
18     GVL.CanSteering := cl_make_word ( byte0 := fbCanRx1.oData [ 4 ] ,
19                                     byte1 := fbCanRx1.oData [ 5 ] );
20     data4 := fbCanRx1.oData [ 6 ] ;
21     data5 := fbCanRx1.oData [ 7 ] ;
22 ELSEIF fbCanRx1.oCanTimeout THEN
23     GVL.CanGas := 0 ;
24     GVL.CanBrake := 0 ;
25     GVL.CanSteering := 0 ;
26     data4 := 0 ;
27     data5 := 0 ;
28 END_IF
29
30 cl_unpack_bits ( input := data4 ,
31                 bit0 => ,
32                 bit1 => ,
33                 bit2 => ,

```

---

POU: CANcomm

---

```

34             bit3 => ,
35             bit4 => ,
36             bit5 => ,
37             bit6 => ,
38             bit7 => ) ;
39
40   GVL.CanEMGstop      := cl_unpack_bits . bit0 ;
41   GVL.CanButton1      := cl_unpack_bits . bit1 ;
42   GVL.CanButton2      := cl_unpack_bits . bit2 ;
43   GVL.CanButton3      := cl_unpack_bits . bit3 ;
44   GVL.CanButton4      := cl_unpack_bits . bit4 ;
45   GVL.CanSpeedControl := cl_unpack_bits . bit5 ;
46   GVL.CanHandBrake    := cl_unpack_bits . bit6 ;
47   GVL.CanGearD        := cl_unpack_bits . bit7 ;
48
49   cl_unpack_bits (    input := data5 ,
50                     bit0 => ,
51                     bit1 => ,
52                     bit2 => ,
53                     bit3 => ,
54                     bit4 => ,
55                     bit5 => ,
56                     bit6 => ,
57                     bit7 => ) ;
58
59   GVL.CanGearP        := cl_unpack_bits . bit0 ;
60   GVL.CanGearR        := cl_unpack_bits . bit1 ;
61   GVL.CanSpeedSlow    := cl_unpack_bits . bit2 ;
62   GVL.CanHitchEnd     := cl_unpack_bits . bit3 ;
63

```

## Liite 10. Järjestelmän ohjelma: CNTRLS\_PRG

## POU: CNTRLS\_PRG

---

```

1  PROGRAM CNTRLS_PRG
2  VAR
3      JS_MIN : WORD := 500 ;
4      JS_MAX : WORD := 4500 ;
5      JS_DeadbandUP : WORD := 2550 ;
6      JS_DeadbandDN : WORD := 2450 ;
7      swapHB1 : UINT ;
8      swapHB2 : UINT ;
9      swapBP1 : UINT ;
10     swapBP2 : UINT ;
11     DA_HB ,
12     DA_SC : WORD ;
13     filter : cl_pt1_filter_dint ;
14     avg : cl_mov_average_dint_50 ;
15     handbrake : DINT ;
16     afil : DINT ;
17 END_VAR
18
19
20 IF ( Safety_CHK . safety_enabled = FALSE ) THEN
21
22 IF GVL.gAutManDrive = FALSE THEN (* Automatic drive (Button 3) is not
23     active, module reads inputs from joysticks and buttons. When active, module
24     controls PWM output with commands from CAN. *)
25
26     (* Control panel buttons route to output pins *)
27     GVL.gHitch := GVL.gButton1.pin_value ;
28     GVL.gEngineStarter := GVL.gButton4.pin_value ;
29     GVL.gGearD := GVL.gRockerSwitchA.pin_value ;
30     GVL.gGearR := GVL.gRockerSwitchB.pin_value ;
31
32 IF ( GVL.gRockerSwitchA.pin_value = FALSE AND GVL.gRockerSwitchB.pin_value
33     = FALSE AND PLC_PRG.InitModule = FALSE ) THEN (* *)
34     GVL.gGearP := TRUE ;
35 ELSE
36     GVL.gGearP := FALSE ;
37 END_IF
38
39     (* Control panel joysticks route and scale to output pins *)
40     (* Left Joystick X-axis *)
41
42     GVL.gSteering := DINT_TO_WORD ( PID ( SP := GVL.gLeftJoystick_X.value ,
43         FB := GVL.gSteeringCurrent.values [ 0 ] ,
44         S_iMIN := JS_MIN ,
45         S_iMAX := JS_MAX ,
46         S_oMIN := 0 ,
47         S_oMAX := 1000 ,
48         oMIN := 0 ,
49         oMAX := 65535 ) ) ; (* Control PWM out with
50     analog joystick through PID controller, current feedback
51     *)

```

---

## POU: CNTRLS\_PRG

```

29
30  (* Left Joystick Y-axis *)
31  swapHB1 := 65535 XOR (DWORD_TO_UINT (GVL.gLeftJoystick_Y.value));
32  (* Joystick gives values from 2500 -> 500 when using it towards - (minus) axis.
33  This horrendous XOR+scale operation inverts that (500 - 2500). *)
34  swapHB2 := DINT_TO_UINT (cl_scaling_dint (input := UDINT_TO_DINT (swapHB1),
35      in_min := 65535 - JS_DeadbandDN,
36      in_max := 65535 - 500,
37      out_min := JS_MIN,
38      out_max := JS_DeadbandDN,
39      saturate := TRUE));
40
41  IF (GVL.gLeftJoystick_Y.value < JS_DeadbandDN) THEN
42      GVL.gHandBrake := DINT_TO_WORD (PID (SP := swapHB2,
43          FB := GVL.gHandBrakeCurrent.values[0],
44          S_iMIN := JS_MIN,
45          S_iMAX := JS_DeadbandDN,
46          S_oMIN := 0,
47          S_oMAX := 1000,
48          oMIN := 0,
49          oMAX := 65535)); (* Control PWM out with
50      analog joystick through PID controller, current feedback *)
51  END_IF
52  IF (GVL.gLeftJoystick_Y.value > JS_DeadbandUP) THEN
53      GVL.gSpeedControl := DINT_TO_WORD (PID (SP := GVL.gLeftJoystick_Y.value,
54          FB := GVL.gSpeedControlCurrent.values
55      [0],
56          S_iMIN := JS_DeadbandUP,
57          S_iMAX := JS_MAX,
58          S_oMIN := 0,
59          S_oMAX := 1000,
60          oMIN := 0,
61          oMAX := 65535)); (* Control PWM out
62      with analog joystick through PID controller, current feedback *)
63  END_IF
64  IF (JS_DeadbandUP > GVL.gLeftJoystick_Y.value AND JS_DeadbandDN < GVL.
65      gLeftJoystick_Y.value) THEN
66      GVL.gHandBrake := 0;
67      GVL.gSpeedControl := 0; (* Zero control values if joystick is centered *)
68  END_IF
69  (* Right Joystick Y-axis *)
70  swapBP1 := 65535 XOR (DWORD_TO_UINT (GVL.gRightJoystick_Y.value));
71  (* Joystick gives invert values from 2500 -> 500 when using it towards - axis.
    This horrendous XOR+scale operation inverts that. *)
    swapBP2 := DINT_TO_UINT (cl_scaling_dint (input := UDINT_TO_DINT (swapBP1),
        in_min := 65535 - JS_DeadbandDN,

```

## POU: CNTRL5\_PRG

```

72         in_max := 65535 - 500 ,
73         out_min := JS_MIN ,
74         out_max := JS_DeadbandDN ,
75         saturate := TRUE ) ) ;
76
77 IF ( GVL.gRightJoystick_Y.value < JS_DeadbandDN ) THEN
78 GVL.gBrakePedal := DINT_TO_WORD ( PID ( SP := swapBP2 ,
79         FB := GVL.gBrakePedalCurrent.values [ 0 ] ,
80         S_iMIN := JS_MIN ,
81         S_iMAX := JS_DeadbandDN ,
82         S_oMIN := 0 ,
83         S_oMAX := 1000 ,
84         oMIN := 0 ,
85         oMAX := 65535 ) ) ; (* Control PWM out with
analog joystick through PID controller, current feedback *)
86
87 END_IF
88 IF ( GVL.gRightJoystick_Y.value > JS_DeadbandUP ) THEN
89 GVL.gGasPedal := DINT_TO_WORD ( PID ( SP := GVL.gRightJoystick_Y.value ,
90         FB := GVL.gGasPedalCurrent.values [ 0 ] ,
91         S_iMIN := JS_DeadbandUP ,
92         S_iMAX := JS_MAX ,
93         S_oMIN := 0 ,
94         S_oMAX := 1000 ,
95         oMIN := 0 ,
96         oMAX := 65535 ) ) ; (* Control PWM out with
analog joystick through PID controller, current feedback *)
97
98 END_IF
99 IF ( JS_DeadbandUP > GVL.gRightJoystick_Y.value AND JS_DeadbandDN < GVL.gRightJoystick_Y.value ) THEN
100     GVL.gGasPedal := 0 ;
101     GVL.gBrakePedal := 0 ; (* Zero control values if joystick is centered *)
102 END_IF
103
104 ELSE
105     (* Control PWM out with CAN messages through PID controller, current
feedback *)
106
107     GVL.gGasPedal := DINT_TO_WORD ( PID ( SP := WORD_TO_DWORD ( GVL.
CanGas ) ,
108         FB := GVL.gGasPedalCurrent.
values [ 0 ] ,
109         S_iMIN := 0 ,
110         S_iMAX := 65535 ,
111         S_oMIN := 0 ,
112         S_oMAX := 1000 ,
113         oMIN := 0 ,
114         oMAX := 65535 ) ) ;
115
116     GVL.gBrakePedal := DINT_TO_WORD ( PID ( SP := WORD_TO_DWORD ( GVL.

```



## POU: CNTRLs\_PRG

```

CanBrake ) ,
117                                     FB      := GVL . gBrakePedalCurrent .
values [ 0 ] ,
118                                     S_iMIN  := 0 ,
119                                     S_iMAX  := 65535 ,
120                                     S_oMIN  := 0 ,
121                                     S_oMAX  := 1000 ,
122                                     oMIN    := 0 ,
123                                     oMAX    := 65535 ) ) ;
124
125     GVL . gSteering := DINT_TO_WORD ( PID ( SP      := WORD_TO_DWORD ( GVL .
CanSteering ) ,
126                                     FB      := GVL . gSteeringCurrent .
values [ 0 ] ,
127                                     S_iMIN  := 0 ,
128                                     S_iMAX  := 65535 ,
129                                     S_oMIN  := 0 ,
130                                     S_oMAX  := 1000 ,
131                                     oMIN    := 0 ,
132                                     oMAX    := 65535 ) ) ;
133
134     GVL . gEMGStop := GVL . CanEMGstop ;
135     GVL . gHitch  := GVL . CanButton1 ;
136     GVL . gDoor   := GVL . CanButton2 ;
137     GVL . gAutManDrive := GVL . CanButton3 ;
138     GVL . gEngineStarter := GVL . CanButton4 ;
139     GVL . gGearD  := GVL . CanGearD ;
140     GVL . gGearP  := GVL . CanGearP ;
141     GVL . gGearR  := GVL . CanGearR ;
142
143     IF ( GVL . CanHandBrake = TRUE ) THEN
144         DA_HB := 65535 ;
145     ELSE
146         DA_HB := 0 ;
147     END_IF
148     IF ( GVL . CanSpeedControl = TRUE ) THEN
149         DA_SC := 65535 ;
150     ELSE
151         DA_SC := 0 ;
152     END_IF
153
154     GVL . gHandBrake := DINT_TO_WORD ( PID ( SP      := WORD_TO_DWORD ( DA_HB
) ,
155                                     FB      := GVL . gHandBrakeCurrent .
values [ 0 ] ,
156                                     S_iMIN  := 0 ,
157                                     S_iMAX  := 65535 ,
158                                     S_oMIN  := 0 ,
159                                     S_oMAX  := 1000 ,
160                                     oMIN    := 0 ,
161                                     oMAX    := 65535 ) ) ;
162     GVL . gSpeedControl := DINT_TO_WORD ( PID ( SP      := WORD_TO_DWORD ( DA_SC

```

---

POU: CNTRLS\_PRG

---

```
163      ) ,
164      gSpeedControlCurrent . values [ 0 ] ,
165
166      S_iMIN := 0 ,
167      S_IMAX := 65535 ,
168      S_oMIN := 0 ,
169      S_oMAX := 1000 ,
170      oMIN := 0 ,
171      oMAX := 65535 ) ) ;
172
173      END_IF
174      END_IF
```



## Liite 11. Järjestelmän ohjelma: Safety\_CHK

## POU: Safety\_CHK

---

```

1  PROGRAM Safety_CHK
2  VAR
3      appld : WORD := 650 ;
4      PWM_on : WORD := 49000 ;
5      PWM_off : WORD := 0 ;
6      safety_enabled : BOOL := FALSE ;
7  END_VAR
8
9
10
11
12
13
14
15
16
17
18
19
20
21

```

---

```

1  IF ( GVL.gGasPedal > appld AND GVL.gBrakePedal > appld ) THEN (* if gas
    and brake pedal is applied simultaneously, gaspedal is set to 0 *)
2      GVL.gGasPedal := PWM_off ;
3  END_IF
4
5  IF ( GVL.gDoor = TRUE ) THEN (* if door is opened it will stop the machine
    and prevents it to move *)
6      safety_enabled := TRUE ;
7      GVL.gGasPedal := PWM_off ;
8      GVL.gBrakePedal := PWM_on ;
9      GVL.gHandBrake := PWM_on ;
10     GVL.gGearD := FALSE ;
11     GVL.gGearR := FALSE ;
12     GVL.gGearP := TRUE ;
13 ELSE
14     safety_enabled := FALSE ;
15 END_IF
16
17 IF ( GVL.CanHitchEnd = FALSE ) THEN (* HitchEnd is TRUE when hitch is either
    bottom or top endstop. This stops vehicle when actuating the hitch *)
18     GVL.gGasPedal := PWM_off ;
19     GVL.gBrakePedal := PWM_on ;
20 END_IF
21

```

---

## Liite 12. Järjestelmän ohjelma: PID

POU: PID

---

```

1  FUNCTION PID : DINT
2  VAR_INPUT
3      SP : DWORD; (* SetPoint *)
4      FB : WORD; (* Feedback *)
5      S_iMIN : WORD; (* Scale In Min *)
6      S_iMAX : WORD; (* Scale In Max *)
7      S_oMIN : WORD; (* Scale Out Min*)
8      S_oMAX : WORD; (* Scale Out Max *)
9      oMIN : DINT; (* PID Out Min *)
10     oMAX : DINT; (* PID Out Max *)
11
12  END_VAR
13  VAR
14      fbPIDController : cl_pid;
15      PIDout : DINT;
16  END_VAR
17

```

---

```

1  fbPIDController (
2      setpoint := cl_scaling_dint (   input := DWORD_TO_DINT ( SP ),
3                                      in_min := S_iMIN ,
4                                      in_max := S_iMAX ,
5                                      out_min := S_oMIN ,
6                                      out_max := S_oMAX ,
7                                      saturate := TRUE ),
8
9      feedback := FB ,
10     fw_gain := ,
11     p_gain := ,
12     i_gain := ,
13     d_gain := ,
14     output_min := oMIN ,
15     output_max := oMAX ,
16     reset_hold := ,
17     reset_val := ,
18     output => PIDout ,
19     control_error => ,
20     integrator => );
21
22  PID := PIDout ;

```

---