EESTI MAAÜLIKOOL
Tehnikainstituut

Gert Truu

PNEUMOSÕIDUKI ANDMEHÕIVE JA AUTOMAATJUHTIMISE SÜSTEEM

AUTOMATIC SYSTEM FOR DATA ACQUISITION AND CONTROL OF PNEUMATIC VEHICLE

Magistritöö
energiakasutuse erialal

Juhendaja: lektor Erkki Jõgi, MSc

Tartu 2016

www.emu.ee

Eesti Maaülikool

Estonian University of Life Sciences

Eesti Maaülikool Fr. R. Kreutzwaldi 1, Tartu 51014		Magistritöö lühikokkuvõte	
Autor: Gert Truu		Õppekava: Energiakasutus	
Pealkiri: Pneumosõiduki andmehõive ja automaatjuhtimise süsteem			
Lehekülgi: 58	Jooniseid: 36	Tabeleid: 11	Lisasid: 22
Osakond: Energeetika Uurimisvaldkond (ja mag. töö puhul valdkonna kood): ETIS teadusvaldkond: 4. Loodusteadused ja tehnika ETIS teaduseriala (alamvaldkond): 4.17 Energeetikaalased uuringud CERCS eriala: T140 Energeetika Juhendaja: Erkki Jõgi Kaitsmiskoht ja -aasta: Tartu, 2016			

Aasta 2008 mai korraldati ungaris esimene Pneumosõidukite võistlus. Kuni praeguse aastani 2016 on igal aastal mai kuus korraldatud sama võistlust. Võistluse eesmärgiks on säästlikult kulutada suruõhku, mida hoitakse 10 L ja 200 bar balloonis. Võistluse peakorraldajaks on Bosch Rexroth grupp Aventics.

Masina üks peamisi juhtimise osi on automaatika. Kuna automaatika ei saa ilma liikumist tekitada on peamiseks osaks siiski masina mehaanika. Juhtimise automaatika koosneb tööstuslikust loogikakontrollerist, täituritest ja anduritest. Automaatjuhtimise ülessandeks on juhtida mehanismide liikumist vastavalt anduritelt saadavale tagasiside informatsioonile.

Uurimustöö põhieesmärgiks oli projekteerida ja valmis ehitada pneumosõiduki automaatjuhtimise süsteem ja võistelda Ungaris.

Märksõnad: pneumosõiduk, juhtsüsteem, telemeetria, programmeerimine, pneumaatika

www.emu.ee
Eesti Maaülikool
Estonian University of Life Sciences

Estonian University of Life Sciences Fr. R.Kreutzwaldi 1, Tartu 51014		Abstract of Master's Thesis	
Author: Gert Truu		Speciality: Energy Application Engineering	
Title: Automatic System for Data Acquisition and Control of Pneumatic Vehicle			
Pages: 58	Figures: 36	Tables: 11	Appendixes: 22
Department: Energy managment Field of research (and for Master's Thesis add research field code): Natural Sciences and Engineering 4.17 Energetic Research T140 Energy research Supervisor: Erkki Jõgi Place and date: Tartu, 2016			

Year 2008 May started the event Pneumobile in Hungary. Year 2008 til now every year students have builted a vehicle witch energy source is 10L copressed air bottle at 200 bar of preassure. Main organaizer of the event is Bosch Rexroth group Aventics. Main problrms competiting the race is how to use the copressed air in most economical way. Following work introduces automatic control system of the pneumatic vehicle.

Control automatics is important but not the most important part of the penumatic vehicle. Vehicle starts with mechanical build and penumatic system build. PLC, electronic actuators and sensors helps to automate the mechanical system. Pneumatic motor witch is introduced in follwing research is to complex to operate it manualy so Industrial PLC is used.

Main goal of following research was to build a automation system of Pneumobile vehicle and competite in Hungary with other teams.

Key words: Pneumobile, Control system, Telemetry, Programming, Pneumatics

SISKUKORD

LÜHIKOKKUVÕTE **2**

ABSTRACT **3**

TÄHISED JA LÜHENDID **6**

SISSEJUHATUS **8**

1. PROJEKTEERIMISE TINGIMUSED JA SEADMETE VALIK **10**

1.1 Süsteemi kirjeldus 10

1.2 Pneumosõiduki automaatikas ja andmehõives kasutatavad seadmed 11

2. PNEUMOSÕIDUKI AUTOMAATIKA KIRJELDUS **13**

2.1 Programmeeritav loogikakontroller 13

2.2 Arvuti abiseadmete juhtmiseks 19

2.3 Pneumosüsteemi andurid 20

2.4 Pneumosüsteemi täiturid 21

3. PNEUMOSÕIDUKI AUTOMAATJUHTIMINE **24**

3.1 Programmeerimise keeled 24

3.2 Andurite ja täiturite juhtimine loogikakontrolleriga 30

3.3 Automaatjuhtimise kirjeldus 31

3.4 Automaatika skeemid 34

3.5 Piirangud ja võimalused 35

4. ANDMEHÕIVE JA KAUGJUHTIMINE **36**

4.1 Andmehõive süsteemi kirjeldus 36

4.2 Andmete salvestamine 40

4.3 Graafiline liides juhtimiseks ja tekitatavad graafikud 42

KOKKUVÕTE **53**

SUMMARY **55**

KIRJANDUS **57**

LISAD **59**

LISA A 60

LISA B 62

LISA C 64

LISA D 66

LISA E 68

LISA F 70

LISA G 72

LISA H 74

LISA I 76

LISA J 78

LISA K 80

LISA L 82

LISA M 84

LISA N 86

LISA O Globaal muutujad 88

LISA P Põhiprogramm 90

LISA Q Pneumojaotite lülitamise funktsiooniplokk 92

LISA R Käigu vahetamise süsteem 94

LISA S Mootori ja masina kiiruse mõõtmine 97

LISA T Analoog sisend 100

LISA U Enkooder 104

LISA V Analoog väljund 106

LISA S Lihtlitsents lõputöö salvestamiseks ja üldsusele kättesaadavaks tegemiseks ning juhendaja(te) kinnitus lõputöö kaitsmisele lubamise kohta 108

TÄHISED JA LÜHENDID

bit	–	Arvsüsteemides väikseim ühik, mille saab olla kask väärtust, kas 1 või 0
CAT5e	–	Interneti kaabel
dec	–	Kümmnend süsteemi arv
DINT	–	Märgiga topelt reaalarv on muutuja tüüp, mis mahutab kuni 31 bitist märgiga reaalarvu
DWORD	–	Topelt arvsõna on muutuja tüüp, mis mahutab kuni 32 bitist märgita arvu.
FBD	–	Funktsiooniplokkide diagramm (*Function Block Diagram*), programmeerimise keel
IEC 60529	–	Seadmete ümbriste kaitse astme standard
IEC61131-3	–	Loogika kontrolleri programmeerimise standard
IL	–	Instruktsioonide loend (*Instruction List*), porgrammeerimise keel
Indra Logic	–	Tarkvara, kus luuakse loogikakontrollerile programm
Indralogic L10	–	Loogikakontroller
Indraworks Engineering	–	Tarkvara, kus luuakse projekt ja seadistatakse moodulid
INT	–	Reaalarv on muutuja tüüp, mis mahutab kuni 15 bitist märgiga arvu
L/min	–	Õhuvoolu ühik liitrit minutis
LD	–	Redel loogika (*Ladder Diagram*), programmeerimise keel
Modbus TCP	–	Andmeside protokoll loogikakontrolleir ja arvuti vaheliseks suhtluseks
PLC_PRG	–	Põhiprogrammi nimetus, mis peab olema igas loodud programmis.
POU	–	Programmi osa (*Program organization unit*)
Rasperry Pi	–	ARM protsessoril põhinev mikro arvuti
R-IB IL 24 DI 16-PAC	–	Diskreet sisend mooduli nimetus, 16 sisendit
R-IB IL 24 DO 16-PAC	–	Diskreet väljund mooduli nimetus, 16 väljundit
R-IB IL AI 2/SF-PAC	–	Analoog sisend mooduli nimetus, 2 sisendit
R-IB IL AI 8/SF-PAC	–	Analoog sisend mooduli nimetus, 8 sisendit
R-IB IL AO 1/SF-PAC	–	Analoog väljund mooduli nimetus, 1 väljund
R-IB IL SSI-IN-PAC	–	Enkooderi juhtimise mooduli nimetus, 1 enkooder
RJ-45	–	Interneti kaabli pistik ühenduse standard
RS-422	–	Andmeside protokoll, millel põhineb SSI
SCADA	–	Kaugjuhtimine ja andmehõive (*Spervisory Control And Data Aquasition System*)

SFC	–	Järjestik funktsioonide jada (*Sequential Function Chart*), porgrammeerimise keel
SINT	–	Märgiga sümbol on muutuja tüüp, mis mahutab kuni 7 bitist märgiga arvu.
SSI	–	Sünkroon andmeside protokoll (*Synchronous Serial Interface*)
ST	–	Kõrgtaseme programmeerimise keel (*Structured Text)*
TCP/IP	–	Interneti ühenduse protokoll
Technics 1	–	Esimese meeskonna/masina nimetus
Technics 2	–	Teise meeskonna/masina nimetus
Technics 3	–	Kolmanda meeskonna/masina nimetus
UDINT	–	Märgita topelt reaalarv on muutuja tüüp, mis mahutab kuni 32 bitist reaalarvu
UINT	–	Märgita reaalarv on muutuja tüüp, mis mahutab kuni 16 bitist märgita arvu
USINT	–	Märgita sümol on muutuja tüüp, mis mahutab kuni 8 bitist märgita arvu
VDC	–	Volti alalispinget
WORD	–	Arvsõna on muutuja tüüp, mis mahutab 16 bitist märgita arvu
WRT 1900 AC	–	Ruuter juhtmevaba andmehõive süsteemi loomiseks
P	–	Gaasi rõhk, Pa
V	–	Õhuvoolu hulk, $\frac{\mathrm{L}}{\mathrm{min}}$
n	–	Avijadri arv, $6{,}022 \cdot 10^{23}$
R	–	gaasikonstant, $8{,}314\ \mathrm{J} \cdot \mathrm{K}^{-1} \cdot \mathrm{mol}^{-1}$
V1	–	õhuvoolu hulk normaalliitrites, $\frac{\mathrm{m}^3}{\mathrm{s}}$
P2	–	tegelik rõhk, Pa
P1	–	atmosfääri rõhk, Pa
T1	–	väliskeskonna temperatuur, K
T2	–	mõõdetud gaasi temperatuur, K

SISSEJUHATUS

Käesoleva töö eesmärgiks on projekteerida pneumaatilise sõiduki juhtimise automaatika, mis vastab Ungaris toimuva võistluse *International Aventics Pneumobile Competition* reegilitele. Võistlus toimunud juba kaheksa aastat. Igal aastal on meeskonnad erinevatest riikidest projekteerinud ja valmis ehitanud rastastel liikuva sõiduki, mille peamiseks energiaallikaks on suruõhuga täidetud balloon.[1]

Suruõhuballooni maht on kümme liitirt ja õhusurve on 20 MPa (200 bar). Võistluse eesmärgiks on kasutada suruõhku kuluefektiivselt ja seejuures peab pneumaatilisel mootoril olema piisavalt võimsust. Pneumaatilises mootoris on suruõhupaagis olevat energiat liikumise energiaks muundavaks komponendiks pneumaatiline silinder. Kolvi liikumisega muundatakse pneumaatilises silindris balloonis salvestatud suruõhu potentsiaalne energia kineetiliseks energiaks.

.

Projekteeritud ja valmis ehituatud automaatika ja andmehõive lahendus katsetati ja kasutati suruõhu jõul liikuval sõidukil, mis ehitati valmis 2016. aasta võistluseks. Võistlusel võisteldakse kolmel erineval alal, milleks on: kestvus-, ringraja- ja kiirenduse sõit. Kestvussõidu eesmärgiks on läbia võimalikult pikk distants minimaalse energiakuluga, seejuures ei tohi keskmine kiirus langeda alla lubatud piirmäära. Ringraja sõidu eesmärgiks on saavutada ringide läbimisel parim aeg. Kiirenduse sõidu eesmärgiks on läbida distants kiireima ajaga. [1]

Eraldi kategooriasse kuulub andmehõivesüsteem. Andmehõive eesmärgiks on sõidu ajal anduritelt saadud informatsiooni salvestamine. Andmehõive süsteem töötab üle juhtmevaba andmeside võrgu reaalajas ja saadud informatsioon salvestatakse andmebaasi. Salvestatud väärtused peavad olema *Microsoft Office Excel* failis reglemendis määratud kujul. [1]

Projekteerimise aluseks on eelnevatel aastatel valmistatud suruõhu jõul liikuvad sõidukid. 2015. aastal ehitatud pneumosõiduki mootor on katsestendis uuritud ning saadud andmete põhjal koostati 2016. aasta masinale suruõhu säästlikum juhtimisautomaatika.

Töös kirjeldatakse automaatika- ja andmehõivesüsteemis kasutatavaid seadmeid. Pneumaatika juhtimise seadmed on tagatud korraldajate poolt ning masina ehituseks vajalikud vahendid

tuleb võistkondadel ise leida. Automaatjuhtimise kirjelduses on välja toodud komponentide loetelu ja kirjeldus tööprotsesside kohta.

Pneumosõiduki automaatika- ja andmehõivesüsteemi projekteerimiseks ja valmis ehitamiseks seati eesmärgiks täita järgnevad ülesanded:

1. Projekteerida pneumosõiduki automaatika vastavalt võistluse reeglitele.
2. Valida pneumosõiduki automaatjuhtimiseks sobivad seadmed vastavalt võistluse reeglitele.
3. Luua matemaatilised funktsioonid masina rõhu ja õhuvoolu reguleerimiseks.
4. Leida parim võimalik meetod masina kiiruse automaatseks reguleerimiseks.
5. Koostada juhtprogramm, mis juhiks masina mootorit suruõhku optimaalselt kulutades.
6. Luua SCADA pneumosõiduki andurite väärtuste jälgimiseks ja täiturite juhtimiseks.

1. PROJEKTEERIMISE TINGIMUSED JA SEADMETE VALIK

1.1 Süsteemi kirjeldus

Automaatjuhtimise ja andmehõive projekteerimise aluseks on võistluskorraldajate poolt paika pandud reeglid [1]:

1. Soovitatav toitepinge on 24 VDC ja maksimaalne lubatud toitepinge on 48 VDC.
2. Elektriseadmed peavad olema kaitstud ilmastikuolude eest. Kaitseklass ei tohi olla väiksem, kui IP54 lähtudes standardist IEC 60529.
3. Automaatjuhtimine võib töötada kas ise tehtud mikrokontrolleriga või tööstusliku loogikakontrolleriga. Kui meeskond otsustab kasutada tööstuslikku loogikakontrollerit, siis on lubatud kasutada ainult IndraControl L10 või IndraLogic L10 loogikakontrollerit.
4. Andmehõivesüsteem peab töötama reaalajas. Graafiliselt liideselt peavad olema loetavad kolm graafikut, milleks on: kolvi liikumise aegdiagramm, kolvi positsiooni muutuse jõu diagramm, masina kiirus, kolvi liikumise kiirus, õhu kulu diagramm. Andmed peavad olema salvestatud *Microsoft Office 2007 Excel faili formaadis* (.xls).
5. Kõik pneumaatika komponendid: andurid, silindrid, pneumojaotid, liitmikud ja torud peavad olema ettevõte *Bosch Rexroth* toodang.

Täituriteks on süsteemis: pneumaatilised klapid, samm-mootor, rõhuregulaator. Pneumaatiliste klappide eesmärgiks on lubada ja piirata silindrisse sisenevat ja silindrist väljuvat suruõhku. Ümberlülitused on elektrilised. Samm-mootor reguleerib sujuvalt muutuvat ülekande suhet. Elektroonilise rõhu regulaatoriga reguleeritakse mootori töörõhku.

Andurid on vajalikud süsteemis toimuvate muutuste kohta tagasiside saamiseks. Meeskonnal *Technics 1* on rõhuandurid paigutatud kahelepoole mootorit pneumo jaotite peaõhuvoolu sisendisse, et salvestada rõhu väärtuseid reaalajas kogu sõidu jooksul. Rõhu väärtuste salvestamine võimaldab hilisemalt analüüsida koostatud süsteemi efektiivsust ja seeläbi muuta automaatjuhtimist kuluefektiivsemaks. Absoluutenkooderit on masinal tarvis, et mootori võllinurk oleks pidevalt teada. Vastavalt võlli pöördenurgale juhib programm klappide lülitusi. Rõhu regulaatoriga reguleeritakse masina töörõhku.

Täiturite ja andurite tööd juhib programmeeritav loogikakontroller. Programmeeritav loogika-kontroller on seade, millel on sisendid ja väljundid. Sisendid ja väljundid töötavad vastavalt

kontrollerisse salvestatatud programmile. Programmjuhtimine võimaldab süsteemi automatiseerida. Automatiseeritud süsteem tagab kuluefektiivsema õhu kasutuse süsteemis.

Telemeetria süsteem võimaldab andmeid koguda ja masinat distantsjuhtida. Telemeetria süsteem on valmis tehtud *SCADA* tarkvaraga *Ignition*, mille tootjaks on Ameerika ettevõte *Inductive Automation*. Reaalajas andmevahetus töötab *WiFi* vabaleviala võrgu protokolliga.

1.2 Pneumosõiduki automaatikas ja andmehõives kasutatavad seadmed

Pneumaatika komponendid on tagatud võistluse peasponsori *Bosch Rexroth* ettevõtte poolt. Sponsoreeritud komponentide hulka kuuluvad pneumosüsteemi aktuaatorid ja andurid. Sellest tulenevalt on andurite ja aktuaatorite kasutamine pneumosõidukis piiratud, tohib kasutada vaid *Bosch Rexroth* toodangut.

Telemeetria süsteemi loomiseks *Bosch Rexroth* vajalikke komponente ei tooda. Andmeside ühenduseks vajalikud seadmed tuleb meeskondadel valida ja soetada ise. Telemeetriaga masinat projekteerides on otstarbekas kasutada masina juhtimiseks tööstusliku loogikakontrollerit. Seda põhjusel, et kui kasutada mõnd vabavaraliselt kättesaadavat *SCADA* tarkvara graafilise liidese loomiseks, saab ühenduse tarkvara ja loogikakontrolleri vahel luua kasutades tööstuslike andmeside protokolle nagu Modbus.

Kõrvalise elektroonika kasutamine abistavas süsteemis ei ole piiratud. Näiteks projekteeriti õhuvoolu kiiruse piiramiseks elektrooniline klapp. Valmistamiseks kasutati Bosch Rexroth ettevõtte poolt toodetud ventiili, mis on ettenähtud töötama suruõhu süsteemides ja servo-mootorit, mida kasutatakse hobilennukite ehitusel. 2015. aastal oli kasutusel analoogne lahendus, kus õhuvoolu piiravaks komponendiks oli pneumosüsteemi drossel, mis oli ühendatud alalisvoolu mootoriga. Kohtunikud reeglitega vastuolu taolise lahenduse puhul ei leidnud.

Tabel 1.1. *Indralogic L10* parameetrid [2]

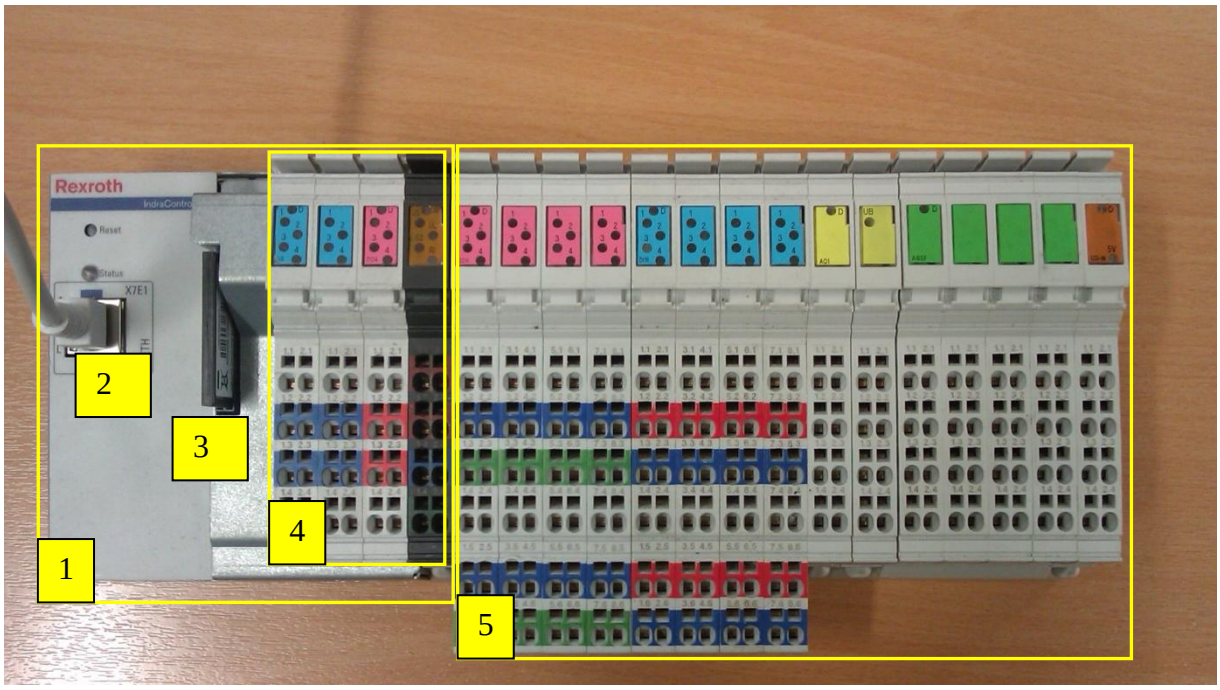
Tootekood	Nimetus	Märkused
R412010773	Rõhuandur	
R412010674	Õhuvooluandur	
3520323500	Ventiil	
5772080220	Pneumojaoti	Väline pilootõhk
R414002411	Elektrooniline rõhuregulaator	
R911170784	Analoogsisendi moodul	1 sisend
R911308493	Analoogsisendite moodul	8 sisendit
R911170787	Analoogväljundi moodul	1 väljund
R911170752	Diskreetsisendite moodul	16 sisendit
R911170757	Diskreetväljundite moodul	16 väljundit
R911171514	SSI moodul	Absoluutenkooderi juhtimine
R911170722	*Indralogic L10* loogikakontroller	Pneumosõiduki loogikakontroller
	Ubiquiti Nanostation Loco M2	WiFi kaugühenduse antenn koos ruuteriga
WRT 1900 AC	*Linksys* ruuter	WiFi ruuter pneumosõidukis

Tabelis toodud tooteid kasutatakse pneumosõiduki juhtimise automaatika valmis ehitamisel.

2. PNEUMOSÕIDUKI AUTOMAATIKA KIRJELDUS

2.1 Programmeeritav loogikakontroller

Pneumosõidukit juhib tööstuslik loogikakontroller (joonis 2.1). Tööstuslik loogikakontroller, mida kasutatakse on *Indralogic L10*. Võistlusreglemendis on määratud, kui võistlusmasin kasutab tööstuslikku loogikakontrollerit, peab see olema *Bosch Rexroth* loogikakontroller *Indralogic L10*. Kui tööstuslikku loogikakontrollerit ei kasutata võib kasutada mõne muu tootja mikrokontrollerit. Võistlusel võis näha, et levinud mikrokontroller juhtimiseks on *STMicroelectronics*. *STMicroelectronics* mikrokontrollerid töötavad *ARM* mikroprotsessoriga.



Joonis 2.1. Tööstuslik loogikakontroller moodulitega: 1 − Tööstuslik loogikakontroller; 2 − Võrguliides seadme programmeerimiseks ja võrku ühendamiseks 1 tk.; 3 − Mälukaart programmi salvestamiseks 1 tk.; 4 − Tööstusliku loogika kontrolleri sisendid ja väljundid: 8 diskreetsisendit, 4 diskreetväljundit; 5 − Lisamoodulid: 16 diskreetsisendit, 16 diskreetväljundit, 1 analoogväljund, 8 analoogsisendit, 1 SSI sisend

Loogikakontrolleri parameetritest (tabel 2.1) kõige olulisem on tsükli aeg. Tsükli aeg näitab mikroprotsessori arvutuskiirust programmi ühe tsükli läbimisel. Mida väiksem on tsükliaeg, seda vähem kulub mikroprotsessoril aega programmi läbitöötamisel.

Diskreetsisendite kiirus on parameeter, mis määrab suurima lülitus taktsageduse. Kui süsteemis on andur, mis edastab oma olekut loogikakontrollerile kiirusega 300 kHz ja digitaalsisendi maksimaalne lubatud taktsagedus on 100 kHz, siis erineuvsest tulenevalt loendatakse kolm korda vähem anduri lugemeid kui andur edastab. Pneumosõiduki projektis on tarvis kiiret sisendit, et loendada pöördenkooderi signaale. Loogikakontrolleril on ka kiired väljundid, mille lülituse aeg on 500 µs, aga kuna pneumojaotite asendi muutmise kiireim aeg on 100 ms, siis kiireid digitaalseid väljundeid tarvis kasutada ei ole.

Tabel 2.1. *Indralogic L10* parameetrid [3]

Nimetus	**Kirjeldus**	**Väärtus**
Mälu maht, MB	Programmi kood	4
Võrguliides (*TCP/IP*)	-	1
Kiire diksreet sisend	Katkestuste programmeerimise võimalus, kiirus: 50 µs	8
Kiire diskreet väljund	0,5 A, 500 µs	4
Indralogic 1G operatsioonisüsteem	Ühildub programmeerimise standardiga IEC61131-3	
Tsükliaeg, 1000 instruktsiooni, µs	Erinevad ülesanded	150
	Boole'i loogikaülesanded	150
	Arvsõnadega ülesanded	150
Toitepinge, VDC	-	24

Loogikakontrolleri programmeerimine toimub läbi võrguühenduse. Võrguühendus luuakse kasutades *RJ-45* pistikutega võrgukaablit. Võrgukaabliks võib kasutada näiteks standardile *CAT5e* vastavat kaablit. Ühendades kaabliga loogikakontrolleri ja arvuti saab luua programmeerimiseks ühenduse loogikakontrolleri ja arvuti vahel. Kuna loogikakontrollerit *Indralogic L10* on võimalik ühendada võrku, ei pea programmeerimiseks mõeldud arvuti ja loogikakontroller olema samas ruumis, vaid samas võrguühenduses.

Võrguühenduse kaudu on võimalik loogikakontrollerist *Modbus TCP* suhtlusprotokolli kasutades edastada muutujate väärtuseid. Muutujate väärtuseid saab edastada kas järgmisele loogikakontrolleritele või *SCADA* rakendusele.

Loogikakontrollerile saab lisada ka lisamooduleid. Algselt on loogikakontrolleril piiratud hulk sisendeid ja väljundeid. *Indralogic L10* loogikakontrolleril on 8 diskreetset sisendit ja 4 diskreetset väljundit. Pneumosõidukil on tarvis vähemalt üht analoogsisendit ja -väljundit ning sõltuvalt nuppude, LED valgustite, pneumojaotite ja muude seadmete arvust digitaalseid sisendeid ja väljundeid. Kui loogikakontrolleri küljes olevatest sisenditest ja väljunditest ei piisa on tarvis lisamooduleid (tabel 2.2).

Tabel 2.2. Pneumosõidukitel kasutatavad lisamoodulid [4]

Nimetus	**Kirjeldus**	**Kogus**	**Nr. joonisel**
R-IB IL 24 DO 16-PAC	Diskreetväljundite moodul	16	2
R-IB IL 24 DI 16-PAC	Diskreetsisendite moodul	16	3
R-IB IL AI 8/SF-PAC	Analoogsisendite moodul	8	5
R-IB IL AI 2/SF-PAC	Analoogsisendite moodul	2	puudub
R-IB IL AO 1/SF-PAC	Analoogsisendite moodul	1	4
R-IB IL SSI-IN-PAC	SSI sisendi moodul	1	6

Diskreetväljundite moodulil on 16 digitaalsest väljundit. Diskreetse väljundmooduliga lülitatakse seadmeid, mille toitepingeks on +24 VDC. Voolutarbimine võib maksimaalselt olla 0,5 A. Seadmete ühendamine toimub kas kahe- või kolmejuhtmeliselt. Täitureid on süsteemi ühendatud kahe juhtmega, kuna neil puudub tagasiside.

Diskreetsete sisendite moodul on mõeldud mõõtma pinge nivood. Kui sisendisse lülitada pinge +24 VDC kirjutatakse loogikakontrolleri registrisse loogiline 1 ja kui pinge sisendis on 0 VDC kirjutatakse loogiline 0. Sisendmoodulil on vastavad klemmid, mis väljastavad andurile toitepingeks mõeldud +24 VDC.

Analoogsisendite moodulit (*R-IB IL AI 8/SF-PAC*) on tarvis andurite pinge või voolu signaali mõõtmiseks. *Indralogic L10* analoog sisend moodulit saab seadistada mõõtma nii voolusignaali, kui pingesignaali (tabel 2.3).

Tabel 2.3. *R-IB IL AI 8/SF-PAC* mõõtepiirkonnad [5]

Voolusignaali mõõtepiirkonnad	Pingesignaali mõõtepiirkonnad
0 ... 20 mA	0 ... 10 V
±20 mA	±10 V
4 ... 20 mA	0 ... 5 V
0 ... 40 mA	±5 V
±40 mA	0 ... 25 V
	±25 V

Analoogsisend moodulit on pneumosõiduki automaatjuhtimises tarvis, et kvantiseerida andurite pinge või voolu signaalid numbrilisele kujule, mille väärtust on võimalik loogika kontrolleri programmi lugeda. Analoogsisned moodul töötab kui analoog-digitaal muundur.

Pneumosõiduki automaatjuhtimiseks kasutatakse rõhu anduritel, õhuvooluanduril ja reguleeritaval takistil voolusignaali mõõtepiirkonda (tabel 2.4) 4 mA ... 20 mA. Selle mõõtepiirkonna kasutamine aitab tuvastada võimalikke probleeme anduri töös.

Tabel 2.4. Analoogsisendi voolusignaal [6]

Konrollerile edastatav detsimaalväärtus	0 ... 20 mA hetkväärtus	Konrollerile edastatav detsimaalväärtus	4 ... 20 mA hetkväärtus
dec		**dec**	
OR	> +21,6746	OR	> + 21,339733
32512	+21,6746	32512	+ 21,339733
30000	+20,0	30000	+ 20,0
1	+ 0,66667 µA	1	+ 4,0005333
0	0	0	+ 4,0 - 3,2
0	< 0	Avatud vooluring	< + 3,2

Kui anduri tagasisides on vool liialt kõrge väljastatakse kontrollerile arvsõna, mis on veateade. Kui mõõdetud voolusignaal on alla 4 mA väljastatakse samuti kontrollerile arvsõna, mis on veateade. Rõhu reguleerimine töötab voolusignaaliga 0 mA ... 20 mA. Selles mõõtepiirkonnas saab tuvastada viga vaid siis, kui seadistatud vool on üle 20 mA. Eelistatud

mõõtepiirkonna valik pneumosõiduki automaat süsteemis on 4 mA ... 20 mA. Selles piirkonnas on võimalik kontrollida automaatika seadme korrasolekut.

Analoogsisendmooduli mõõdetud tulemus esitatakse loogikakontrollerile numbrilise väärtusena. Pinge või voolusignaalist numbriline väärtus saadakse analoogmoodulis olevast analoog-digitaal muundurist. *Bosch Resxroth* analoog sisned mooduli muundur ühlasi muundab analoog signaali ja seejuures ka stabiliseerib lõpptulemust. Moodulit seadistades on võimalik saada kuni 32 bitine muundamine.

Numbrilised väärtused loetakse kontrolleri registritesse 16 bitise numbrilise väärtusena. väärtuse 16. bit on eraldatud märgi jaoks. kui bit 16 on kõrge on arv negatiivne ja kui madal, siis positiivne. Seetõttu saab registrisse kirjutatud väärtus olla 0 ... 32767 (suurim 15 bit väärtus).

Analoogväljundite moodul on pneumosõiduki automaatikas mõeldud analoog pinge- või voolusignaali väljastamiseks (tabel 2.5). Analoog voolu signaali on tarvis rõhu reguleerimises. Analoogväljund moodul töötab vastupidiselt analoog sisend moodulile. Kui analoog sisend moodul kvantiseeris pinge- või voolusignaali numbriliseks väärtuseks, siis analoogväljundite moodul väljastab pinge- või voolusignaali vastavalt numbrilisele väärtusele.

Tabel 2.5. Analoogväljund signaal [7]

Numbriline väärtus	**Pinge väljund 0 ... 10 V**	**Voolu väljund 0 ... 20 mA**
dec	**hetkväärtus**	**hetkväärtus**
65535	9,99985	19,9997
65534	9,99969	19,9994
32768	5,0000	10,000
1	0,153 mV	0,305 µA
0	0,0000	0,0000

SSI (*Syncronus Serial Interface)* sisendmoodulit kasutatakse pneumosõiduki automaatjuhtimises absoluutenkooderi nurga väärtuse loogikakontrollerile loetavaks muundamisel. SSI on tööstuses laialdaselt kasutatav suhtlusprotokoll, kus Isand seade suhtleb

ori seadmega. Tavaliselt on isand seadmeks loogikakontroller ja ori seadmeks andur. Suhtlus põhineb RS-422 protokoll, mis põhineb andmeside standarditel.

Bosch Rexroth SSI moodul on mõeldud absoluutenkooderite nurga väärtuste lugemiseks. Mooduliga saab kasutada enkoodereid, mille maksimaalne lubatud bitide arv on 25 (tabel 2.6). Bittide arv näitab, kui suur on pöördenkooderi resolutsioon ühel pöördel. Pneumosõidukis kasutatakse enkooderit, mille ühe pöörde resolutsioon on 12 bitti.

Tabel 2.6. SSI mooduli andmed [8]

Parameeter	Väärtus	Märkus
Ühe- või mitmeöördeline lugemine		Võimaldab kasutada enkooderit mis teab oma asendit mitme pöörde ulatuses
Resolutsioon	8 ... 25 bit	Näitab enkooderi ühe pöörde resolutsiooni.
Gray- ja binaarkood		Võimalik seadistada SSI moodul lugema Gray või binaarkoodi
Andmeedastus kiirus	100 kHz, 200 kHz, 400 kHz, 800 kHz, 1 MHz	Maksimaalne enkooderi võlli pöörlemise sagedus.
Toitepinge enkooderile	5 V	
Veatuvastus	Diagnostika ja oleku indikaator LED	
Toitepinge moodulile	+24 VDC	
Kaablile lubatud maksimaalne pikkus	< 30 m varjestatud kaabel.	

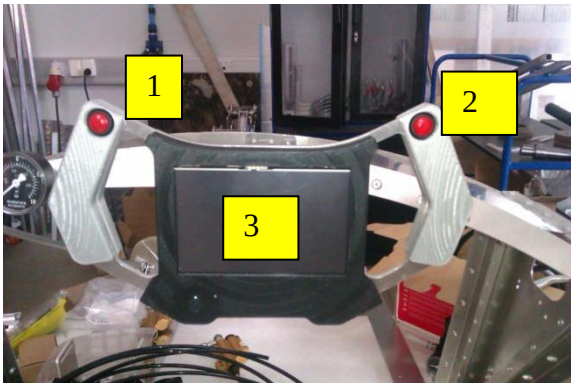
Moodulil on ühendamiseks 8 klemmi. Ideaalolukorras on kõik kasutuses, kuid kuna pneumosõidukil puudub varje jaoks vajalik maandus, siis reaaselt kasutatakse vaid 6 ühendust.

2.2 Arvuti abiseadmete juhtmiseks

Pneumosõiduki automaatjuhtimise süsteemis kasutatakse lisaks tööstuslikule loogika-kontrollerile ka arvutit *Rasberry Pi*. *Rasberry Pi* on pneumosõiduki automaatjuhtimiseks tarvilik lisafunktsioonide talitluses. Lisafunktsioonidena paigaldatakse pneumosõidukile sõitjale informatsiooni kuvamise puutetundlik vedelkristallekraan. Lisaks puutetundlikule vedelkristall ekraanile kasutatakse lisa arvutit servomootori juhtimiseks, mis on kasutusel käiguvahetus süsteemi elemendina ja muudab sujuvalt ülekandesuhet.

Sõitjale informatsiooni kuvamise seadmeks on pneumosõiduki rooli külge kinnitatud 5-tolline puutetundlik vedelkristallekraan. Ekraanil kuvatakse sõitjale:

1. Rõhu seadeväärtus.
2. Rõhu hetkeväärtus.
3. Masina liikumise kiirus.
4. Mootori väljundvõlli pöörlemise kiirus.
5. Pedaali asend (0 ... 100%).
6. Mootori töörežiim.
7. Õhukulu.



Joonis 2.7. pneumosõiduki rool: 1 − signaali nupp; 2 − püsikiiruse hoidmise nupp.; 3 − ekraan sõitjale informatsiooni kuvamiseks.

Lisaks ekraanile on rooli küljes: sõidurežiimi valiku, helisignaali ja püsikiiruse hoidmise nupp (joonis 2.7).

2.3 Pneumosüsteemi andurid

Pneumosõiduki automaatjuhtimise loomiseks on tarvis mõõta pneumosüsteemi ja mehhanismide liikumise parameetreid. Pneumosüsteemi põhilisteks parameetriteks on rõhk ja õhuvoolukiirus. Mehhanismide põhilisteks parameetriteks on: kolbi liikumise kiirus, masina liikumise kiirus, mootori väljundvõlli pöörlemise kiirus, mootori väljundvõlli pöördenurga väärtus.

Parameetrite mõõtmine toimub anduritega. Pneumosõiduki automaatjuhtimises kasutatakse järgmisi andureid:

1. Rõhuandur.
2. Absoluutenkooder.
3. Inkrementaalenkooder.
4. Pneumosilindri kolvi positsiooniandur.
5. Keelrelee.

Rõhku mõõdetakse süsteemis kahel meetodil. Esimeseks meetodiks on rõhu mõõtmine rõhuanduriga, mis paikneb pneumojaotite õhu pealevoolu sisendis ja teiseks meetodiks on rõhu regulaatorilt regulaatori tagasisidena. Rõhu regulaatori tagasiside on mõeldud reguleeritud rõhu ja tegeliku rõhu võrdlemiseks. Ideaalolukorras on mõõdetud ja reguleeritud rõhk võrdsed. Rõhuregulaatoris mõõdetud rõhk aitab tuvastada kui põhipaagis olev rõhk ei ole piisav, et hoida süsteemis reguleeritud rõhku.

Rõhu anduri funktsioone saab pneumosõiduki automaatjuhtimise süsteemis kasutada mitmel viisil. Ühel võistluse masinatest kasutatakse rõhu andurit telemeetria süsteemi nõuete täitmiseks ja teisel masinal kasutatakse rõhu andurit jääkõhkude tagasiside süsteemis.

Õhukulu mõõtmine pnemosõidukis on tarvilik telemeetria kategoorias võisteldes. Reeglite kohaselt on tarvis telemeetria süsteemis kuvada õhukulu diagrammi ajas. Õhukulu mootori juhtimisel mõju ei avalda.

Pneumosõiduki mootori väljundvõlli asendit on tarvis positsioneerida ja see on vajalik pneumaatiliste jaoturite lülitamiseks. Vastavalt võlli pöördenurgale nullpunkti suhtes lülitatakse pneumojaoteid selliselt, et mootori võll hakkaks pöörlema. Mootori väljundvõlli

asendi positsioneerimiseks kasutatakse absoluutenkooderit. Positsioneerimiseks on tarvis absoluutenkooderit, kuna puudub võimalus enne esmast mootori käivitamist väljundvõlli nurga asend nullida. Absoluutenkooder teab alati oma asendit ühe pöörde ulatuses.

Kolvi positsiooni andurit (joonis 2.8) kasutatakse süsteemis kolvi kiiruse mõõtmiseks. Lineaarne positsiooniandur teab silindris oleva kolvi asukohta kogu käigu ulatuses. Kolvi positsioon edastatakse kontrollerile analoogsignaalga (0 … 20 mA). [9]

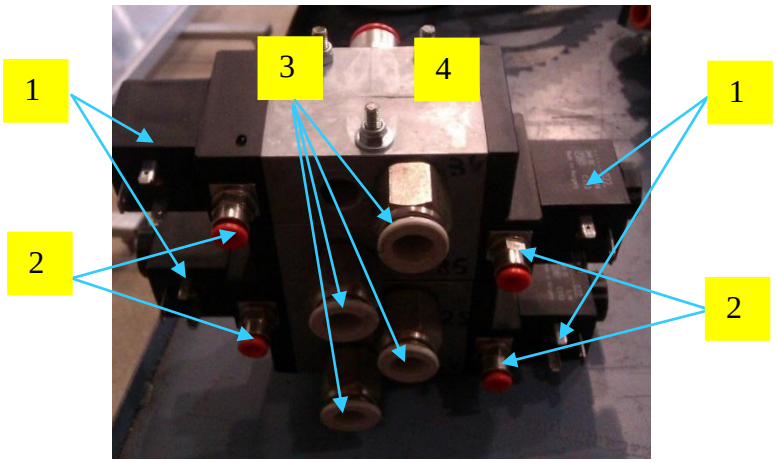
Joonis 2.8. Lineaarne kolbi positsiooni andur: 1 − oleku indikaator; 2 − nupp anduri seadistamiseks..

Positsiooni informatsioon jääb alles ka pärast toitepinge kadumist.

2.4 Pneumosüsteemi täiturid

Suruõhu liikumist vahepaagi ja silindri vahel piiratakse penumojaotitega (joonis 2.8). *Technics 1* pneumosõiduki juhtimiseks kasutatakse 3/2 pneumojaotit. Number 3 määrab, et jaotil on kokku kolm kanalit õhu juhtimiseks ja number 2 näitab, et klapil on kaks asendit. Klapi asendi muutmine toimub elektriliselt ja tagastamine vedruga. Elektriliseks juhtimiseks tuleb jaoti solenoidi pingestada +24 VDC. Klapi asendi muutmiseks on tarvis ka pilootõhku. Pilootõhk on vajalik, et solenoidi pingestamisel klapp muudaks asendit ka juhul, kui klapi peaõhuvool puudub.

Technics 1 mootori tüüp on radiaalmootor. Radiaalmootori puhul on tarvis, et silindritesse õhu suunamine toimuks mõlemast otsast eraldiseisvalt. Kui näiteks kaks silindrit töötavad, siis ülejäänud kaks peavad liikuma vabalt. 3/2 klapp võimaldab õhuvoolu juhtida sedasi, et ühes asendis liigub õhk silindrisse ja teises asendis lastakse silindris olev õhk atmosfääri ja õhukanal jääb avatuks.



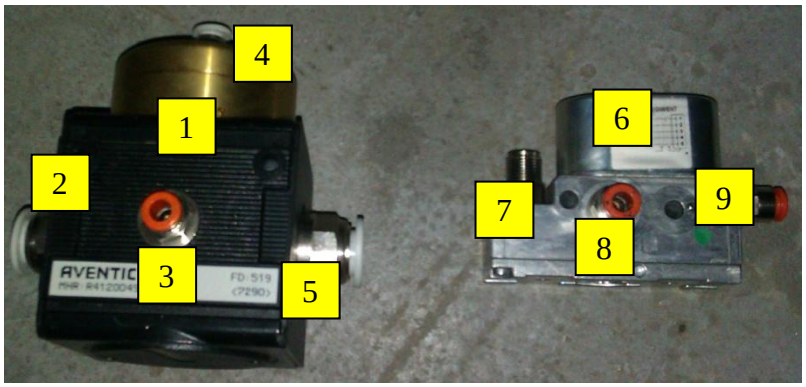
Joonis 2.8. Pneumojaotid: 1 − Solenoid pneumojaoti lülitamiseks; 2 − Pilootõhu nippel; 3 − Nippel õhuvooluks silindrisse; 4 − Pneumojaotite plokk, 4 tk.

Pneumojaotite olulisemateks parameetriteks on:

1. Suruõhuläbilaske võime, L/min.
2. Lülituse tüüp (3/2, 5/2, 5/3).
3. Lülitamine pilootõhu abil või ilma pilootõhuta.
4. Sisse- ja väljalülitamise aeg.
5. Töörõhk.

Mootorisse mineva suruõhu rõhu muutmiseks kasutatakse pneumosõiduki automaatjuhtimises elektroonilist rõhuregulaatorit (joonis 2.9). Kuna elektroonilise rõhuregulaatori suruõhu läbilaske võime on väike kasutatakse rõhureguleerimise süsteemis ka pneumaatiliselt juhitavat rõhuregulaatorit.

Elektrooniline rõhuregulaator töötab koostöös pneumaatilise rõhuregulaatoriga. Pneumaatilises rõhuregulaatoris reguleeritakse rõhku proportsionaalselt reguleerivasse sisendisse lastava rõhuga.



Joonis 2.9. Elektrooniline rõhuregulaator ja andur: 1 − Pneumaatiliselt juhitav rõhu regulaator; 2 − Suruõhk mootorisse; 3 − Manomeeter hetkerõhu manuaalseks kuvamiseks; 4 − Suruõhu sisselaske nippel rõhu reguleerimiseks; 5 − Süsteemi suruõhu pealevool; 6 − Elektrooniline rõhuregulaator; 7 − Isane pistik signaalikaabli ühendamiseks; 8 − Ühendatakse nippliga 4; 9 − ühendatakse süsteemi suruõhu pealevooluga

Rõhuregulaator töötab +24 VDC pingega ja reguleerimiseks on tarvis loogikakontrollerilt analoogväljund signaali. Reguleerimise signaali vahemik on 4 ... 20 mA. Elektroonilisel rõhuregulaatoril on rõhu hetkeoleku tagasiside, mis on samuti 4 ... 20 mA analoogsignaal.

3. PNEUMOSÕIDUKI AUTOMAATJUHTIMINE

3.1 Programmeerimise keeled

Tööstuslike loogikakontrollerite programmeerimine vastab IEC-61131 standardile. *Indracontrol L10* kontrolleri programmerimine käib *Bosch Rexroth* tarkvara *Indralogic* keskonnas. *Indralogic* keskonnas saab luua uut programmi ja simuleerida lõpptulemust.

Indralogic programmeerimise keskonnas saab programmi luua viies erinevas programmeerimise keeles, mis vastavad standardile IEC-61131[14]:

1. Instruktsioonide loend (*Instruction List, IL).*
2. Järjestikfunktsioonide diagramm (*Sequential Function Chart, SFC*).
3. Funktsiooniplokkide diagramm (*Function Block Diagram, FBD*).
4. Redelloogika diagramm (*Ladder Diagram, LD*).
5. Kõrgetaseme programmeerimise keel (*Structured Text, ST).*

Instruktsiooni loend on programmeerimise keel, kus porgrammeerimine toimub järjestikku asetatud käskudega, mis piltlikult moodustavad loendi. Instruktsiooni loend on sarnane APJ tööpinkide programmeerimise keelega, kus käske täidetakse rida rea haaval.

Järjestikfunktsioonide diagramm on programmeerimise keel, kus programmi voogu kontrollitakse järjestikku paigutatud kastidega. Igale kastile tuleb kirjutada omakorda vastav programm, mida ühes või teises kastis täidetakse. Kastide veerge on võimalik lisada, luues sellega programm, kus täidetakse mitut ülessannet korraga. Eraldi saab tingimusi luua järgmisesse kasti liikumise piiramiseks.

Funktsiooni plokkide diagramm on programmeerimise keel, kus saab luua lihtsamat ülessannet täitvat programmi. Funktsiooniplokkide diagramm keelt kasutades saab luua programmi, kus saab kasutada vaid booli loogika elemente. Selles programmeerimise keeles on otstarbekas luua programm, kui süsteemis on kasutusel vaid elemendid, mis töötavad diskreetväärtustel.

Redeldiagrammi programmeerimise keele struktuur meenutab elektrimootorite juhtimiseks mõeldud juhtahelate skeemi, kus juhtskeemi elementide paigutus skeemis moodustab redeli. Loogikakontroller täidab iga programmi rida korraga.

Pneumosõiduki programmeerimiseks kasutatakse kõrgtaseme programmeerimise keelt, *ST* (*Structured Text). ST* keele funktsioonid, muutujad, konstandid, operaatorid, andmetüübid (tabel 3.1) sarnanevad programmeerimiskeelega *Pascal*.

Tabel 3.1. Kõrgtaseme programmeerimise keele operaatorid ja funktsioonid [10]

Nimetus	**Sümbol**	**Näited**
Funktsiooni kutsumine	Funktsiooni nimi (parameetrid)	-
Eksponent	*EXPT*	*EXPT(5)*
Negatiivne number	-	*-5; -1*
Eitus	*NOT*	*Var1 := NOT 2#10001_0011, tulemus: 2#0110_1100*
Korrutamine	*	*5*5; 2*2*
Jagamine	/	*5/5; 2/2*
Jagamise jääk	*MOD*	*9 MOD 2, tulemus: 1*
Liitmine	+	*5+5; 2+2*
Lahutamine	-	*5-5; 2-2*
Võrdlemine	<, >, <=, >=	*A < B; C > A; D <= B; F >= A*
Võrdne	=	*A = C*
Ei ole võrdne	<>	*A <> B*
Booli, ja	*AND*	IF A > B AND C < D THEN
Booli, välistav või	*XOR*	IF A > B XOR C < D THEN
Booli, või	*OR*	IF A > B OR C < D THEN
Omistamine	*A:=B; CV:=CV+1; C:=SIN(X);*	
Funktsiooni kutsumine	*FUNC(IN1:=1, IN2:=3);*	
Funktsiooni muutujate lugemine	*FUNC.OUT1;*	
Tagasi algusesse	*RETURN;*	RETURN;

Tabel 3.1. Kõrgtaseme programmeerimise keele operaatorid ja funktsioonid (*järg*)

Nimetus	**Sümbol**	**Näited**

IF funktsioon	*IF*
	ELSIF
	ELSE
		IF D<0 THEN
		C:=A;
		ELSIF D=0 THEN
		C:=B;
		ELSE
		C:=D;
		END_IF;

CASE, programmi voo kontrollimine		*CASE INT1 OF* *1: BOOL1 := TRUE;* *2: BOOL2 := TRUE;* *ELSE* *BOOL1 := FALSE;* *BOOL2 := FALSE;* *END_CASE;*

FOR tsükkel		*FOR I:=1 TO 100 BY 2 DO*
		IF ARR[I]=70 THEN
		J:=I;
		EXIT;
		END_IF;
		END_FOR;

WHILE tsükkel		*WHILE J<=100 AND ARR[J] <> 70 DO* *J:=J+2;* *END_WHILE;*
REPEAT tsükkel		*J:=1* *REPEAT* *J:=J+2;* *UNTIL J=101 OR ARR[J] = 70* *END_REPEAT;*
Väljumine	EXIT;	*EXIT;*

Programmi loomine algab *Indraworks Engineering* keskonnas. *Indraworks Engineering* keskonnas saab alguse projekti loomine. Projektile valitakse nimi ja savlestamise asukoht. Kui uus projekt on loodud tuleb valida programmeeritav loogikakontroller, mis on aluseks kogu programmi loomisele. Pneumosõiduki projekti loomiseks tuleb valida *Indra Logic L10.* Valitud loogika kontrolleri seadistustes tuleb valida õige püsivara (*firmware)* versioon ja määrata seadme *IP* aadress. Kui loogikakontroller on seadistatud korrektselt on võimalik minna *online* olekusse, kus luuakse ühendus arvuti ja loogika kontrolleri vahel.

Pneumosõiduki projektis on alati kasutuses vähemalt 1 lisamoodul. *Technics 1* meeskonna masinal on moodulite arv kokku 5. Algselt loodud projektis moodulid puuduvad ja need tuleb lisada manuaalselt. Moodulid on andmebaasis vastavalt mooduli nimetusele (tabel 3.1). Kui lisamoodulid on projekti lisatud määratakse automaatselt moodulite seadistamiseks mõeldud sisend- ja väljundregistrid. Näiteks on analoog mooduli seadistamiseks kaks sisendregistrit ja kaks väljundregistrit.

Kui moodulid on projekti lisatud ja projekt programmeerimiseks seadistatud, siis võib avada *Indra Logic* keskonna, kus alustatakse juhtprogrammi kirjutamist. Keskkonda sisenedes luuakse alati esimene *POU (Program Organization Units),* mille nimeks on *PLC_PRG.* Väljaarvatud *PLC_PRG* saab iga *POU* olla eelpool kirjeldatud viie programmeerimise keele seadega.

PLC_PRG on loogikakontrolleri programmis põhiprogramm. Selle *POU* kaudu sisenetakse loodud funktsioonidesse ja funktsiooni plokidesse. Funktsioon on *POU* tüüp, mille läbimisel alati tagastakse määratud andmetüübis väärtus. Näiteks, kui luuakse funktsioon *f* ja andmetüübiks (tabel 3.2) on määratud *INT* (reaalarv), siis funktsioonis tehtavate arvutuste lõppedes peab funktsioon tagastama väärtuse, mis on reaalarv. Funktsiooni plokk täidab sarnast ülesannet, kuid erinevalt funktsioonist ei ole tarvis tagastada väärtust. Funktsiooni plokk on eraldiseisev alamprogramm, mida väljakutsudes täidetakse programmi sisu, kuid saadud tulemust ei ole kohustus koheselt kuhugi omistada.

Tabel 3.2. Andmetüübid[11]

Tüüp	Madalaim väärtus	Kõrgeim väärtus	Mälumaht, bit
Bait *(BYTE)*	0	255	8
Sõna *(WORD)*	0	65535	16
Topelt sõna *(DWORD)*	0	4294967295	32
Märgiga sümbol *(SINT)*	-128	127	8
Märgita sümbol *(USINT)*	0	255	8
Reaalarv *(INT)*	-32768	32767	16
Märgita reaalarv *(UINT)*	0	65535	16
Märgiga topelt reaalarv *(DINT)*	-2147483648	2147483647	32
Märgita topelt reaalarv *(UDINT)*	0	4294967295	32

Programmis muutujate andmetüüpide valikul tuleks lähtuda sellest, kui suurt numbrilist väärtust muutujas salvestatakse. Kui numbriline väärtus on väiksem kui 255, on otstarbekas kasutada kas bait või märgita sümbol tüüpi muutujat (tabel 3.2). Kui sama väärtuse korral kasutada reaalarv tüüpi muutjat, siis sellisel juhul kasutatakse üleliigselt loogikakontrolleri mälu, kuna sõltumata väärtuse suurusest kasutatakse ära 8 biti asemel 16 bitti. Lühemates programmides mälu puudust ei teki, kuid on probleemiks, kui soovitakse juhtida suuremat automaatset tootmise liini.

Lisaks põhilistele programmeerimise keele elementidele on Indra Logic programmeerimise keskonnas hulgaliselt funktsioone, mis lihtsustavad programmeerija tööd. *Technics 1* masina projekteerimisel kasutatakse lisafunktsioone:

1. Aja mõõtmiseks.
2. *Modbus* andmeside ühenduse loomiseks.
3. Keskmise väärtuse arvutamiseks.
4. Sageduse mõõtmiseks.
5. *PID* reguleerimiseks.
6. Lineaarse arvu piiramiseks.
7. Digitaal sisendi tõusva frondi tuvastuseks.

Aja mõõtmiseks on programmeerimise keskonnas kolm erineva meetodit: välja lülitatud aja mõõtmine, sisse lülitatud aja mõõtmine ja pulseeriv kell. Välja lülitatud oleku mõõtmise kell

väljastab Boole'i väärtust, mis on väär määratud aja jooksul. Sisselülitatud oleku mõõtmise kell väljastab tõese signaali määratud aja jooksul. Pulseeriv taimer väljastab tõese väärtuse ja hoiab seda määratud aja jooksul.

Modbus andmeside ühenduse funktsiooni kasutati andmete saatmiseks *SCADA* süsteemi. *Modbus* funktsiooni plokki sisestatakse andmeside ühenduse loomiseks vajalikud parameetrid ja registri aadressid, kus andmeid hoitakse. Registrid sisestatakse andmemassiivina.

Keskmise väärtuse arvutamise funktsiooniga stabiliseeritakse kõikuvaid analoog väärtusi. Keskmistamise funktsoon loeb ujuvkoma väärtust ning arvutab välja keskmise väärtuse, maksimaal väärtuse ja minimaal väärtuse. Maksimaalne lugemite arv keskmise arvutamiseks ei ole määratud, seega peab funktsiooni ploki kasutaja piirama seda, taaskäivitades funktsiooni teatud korduste tagant.

Sisendimpulsside sageduse või programmi sisese diskreetse väärtuse muutuse sageduse mõõtmiseks on kasutajal võimalik kasutada vastavat funktsiooniplokki. Sagedus arvutatakse ujuvkoma arvuna ja dimensiooniks on Hz. Sageduse mõõtmise funktsioonil on samuti sisse ehitatud väärtuse keskmistamine, mis võimaldab mõõta sageduse keskmist väärtust kuni 10 viimase pulsi kohta. Sageduse arvutamisel tekib viga, kui sagedus väheneb. Kui mehhanism seisma jäi, ei nullitud sageduse väärtust. Programmis oli tarvis luua vastav tingimus, mis teatud olukorras sageduse väärtuse nullis. Nullimiseks on funktsiooniplikil taaskäivitamise sisend.

PID reguleerimise plokki kasutatakse *Technics 1* masina püsikiiruse hoidmiseks sobiva rõhu seadeväärtuse saamiseks. Regulaatori sisenditeks on mootori hetkekiirus, kui sõitja lülitab sisse nupu vajutusega püsikiiruse hoidmise, siis salvestatakse seadepunkti hetke mootori kiirus ja *PID* regulaator lülitub automaatrežiimile ja hoiab määratud mootori kiirust. *PID* funktsioonil on ka manuaalrežiim, kus loetakse funktsiooniploki *manuaal* sisendisse võistlusmasina pedaali asendiga valitud rõhu väärtus ja *PID* funktsiooni plokk väljastab seadeväärtuse otse analoogväljundi registri sisendisse, millega seadistatakse masina töörõhku.

Lineaarse arvu piiramise funktsiooniga muundati sisendväärtus kasutaja poolt valitud vahemikku. Näiteks pedaal väljastas analoogpinge, mis muundati digitaalseks arvuliseks väärtuseks. Digiteeritud väärtus oli näiteks vahemikus 50 ... 8000 ja väljundite moodul rõhu

reguleerimiseks töötas vahemikus 0 … 65535, muutes pedaali digiteeritud väärtuse sobivasse vahemikku.

Diskreetse sisendi tõusva frondi tuvastust on tarvis kasutada, kui soovitakse mõõta diskreetse sisendi oleku muutust. Oleku muutust on tarvis mõõta nuppudel ja kiiruse mõõtmise anduritel. Kuna silindri kiiruse andurid ja mootori positsiooni andurid digitalisseritakse 16 - bitise arvuna, siis kasutatakse tõusva frondi tuvastust vaid kindla biti oleku muutuse tuvastamiseks.

3.2 Andurite ja täiturite juhtimine loogikakontrolleriga

Pneumosõiduki automaatjuhtimises on kolmel meetodil juhitavaid andureid ja aktuaatoreid:

1. Analoog signaaliga.
2. Diskreetsignaaliga.
3. Sünkroonandmesidega juhitavad (*SSI*).

Analoogsignaaliga suhtlus toimub rõhuandurite ja -regulaatoriga. Rõhuandurid väljastavad analoogsignaali, mida loogikakontrolleri analoogmoodul loeb voolusignaalina. Pneumosõiduki ehitusel kasutatakse rõhuandureid, mis väljastavad voolusignaali vahemikus 4 ... 20 mA. Selles piirkonnas mõõdetud voolu väärtus väljendab rõhu väärtust 0 ... 1000 kPa, kus 4 mA signaal on 0 kPa ja 20 mA signaal on 1000 kPa.

Rõhuregulaator saadab rõhu väärtust loogikakontrollerile ja loogikakontroller reguleerib rõhku voolu signaaliga. Kontrollerilt regulaatorile saadetakse seadistavat voolusignaali. Seadistavaks voolusignaaliks on samuti 4 … 20 mA. Selles voolu vahemikus seadistatakse süsteemi rõhku 0 ... 1000 kPa. Andurilt saadetakse kontrollerile seadistatud rõhu väärtuse tagasiside. Regulaatori tagasiside võimaldab tuvastada süsteemi rõhu langemise algust tingitult kas vahepaakide aeglasest täituvusest või põhipaagi lõpplikust tühjenemisest.

Diskreetsignaaliga juhitakse pneumosõiduki seadmeid, millel ei ole mõõtvat fuktsiooni vaid on kaks olekut: kas sisse või välja lülitatud. Seda tüüpi seadmeteks on pneumosõidukil: pneumojaotid, indikaator lambid ja lülitid. Väljundsignaali tekitamiseks on programmis tarvis vastav bitt kõrgeks kirjutada. Sisendsignaali puhul muutub kontrolleri programmis biti olek vastavalt mooduli sisendklemmile rakendatavale pingele.

Sünkroonandmesidega (*SSI*) juhitakse pnumosõiduki absoluutenkooderit. Sünkroon-andmesides toimub suhtlus kaheksajuhtmelise süsteemina (tabel 3.3).

Tabel 3.3. Absoluutenkooderi ühendamine[12]

Juhtme värv	Signaal	Selgitus
Pruun	Andmed - (*Data* -)	Andmeside +
Valge	Andmed + (*Data* +)	Andmeside -
Must	(*CW / CCW*)	Enkooderi pööramise loendur
Roosa	(*SET*)	Null punkti seadistamine
Kollane	(*Clock* +)	Andmeside kella signaal +
Lilla	(*Clock* -)	Andmeside kella signaal -
Sinine	(*GND*)	Maanduse ühendus
Punane	(+U_s)	Toitepinge

SSI moodul tuleb normaaltalitluseks seadistada. Kui moodul lisatakse projekti, lisatakse automaatselt juurde 2 registrit moodulile andmete saatmiseks ja 2 registrit moodulilt andmete lugemseks. Mooduli seadistamiseks tuleb saata mooduli mõlemasse registrisse seadistamise kood. Kood on 16 bitine number, mis tuleb koostada vastavalt enkooderi parameetritele. Koodi koostamiseks on tarvis teada:

1. Enkooderi resolutsiooni.
2. Enkooderi liikumise suunda.
3. Andmeedastamise kiirust.
4. Gray kood või binaarkood positsiooni lugemine.

Lisaks enkooderi parameetritele saab moodulil seadistada töö olekut: moodul on töörežiimis, loetakse positsiooni väärtust, moodul on välja lülitatud ja veateate teadvustamine. Lisaks töörežiimidele saab seadistada vea tuvastust kontrollbiti alusel.

3.3 Automaatjuhtimise kirjeldus

Pneumosõiduki mootori automaatjuhtimine põhineb peamiselt pneumojaotite lülitamisega. Pneumojaotite lülitused peavad olema ajastatud sedasi, et kolbide liikumisel tekiks väljundvõllil pöördliikumine. *Technics 1* võistlus masinal on radiaalmootor. Mootori tüübi

idee on algselt pärit vanadelt lennukitelt, kus paigutati mootori peavõlli edasi liigutavad kolvid ringiratast ümber pöörleva telje.

Kuna lennuki mootorite ehitusel piiras silindrite arvu vaid see, kui palju kolbe mahtus ringjoonele üksteise kõrvale ühtlaselt jaotades sai paigutada silindreid vastavalt sellele, et jõud jaotuks väljundvõllile ühtlaselt. Pneumosõidukite võistluse reeglitega on silindrite arv piiratud ja kasutada võib vaid 4 silindrit.

Mootori normaalseks talitluseks kasutatakse pneumojaoteid (joonis 1.4), mille lülitamisel lastakse silindrisse suruõhk ja kui klapp välja lülitatakse, lastakse silindris olev suruõhk atmosfääri ning kolb saab edasi liikuda vabalt. Mootori kolbidel on tarvis vabalt liikumist, et ületada hetked, kus kolvi töötamine takistaks mootori väljundvõlli pöördliikumist. Takistav hetk esineb punktis, mil võlli keps on paralleelne diagonaalides asetsevate silindritega. Takistavaid punkte ühe täispöörde jooksul on 4 tükki. Neid punkte ületades lülitatakse paralleelselt asetsevad kolbid vabalt liikuma. Igal takistaval pöördnurgal peab kaks kolbi liikuma 20 kraadi vabalt, mis terve pöörde jooksul moodustab 80 kraadi, mil mootor töötab vaid 50% võimsusega. [13]

Mootori automaatjuhtimiseks on tarvis koostada programm. *Technics 1* võistlusmasinat juhib programm, mis koosneb 8 osast. Esimese osana on defineeritud kõik süsteemiülesed e globaalsed muutujad (LISA O). Globaalsed muutujad on tarvilikud sisend väljund registrite määramiseks, kuna registrid kehtivad kogu programmi ulatuses sarnaselt. Globaalsetele muutujatele on võimalik ligipääseda kõikidest programmidest ja funktsiooni plokkidest piiranguteta. Globaalset tüüpi muutujate juures tuleb olla ettevaatlik, et muutuja kirjutamisel ei tekiks konflikti. Näiteks kui kahes programmis kirjutatakse üle sama muutuja jääb kehtima viimane kirjutamine, kuigi kasutaja soovis kasutada esimest.

Teiseks peamiseks programmi osaks on põhi programm (*PLC_PRG*) (LISA P). Selles programmis kutustakse välja funktsiooni plokid, mis juhivad masina tööd:

1. Pneumojaotite lülitus.
2. Ülekande suhte muutmine.
3. Mootori ja masina liikumise kiirused.
4. Analoogsisendi väärtused.

5. Enkooderi nurga väärtus.
6. Analoogväljundi väärtused.

PLC_PRG programmis saab alguse kogu masina juhtimine. Põhiprogramm on otstarbekas vaid programmi voo kontrollimiseks. Programmi lugemise lihtsustamiseks on iga mooduli juhtimiseks koostatud eraldi funktsiooniplokk.

Pneumojaotite juhtimine toimub programmis, mille nimeks on *Motor* (LISA Q). Selles programmis on defineeritud jaotite lülitamine vastavalt võlli nurgale. Lisaks jaotite lülitamisele arvutatakse kolvi pindalal hetkel rakendunud jõud.

Pneumosõiduki ülekande suhte muutmiseks on loodud funktsioonipokk *GearChange* (LISA R). Käiguvahetuse funktsiooniplokis toimub käikude vahetamine vastavalt masina liikumise kiirusele.

Mootori pöörlemise- ja masina liikumiskiiruse mõõtmiseks on koostatud eraldi funktsiooni plokk (LISA S), pulsside sagedusest tarvis ka keskmistada. Kiiruste mõõtmine toimub peamiselt arvsõna *n*-biti muutumise sageduse mõõtmisega.

Analoogsisendi mooduli funktsiooniplokis (LISA T) seadistatakse analoogmoodul ja loetakse väärtused sisenditest muutujatesse. Enkooderi funktsiooni plokis (LISA U) seadistatakse absoluutenkooder vastavalt enkooderi tehase andmetele, loetakse digitaliseeritud nurga väärtus ja muundatakse saadud väärtus nurga väärtuseks vahemikus 0 ... 360 kraadi.

Analoogväljundi funktsiooniplokis reguleeritakse väljundrõhku (LISA V). Väljundrõhku reguleeritakse masinal kas automaatselt või pedaali asendiga. Lülitades sisse automaatne rõhu reguleerimine, seaditab rõhuregulaatorit *PID* regulaator.

Pneumojaotite automaatjuhtimine on üks osa mootori juhtimisest. Teise osa juhtimisülesandest moodustab suruõhu pealevoolu reguleerimine. Õhu pealevoolu on võimalik piirata kahel meetodil: rõhu reguleerimine ja õhuvoolu reguleerimine. Mõlemad parameetrid piiravad pneumaatilise süsteemi võimsust otseselt, kuna võimsus on defineeritud õhuvoolukiiruse ja rõhu korrutisena (valem 3.1).

$$P = p \cdot Q \tag{3.1}$$

kus P on võimsus, W;

p - rõhk, Pa;

Q - õhuvool, m^3/s;

2015. aasta võistlusmasinal reguleeriti suruõhuvoolu õhuvoolukiiruse järgi. Õhuvoolu kiiruse reguleerimiseks valmistati elektrooniline õhuvoolu regulaator. Regulaator põhines pneumaatilisel drosselil ja elektrimootoril. Kuna oodatud ökonoomsust drosseli kasutamisega ei saavutatud, reguleeritakse suruõhu pealevoolu 2016. aasta masinal rõhu järgi. Rõhu muutmiseks süsteemis on tarvis vaid elektroonilist rõhuregulaatorit ja reguleerimise toime pneumosüsteemile on kiire.

3.4 Automaatika skeemid

Automaatika seadmed on ühendatud juhtmetega, mille ristlõige on 0,75 mm^2. Tööstusstandardile vastavaid kaableid ei ole kasutatud, kuna kaabli mass on liialt suur. Kuna pneumosõiduki läheduses puuduvad seadmed, mis tekitaksid süsteemi häirivat müra, ei ole varjestusega kaablite kasutamine põhjendatud. Vaatamata sellele, et seadmete voolutarve jääb 0 ... 500 mA piiridesse, ei olnud väiksema ristlõikega juhti otstarbekas kasutada, kuna juhtme purunemise oht seetõttu kasvab.

Automaatika süsteemi koostamine algab loogikakontrolleri moodulite ühendamisega (LISA B). Loogika kontrollerite moodulite järjekord peab vastama programmeerimise keskonnas määratud järjekorrale, vastasel juhul väljastatakse veateade. Seadmete ühendus moodulite külge on välja toodud eraldi plokkidena. Diskreetsete sisndite (LISA C, LISA D, LISA E, LISA F) ja analoogsisend moodul (LISA G, LISA H, LISA I, LISA J), mis omakorda koosneb neljast plokist. Plokid ei ole füüsiliselt eraldatud, aga on visuaalselt eristatavad. Plokid on kujutatud eraldi joonistel, kus on ka näidatud seadmete ühendused.

Joonistelt on näha, et kõik maandused on ühendatud kokku. Ühendus on tehtud põhjusel, et vähendada loogikakontrolleri külge minevaid juhtmeid. Taoline lahendus jätab visuaalselt parema pildi ja tagab lihtsama ligipääsu signaalijuhtmetele. Kõik toitepinge ja maanduse ühendused on viidud klemmliistu alla (LISA N), mille teine pool on sillatud.

Tuvastamaks sisendite ja väljundite kogu arvu on koostatud seadmete ühendamise põhimõtteskeem (LISA A). Skeemil on visuaalse pildi loomiseks joonistatud pnumosõiduki pealtvaade, millele on paigutatud kõik automaatika seadmed nende reaalsesse asukohta. Seejärel on näidatud seadmete juhtimise või tagasiside signaai tüüp. Taoline skeem annab ülevaate, kui palju on tarvis erinevat tüüpi sisendeid ja väljundeid.

SSI-sisend mooduli joonisel (LISA L) on kujutatud absoluutenkooderi ühendamise skeem. Kuna *SSI* ühendus toimib 8 juhtmelises süsteemis on joonisel kujutatud hargnev kaabel ja ära tähistatud ühendused. Eraldi on välja toodud värvide tähendused.

Analoogväljundi mooduli joonisel (LISA K) on näidatud rõhuregulaatori ühendus. Rõhu regulaatoril on kaks ühendust, millest tagasiside läheb analoogsisendi mooduli külge ja rõhu reguleerimise signaal tuleb analoogväljundi moodulilt.

Loogikakontrolleri küljes (LISA M) on arvutivõrguliides ja loogikakontrolleri diskreetsed sisendid ja väljundid. Kuna *Tehchics 1* meeskonna masinal diskreetseid signaale on vähe, kasutatakse vaid loogikakontrolleri enda diskreetsisendeid. Võrguliidese külge ühendatakse *WiFi* ruuter, millega luuakse juhtmevaba ühendus arvuti ja loogikakontrolleri vahel.

3.5 Piirangud ja võimalused

Pneumosõiduki mootori automaatjuhtimise eesmärgiks on kontrollida mootori liikumist ja säästa üleliigset õhukulu. Õhukulu säästlikust piirab aga mootori võimsus (valem 2.1). Seega on mootori võimsus üheks peamiseks piirajaks ökonoomse süsteemi loomisel.

Kokkusurutud õhul on omadus paisuda. Paisumist on võimalik ära kasutada, kui lülitada kolvi liikumine välja enne, kui kolb on jõudnud lõpp-positsioonini. Suruõhu paisumist ärakasutava süsteemi loomiseks on tarvis lisada pneumojaoti, mis peataks suruõhu juurdevoolu, kui kolb hakkab lähenema otsa positsioonile, kuid samas peab üks silindri pool olema rõhuta.

Taolist lahendust ei ole otstarbekas kasutada süsteemis, kus on palju silinderid. Silindrite arvu tõttu läheb lisajaotite hulk suureks, mis omakorda pikendab oluliselt pneumotorude kogupikkust. Liialt pikad pneumotorud tekitavad süsteemis rõhulangu, mis on suur kulu pneumosõiduki energiale.

4. ANDMEHÕIVE JA KAUGJUHTIMINE

4.1 Andmehõive süsteemi kirjeldus

Pneumosõiduki andmehõive süsteem on loodud andurite poolt väljastatavate väärtuste salvestamiseks. Andmehõive süsteemi ülesandeks on salvestada mõõdetud väärtused ja luua erinevad graafikud suruõhu süsteemis toimuvate muutuste kohta võistluse ajal. Saadud andmete põhjal on hiljem võimalik hinnata masina võimekust suruõhu jõul liikudes.

Pneumosõiduki andmehõive süsteem koosneb järgmistest komponentidest:

1. Programmeeritav loogikakontroller.
2. *SCADA* tarkvara.
3. WiFi ruuter.

SCADA tarkvaraga kuvatakse mõõdetud andurite väärtuseid graafilisel liidesel, koostatakse graafikud, salvestatakse mõõdetud tulemused andmebaasi. *SCADA* tarkvaraks kasutatakse pneumosõiduki andmehõive loomisel *Inductvie Automation* tarkvara tootja programmi *Ignition.*

WiFi ruuteri ülesandeks andmehõive süsteemis on distantsühenduse loomine. Loogikakontrolleri ja arvuti vaheline suhtlus toimub *Modbus TCP* protokolliga.

Pneumosõiduki andmehõive süsteemi töötamine reaalajas tähendab seda, et andurite väärtused peavad uuenema katkematult kindla aja tagant. Kui masina sõit algab, alustatakse andmete salvestamist ja kui sõit lõppeb, siis ka andmete salvestamine peatub.

Vastavat võisltuse reeglitele peab *SCADA* süsteem salvestama järgmisi andmeid:

1. Kolvi liikumise aegdiagramm.
2. Kolvi jõu diagramm sõltuvalt positsiooni muutusest.
3. Masina liikumise kiirus.
4. Kolvi liikumuise kiirus.
5. Õhu kulu diagramm.
6. Mootori töörõhk.

Kolvi liikumise aegdiagramm saadakse, kui mõõdetakse kolvi positsiooni ja salvestatakse positsiooni väärtus teatud aja tagant. Kolvi positsiooni väärtus mõõdetakse silindri korpuse külge kinnitatud lineaarse positsiooniandurigа. Lineaarne positsiooniandur on magnetandur, mis teab kolvi asendit terve käigu ulatuses ja asend jääb mällu ka toitepinge kadumisel.

Kolvi jõudiagramm sõltuvalt positsioonist saadakse, kui on teada kolbi pindala, mõõdetakse silindris olevat rõhku ja salvestatakse silindri psitsiooni muutus. Kolbi pindalal tekkiv jõud saadakse arvutuse teel (valem 4.1).

$$F = p \cdot A \tag{4.1}$$

kus F on jõud, N;

p - rõhk, Pa;

A - kolvi ristlõike pindala, m^2;

Masina kiiruse mõõtmine toimub relatiiivse enkooderiga. Enkooderit kasutades saab ratta võlli nurga muutumisel teatud hulga pulsse, mille summa jagamisel ajaga saame nurkkiiruse. Teades ratta ringjoone pikkust saame nurkkiiruse teisendada ümber ühikusse km/h. Enkooder tagab täpsema ja kiirema kiiruse mõõtmise võrreldes süsteemiga, kus kasutatakse keelreleed ja magnetit.

Analoog sisendisse loetud analoog signaal digitaliseeritakse 16 bitisesse arvsõnasse. Kolbi kiiruse mõõtmiseks loetakse arvsõna kümnenda biti muutust. Muutus registreeritakse tõusva frondi trigeriga. Triggerit kasutades registreeritakse vaid olukord, kus kümmnes bit muutub kõrgeks, mis välistab kiiruse pideva kasvu, kui silinder jääb seisma hetkel, mil kümmnes bit on kõrges oleksus.

Kuna loogikakontrolleri analoog sisendi digitaliseeritud väärtuse vahemik on 0 ... 30000 ja teades, et selle väärtuse vahemikus liigub kolb 0 ... 150 mm saame välja arvutada kolbi nihke, mil bit number 10 muutub kõrgeks. Binaar arv detsimaal arvuks ümber teisendades saame, et 10 bitine binaar arv on võrdne detsimaal arvuga 1023. Analoog sisendi digitaliseeritud maksimaal väärtus selle arvuga läbi jagades saame teada, kui mitu korda muutub 10. biti väärtus kolvi liikumis ulatuse jooksul. Korduste arvuks antud juhul on 29,3. Kuna 10. bit läbib madala oleku juures sama distantsi, mis kõrges olekus, tuleb kordaja kahega läbi korrutada ja saame 58,6. Kui kolbi käigu kogu pikkus saadud kordajaga läbi jagada saame 10

biti muutuse sagedust mõõtes välja arvutada silindri liikumise kiiruse. Antud juhul on 1 Hz sageduse juures silindri liikumise kiirus 58,6 mm/s

Õhukulu diagrammi on võimalik saada kahel meetodil:

1. Anduriga õhuvoolu mõõtes
2. Arvutuslikul meetodil rõhkusid arvestamta

Täpseima tulemuse õhu hulga kohta saab kasutades õhuvoolu andurit. Lisaks õhuvoolu andurile on võimalik õhuvoolu hulk ajas leida ka arvutuslikul meetodil (valem 4.2). [2]

$$q = V \cdot A \tag{4.2}$$

kus q on õhuvooluhulk, $\frac{\mathrm{m}^3}{\mathrm{s}}$;

V - silindri liikumise kiirus, $\frac{\mathrm{m}}{\mathrm{s}}$;

A - kolvi pindala, m;

Arvutusliku meetodiga saadakse õhuvoolu hulk liitrit minutis. Väärtuse teisendamiseks normaalliitritesse minutis on tarvis teada suruõhu rõhku ja temperatuuri. Õhuvoolu hulk teisendati normaalliitritesse minutis kasutades ideaalgaasi võrrandit (valem 4.3).

$$P \cdot V = n \cdot R \cdot T \tag{4.3}$$

kus P on gaasi rõhk, Pa;

V - õhuvoolu hulk, $\frac{\mathrm{L}}{\mathrm{min}}$;

n - Avokadro arv, $6{,}022 \cdot 10^{23}$;

R - gaasikonstant, $8{,}314\ \mathrm{J} \cdot \mathrm{K}^{-1} \cdot \mathrm{mol}^{-1}$.

Kuna molekulide arv gaasis jääb samaks saame arvutada õhuvoolu hulga normaalliitrites (valem 4.4).

$$V1 = \left(\frac{P2}{P1} \cdot \frac{T1}{T2}\right) \cdot V2 \tag{4.4}$$

kus $V1$ on õhuvoolu hulk normaalliitrites, $\frac{\mathrm{m}^3}{\mathrm{s}}$;

$P2$ - tegelik rõhk, Pa;

*P*1 - atmosfääri rõhk, Pa;

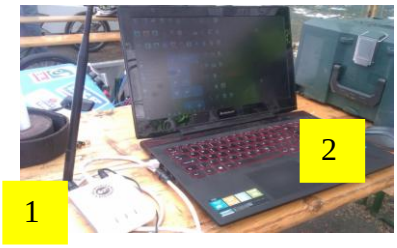
T1 - väliskeskonna temperatuur, K;

T2 - mõõdetud gaasi temperatuur, K.

Mootori töörõhku väärtus saadakse otse andurilt. Rõhu väärtust kasutatakse ka eelkirjeldatud parameetrite välja arvutamisel nt. kolbi ristlõikel tekkiv jõud.

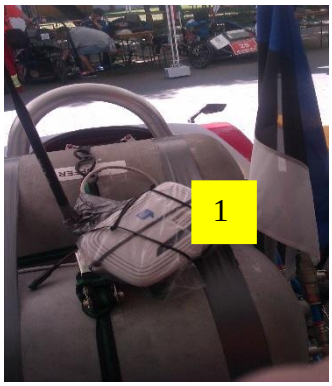
Pneumosõiduki telemeetriasüsteemi toimimiseks on tarvis kontrolleri ja arvuti vahel luua juhtmevaba ühendus. Juhtmevaba ühendust on tarvis, et masina andmed võistlus sõidu ajal reaalajas andmebaasi salvestataks. Telemeetria süsteemi moodustab kolm põhiseadet : arvuti, *WiFi* ruuter, *WiFi* signaali kordaja ja loogikakontroller.

WiFi signaali kordaja (joonis 4.1), mida süsteemis kasutati oli *fon2200*. Signaali kordaja oli ühendatud sülearvutiga võrgukaabli abil. Signaali kordaja oli mõõtmetelt väike ja seda oli lihtne sülearvutiga raja ääres ringi liikudes kaasas kanda. Standardantenn oli välja vahetatud 8 db antenni vastu, mis võimendas signaali tugevust. Toiteks oli tarvis 230 VAC, mis saadi rajal paiknenud võrgutoite jagamise keskustest.



Joonis 4.1. WiFi ligipääsupuntk ja sülearvuti: 1 − wifi ligipääsu punkt; 2 − sülearvuti.

Sõidu ajal oli masina küljes *WiFi* ruuter (joonis 4.2), mille külge oli ühendatud *Rasperry Pi* ja loogikakontroller. *WiFi* ruuteri tööpõhimõtteks oli *WiFi* signaali edastus ja ligipääsupunkti eesmärgiks oli ühenduda selle signaaliga. Kaks seadet olid seadistatud tööle vaid omavhael. Läbi *WiFi* ühenduse oli võimalik masinat programmeerida ja *SCADA* süsteemi andmeid lugeda.



Joonis 4.2. *WiFi* ruuter masina küljes: 1 − *WiFi* ruuter

Lõpptulemusena oli *WiFi* signaal piisavalt tugev ja võistlusrajal ette jäänud puud signaali ei katkestanud. Kiirenduse sõidus andmeid ei salvestatud, kuid prooviti tehnilisest alast masinaga ühendust hoida. Kuna antenn oli madalal ja ette jäi mitmeid takistavaid tegureid nagu teised võistlusmasinad, meeskonnaliikmed, stardialal liikumine. Seetõttu oli ühendus katkendlik juba sõidu alguses. Vaatamata katkendlikkusele püsis ühendus kuni sõidu lõpuni, mille finišiala oli tehnilisest alast ligi 200 ... 300 m eemal.

Arvuti on süsteemis andmete kogumiseks. Arvutisse on installitud *SCADA* tarkvara, mille abil luuakse ühendus loogikakontrolleri ja arvuti vahel. Loogikakontroller on seade, mis paikneb pneumosõidukis ja avaldab masinale juhtivat toimet vastavalt anduritelt kogutud informatsioonile. Kontrolleri külge ühendatud *WiFi* ruuter võimaldab andurite väärtusi ja täiturite olekuid juhtmevaba ühendusega arvutisse saata.

Pneumosõiduki automaatjuhtimiseks on kasutusel mitmed andurid ja täiturid. Sõiduki suruõhu kasutamise optimaalsuse hindamiseks on tarvis teada rõhku ja õhuvooluhulka. Masina mootori tüübi tõttu kasutatakse sõidukil ka absoluutenkooderit. Absoluutenkooderi ülesandeks on mootori võlli pöördenurga väärtuse mõõtmine.

4.2 Andmete salvestamine

Andmed salvestatakse *MySQL* andmebaasi. *MySQL* on vabavarana saadaval olev andmebaasi tarvara. Loogikakontrollerilt saadetud andurite väärtused salvestatakse *MySQL* andmebaasi tabelisse. Tabelit saab ümber salvestada *Microsoft Exceli* faili formaati .csv. Pneumosõiduki projekti raames kasutatakse *MySQL* versiooni 6.3. Kuna *Ignition* toetab *MySQL* andmebaasi kasutamist tuleb see vaid seadistada. Seadistamiseks on tarvis luua uus andmebaas ja *Ignition* seadistuses luua ühendus loodud andmebaasiga.

MySQL andmebaasi tabelisse salvestati järgmised andmed

1. Õhuvoolu hulk, [L/s].
2. Masina liikumise kiirus [km/h].
3. Kolbide liikumise kiirused [mm/s].
4. Kolbide positsioon [mm].
5. Silindrite töörõhk [bar].
6. Kolbi pindalal tekkiv jõud [N].
7. Aeg [H:M:S:ms].

Süsteemi õhuvoolu hulk on teoreetiline suurus, mis arvutati kolvi liikumise kiirusest, rõhkude erinevusest ja temperatuuride erinevusest (valem 4.4). Masina liikumise kiirus on teisendatud ümber ratta pöörlemise sagedusest.

Kolbide liikumise kiirused on leitud mõõtes kolvi positsiooni muutuse sagedust ja sagedusest on tuletatud liigutud distants ajas. Kolbide positsioon saadakse lineaarselt positsiooni andurilt. Silindri töörõhku mõõdetakse rõhu anduriga.

Kuna silindrite pneumojaotid asuvad mõlemal pool masinat, mõõdetakse rõhku kahe anduriga ning väärtus näitab korraga kahe silindri peale kokku tekkivat rõhku. Saadud rõhu väärtus kahel silindril ei tohiks erineda, kuna mehaaniliselt on silindrid lukustatud samale ringjoonele ja seetõttu kolvid liiguvad sama kiiresti. Saadud andmetabelit vaadates on näha, et kahe sama poole kolbide pindal tekkiv jõud erineb maksimaalselt vaid 1 N. Kuna on teada, et rõhk peab olema kolvi ristlõikel sama, siis kolvi liikumise kiirus saab erineda vaid punktides, kus üks neljast kolvist on jõudnud oma lõpp-punkti kus ületatakse kolvi surnud punkt, ehk hetk, mil kolvi töö takistaks mootori pöördliikumist.

Kolvi pindalal tekkiv jõud on teoreetiline väärtus, mis on leitud kolvi ristlõike pindala ja liikumise kiiruse korrutisest. Programmis on arvestatud olukorda, mil kolb liigub välja ja tõukejõud tekitatakse kolvi suuremal pindalal, ning sisse liikudes kolvi väiksemal pindalal.

4.3 Graafiline liides juhtimiseks ja tekitatavad graafikud

Pneumosõiduki graafiline liides on loodud masina juhtimiseks ja andurite väärtuste kuvamiseks graafikul. Graafiline liides koosneb kahest lehest. Esimene leht on masina juhtimiseks tarvilike nuppude ja anduri väärtuste kuvamiseks. Teine leht on mõeldud graafiku kuvamiseks.

Graafilise liidese kaudu on võimalik masinat distantsjuhtida. Distantsjuhtimiseks saab *SCADA* tarkvaraga seadistada masina rõhku, valida sõidurežiim, sisse ja välja lülitada *PID* regulaatoriga juhtimine ning anda käsk sõidu alustamises ja lõpetamiseks.

Distantsjuhtimiseks peab juht esmalt andma loa füüsilise lülitiga. Füüsiline lüliti on süsteemi lisatud põhjusel, et tahtmatult või tahtkikult ei saaks masina tööd segada. Distantsjuhtimise loast annab märku scada ekraanil roheline indikaator, millel on kiri *Scada* (Joonis 4.3 - 8) Järgnevalt tuleb paika seadistada rõhk (joois 4.3 - 3). Seadistatud rõhu väärtust on võimalik näha indikaatoril *EP Preassure* (joonis 4.3 - 7). Rõhu seadistamine lubab määratud rõhul suruõhu puffer paakidest mootorisse. Mootori tööleennistamiseks tuleb teha kaks lülitust. Esmalt tuleb valida sõidurežiim (joons 4.3 - 3) ja seejärel vajutada nuppu (joonis 4.3 - 5), mis annab loogikakontrollerile loa mootor tööle lülitada.

Kaugjuhtimise teel saab masinat tööle seadistada ka automaarežiimis, kus rõhku seadistatakse vastavalt paika seadistatud kiirusele. Kui lülitatakse tööle PID regulaator, kaob *SCADA* ekraanilt rõhu reguleerimise liugur ja see asendub kiiruse liuguriga. Kiiruse liugurilt valitakse mootorile kiirus ja PID regulaator hakaab rõhku reguleerima, et seadistatud kiirust hoida..

Kaugjuhtimise negatiiveks küljeks on turvalisus, mis on lõpuni välja arendamata. Kui masin seadistatakse liikuma, ei saa juht mootori rõhku kuidagi manipuleerida. Pneumosõiduki juht saab masina üle taas kontrolli, kui lülitab sõiduki kohtjuhtimise režiimile. Kohtjuhtimise režiimile üle minnes lõpetatakse eelnevad toimingud ja rõhu reguleerimine toimib taas gaasi pedaali vajutusega.

Joonis 4.3. Graafiline liides: 1 − Masina kiirus ja mootori kiirus; 2 − Rõhu seadistamine; 3 − Sõidu režiimi valik; 4 − Lülitatud käigu indikaator, : 5 − Süsteemi seiskamine ja käivitamine; 6 Silindrite liikumise parameetrid, : 7 − Mootori positsioon, süsteemi üldrõhk ja õhuvoolu kiirus; 8 − Distants või kohtjuhtimise indikaator.

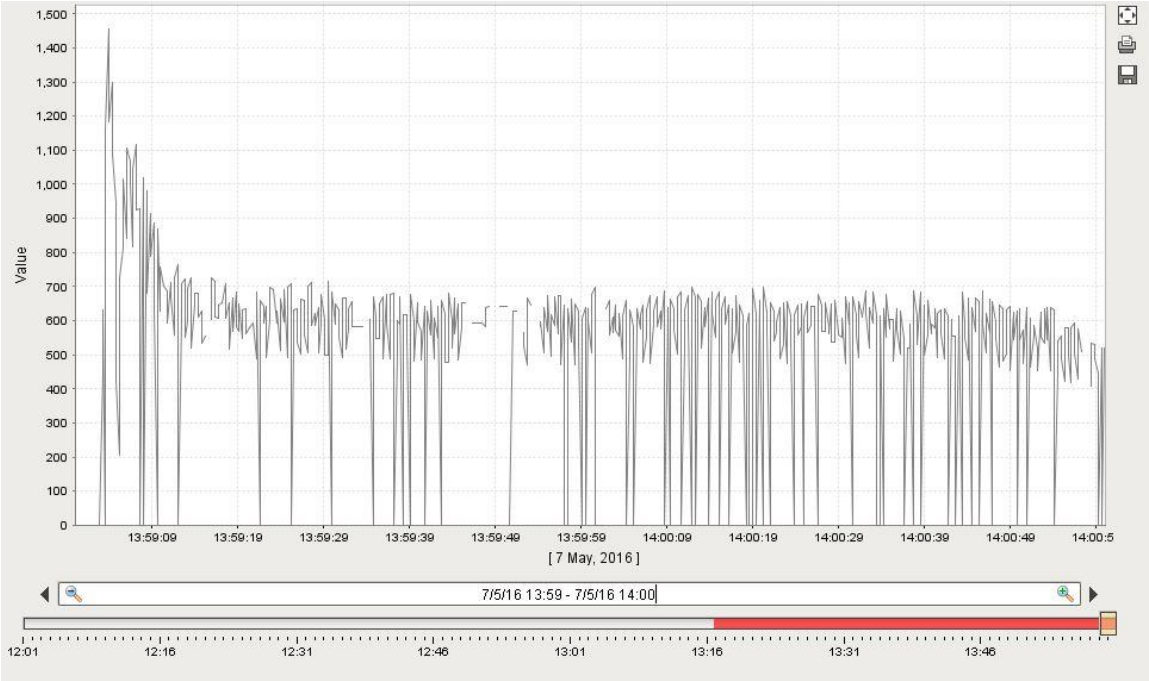
Ühel graafikul on võimalik näidata mitut erinevat muutujat. Kõik eelpool mainitud muutujad on eraldiseisvalt graafikule lisatud. Võistluse lõppedes tuli žüriile hindamiseks esitada kogu andmetabel ja pildid kõikidest graafikutest. *Ignition* tarkvara poolt pakutav graafik võimaldab muutujaid sisse ja välja lülitada. Seetõttu oli võimalik näidata masina andmed eraldiseisvate graafikutena. Andmed saadi ringraja sõidust. Kokku esitati žüriile 14 graafikut.

Õhuvoolu graafikul (joonis 4.4) on võimalik näha masina õhuvoolu kasutust. On näha, et graafikul olevad väärtused on hüplikud, millest võib järeldada, et süsteem on korrektselt tööle seadistamata. Maksimaalne õhuvooluhulk ühes hetkes oli kuni 1600 L/s, mis on liialt suur. Valemi komponentide dimensioone kontrollides viga ei esinenud. Seega oleks tarvis olnud katsetamiseks lisa aega.

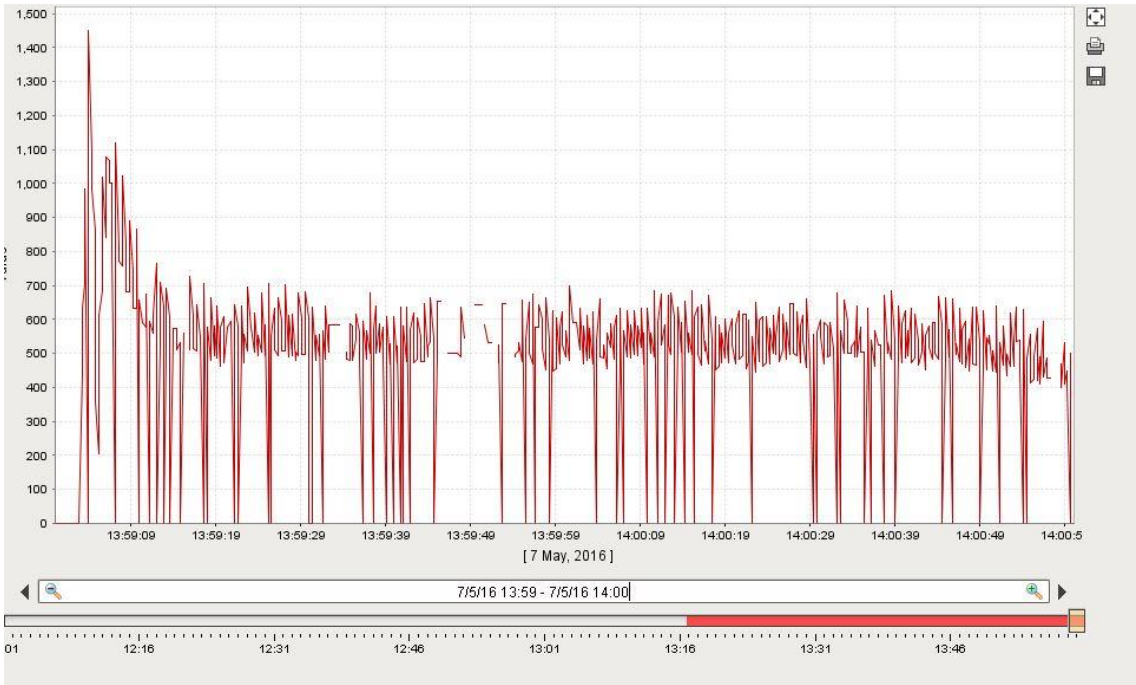
Graafikute probleemiks on andmete mahtumine graafikule. Kuna andmetihedus on liialt suur, siis terve sõidu andmeid korraga ühel graafikul näidata ei ole võimalik.

Joonis 4.4. Õhuvoolu graafik.

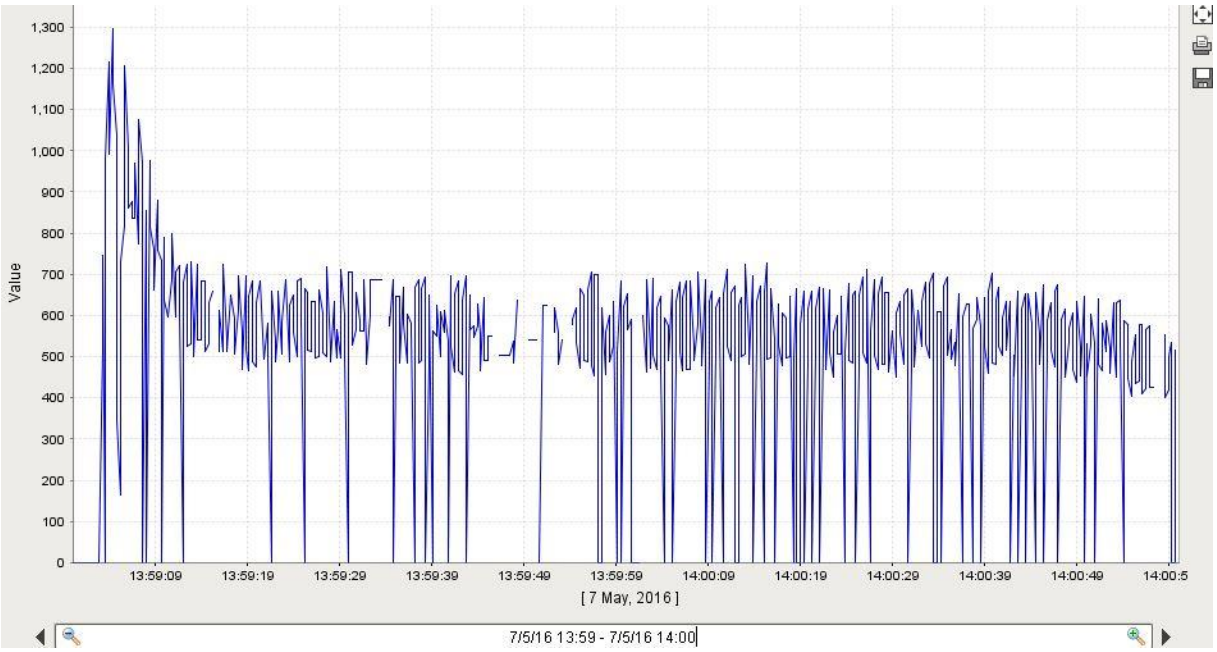
Kolvi liikumisel tekkivate jõudude muutuse graafik (joonis 4.5, joonis 4.6, joonis 4.7, joonis 4.8) näitab kolvi pindalal tekkivat jõudu igal ajahetkel. Kolvi tekitatavate jõudude graafik on samuti pidevalt muutuv. Graafikult on võimalik lugeda, et sõidu alguses, kus masin hakkab liikuma on rakendatav jõud kõige suurem. On näha, et kolvi ristlõikel tekib jõud üle 1400 N. Sõidu kestel jõud väheneb, kuna masin liigub. Pidevad üles-alla kõikumised on seletatavad silindrite jõudmisega surnud punktidesse, kus ühe või teise silndri jõud on 0 N.



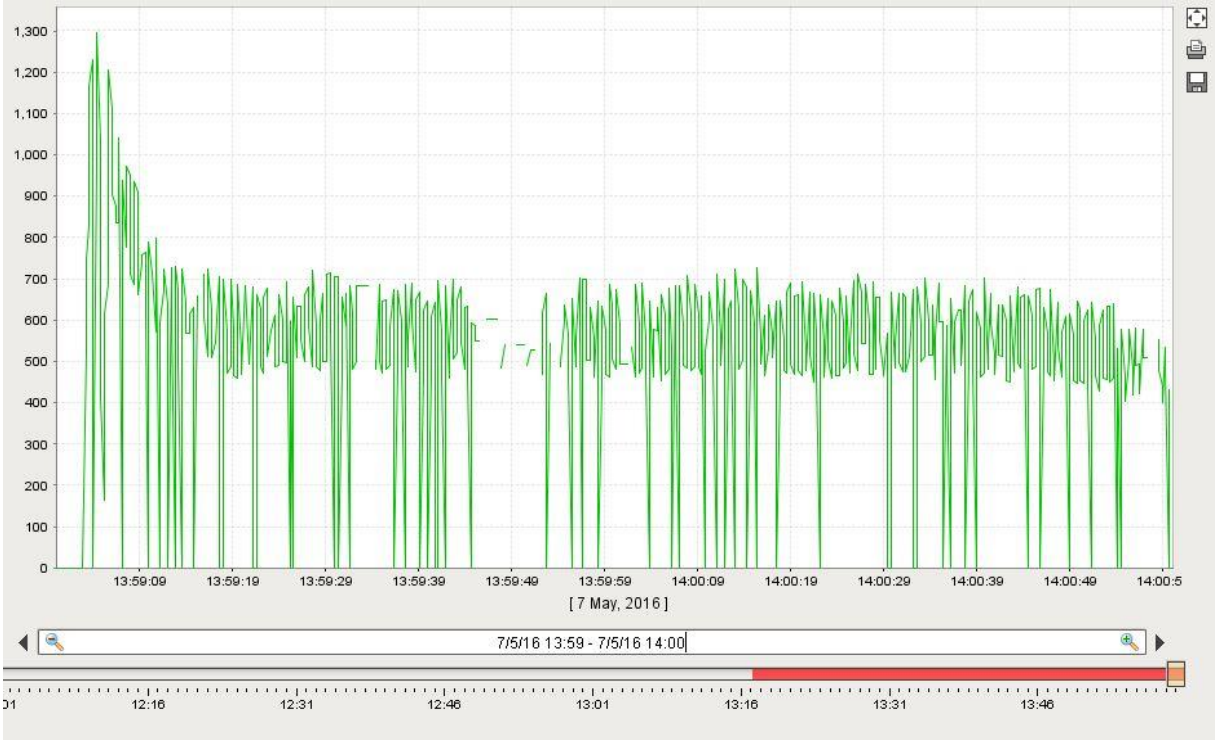
Joonis 4.5. Esimese kolvi ristlõikel rakendatav jõud.



Joonis 4.6. Teise kolvi ristlõikel rakendatav jõud.



Joonis 4.7. Kolmanda kolvi ristlõikel rakendatav jõud.

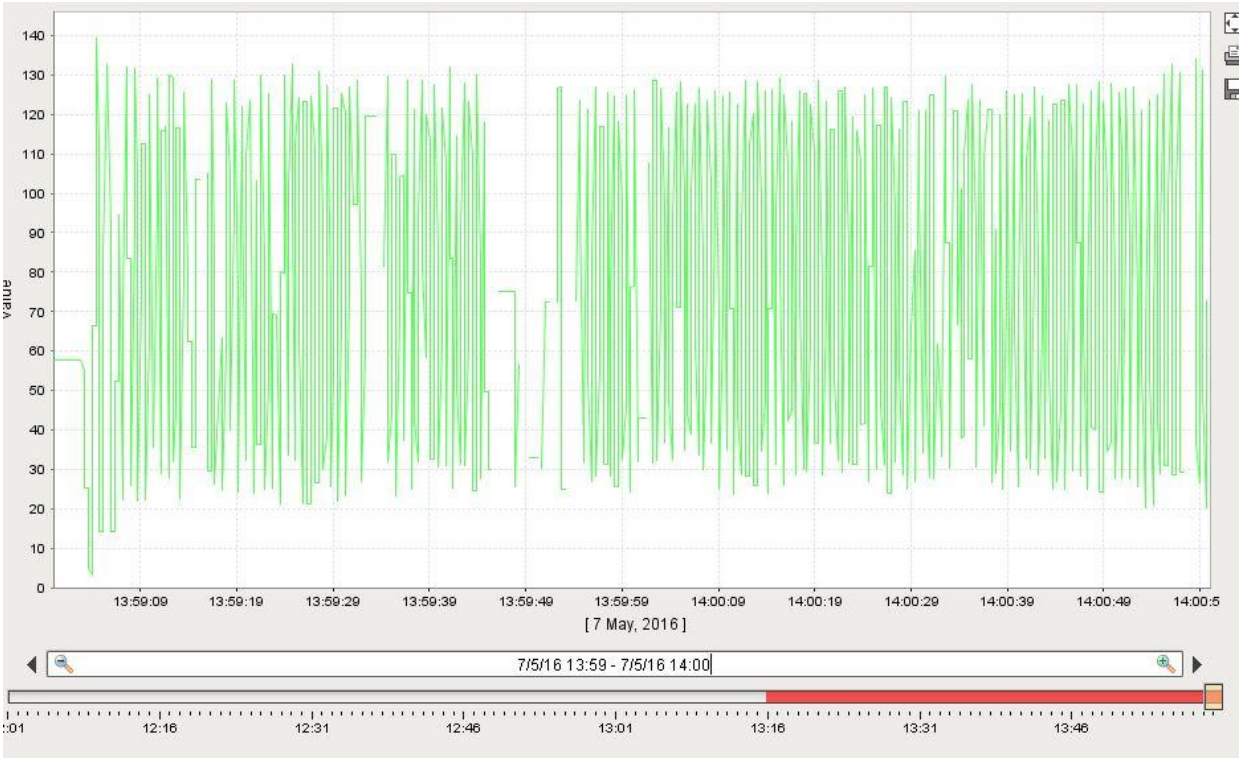


Joonis 4.8. Neljanda kolvi ristlõikel rakendatav jõud.

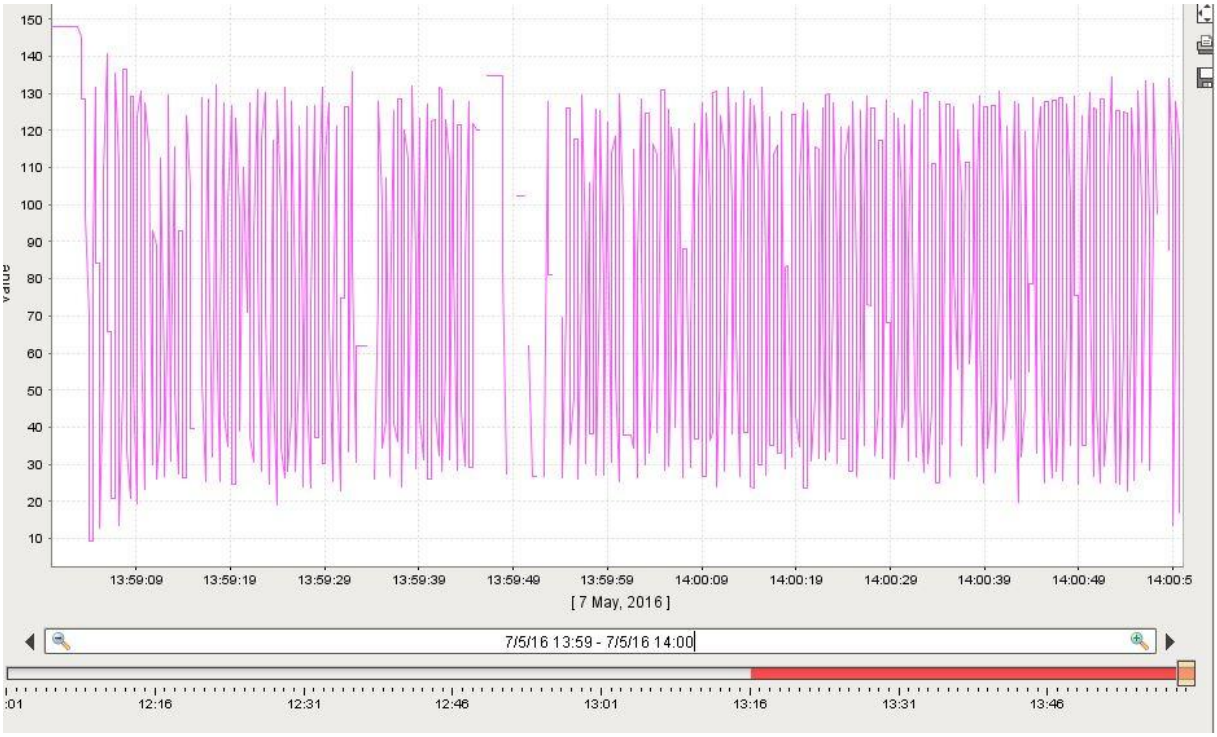
Kolvi asukoha muutuse graafik (joonis 4.9, joonis 4.10, joonis 4.11, joonis 4.12) näitab kolbide asukoha muutust asjas. Kolvikäigu pikkus on 0 ... 150 mm. Graafikut jälgides võib järeldada, et kolb ei ole liikunud 0 asendisse, ega lõppasendisse. See olukord tuleneb süsteemis esinevatest vigadest. Mootori kalibreerimise käigus häälestati silindri positsiooni andurid selliselt, et silinder liigub täpselt 0 asendist lõpp-asendisse (150mm). Esialgsete andmete kohaselt on teadmata, miks taoline probleem süsteemis esineb. On näha, et graafik algab nulli ligidalt ja liigub asendisse 140mm, kuid reaalselt peaks liikumine olema asendi 0 ... 150 mm vahel.

Teise, kolmanda ja neljanda kolvi liikumise graafik on pisut erinev. Graafikutel on võimalik näha, et kolb on ka olnud asendis 150 mm. Vahepealne tühimik väljendab, kas andmete kaotsi-minekut või mootori seismist. Kuna on näha, et tühimikus on üksikud mõõtepunktid on põhjust järeldada, et andmed on läinud kaotsi.

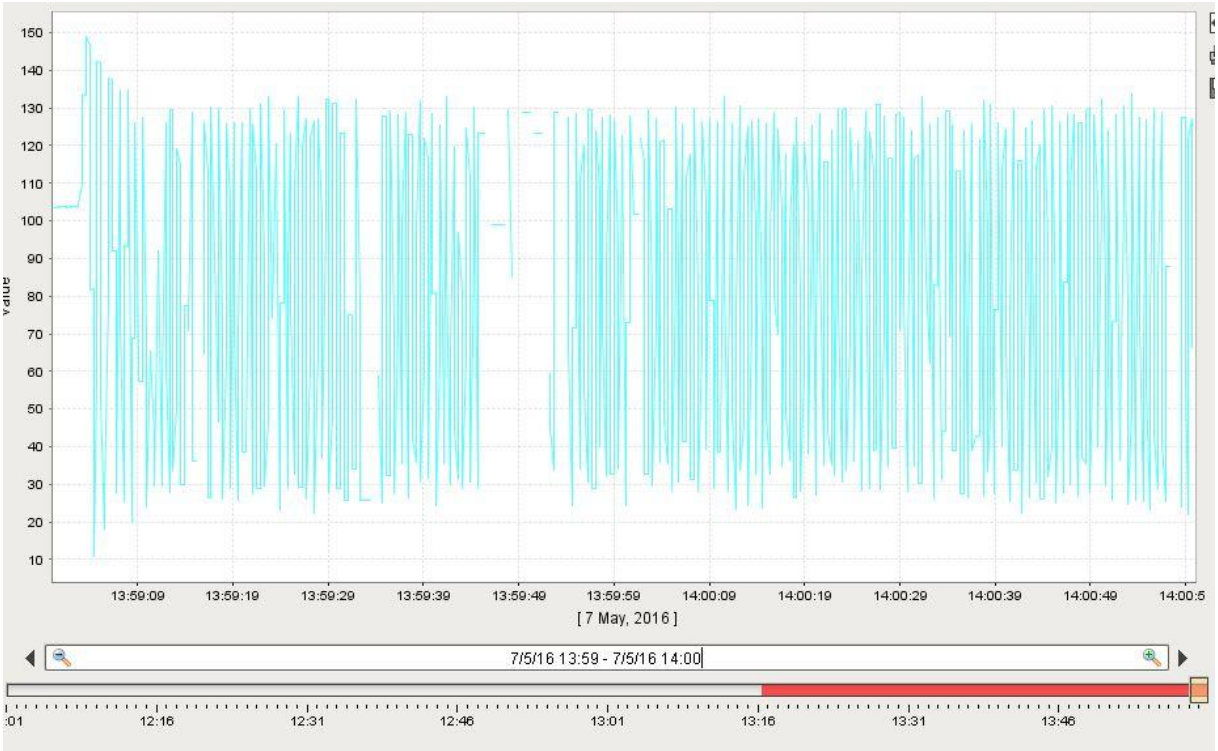
Graafiku üldpilt on korrekte. Mootori pöörlemise jooksul peavad silindrid liikuma algasendisse ja lõppasendisse. Kuna mootori pöörlemissagedus oli vahemikus 0 ... 70 p/min, siis seetõttu on ka graafikult nähtav kolvi asukoha muutuse sagedus suur.



Joonis 4.9. Esimese kolvi asukoha muutuse diagramm.



Joonis 4.10. Teise kolvi asukoha muutuse diagramm.

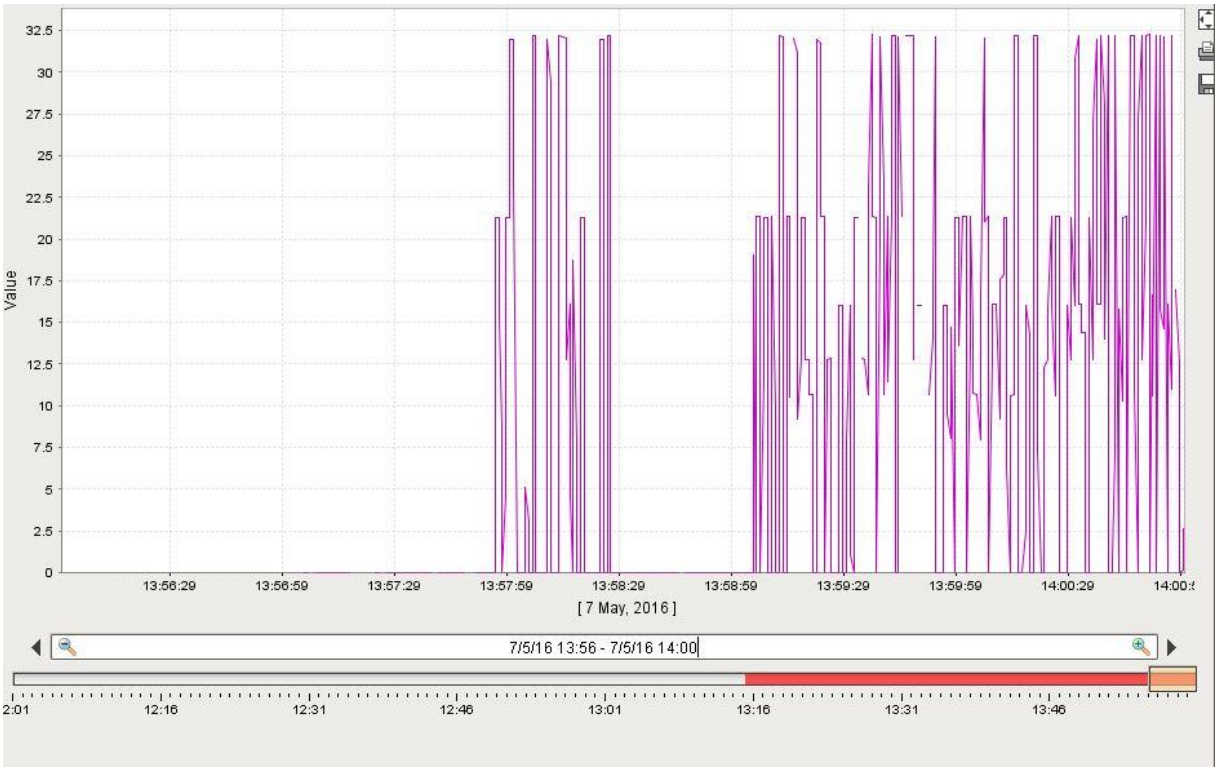


Joonis 4.11. Kolmanda kolvi asukoha muutuse diagramm.

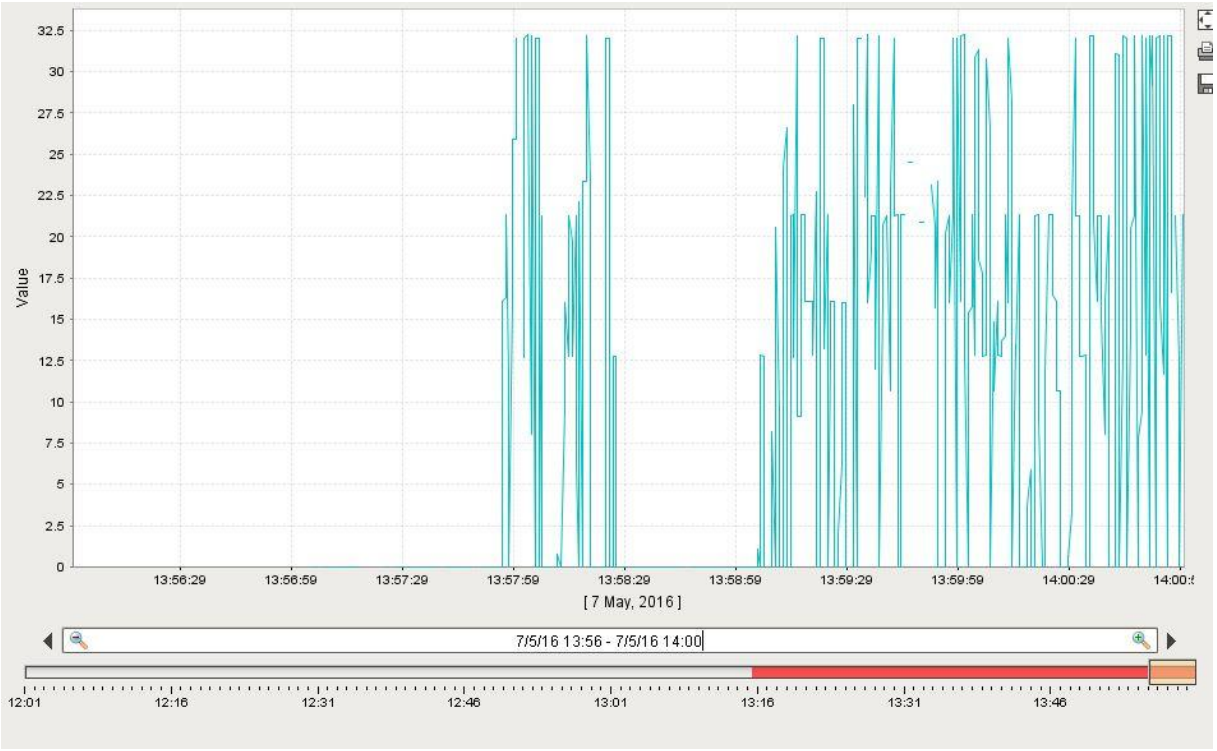
Joonis 4.12. Neljanda kolvi asukoha muutuse diagramm.

Kolvi kiiruse diagramm (joonis 4.13, joonis 4.14, joonis 4.15, joonis 4.16 näitab kolbide liikumise kiiruse muutust masina liikumisel. Graafikule kuvatav andmestik jätab soovida . Graafik peab küll liikuma üles ja alla, aga on näha, et sagedus on liialt suur. Kolbide liikumise kiirus saab muutuda vaid juhul, kui lähenetakse kolbi lõpp või algasendile. Graafikutelt on ka näha, et palju andmeid on läinud kaotsi

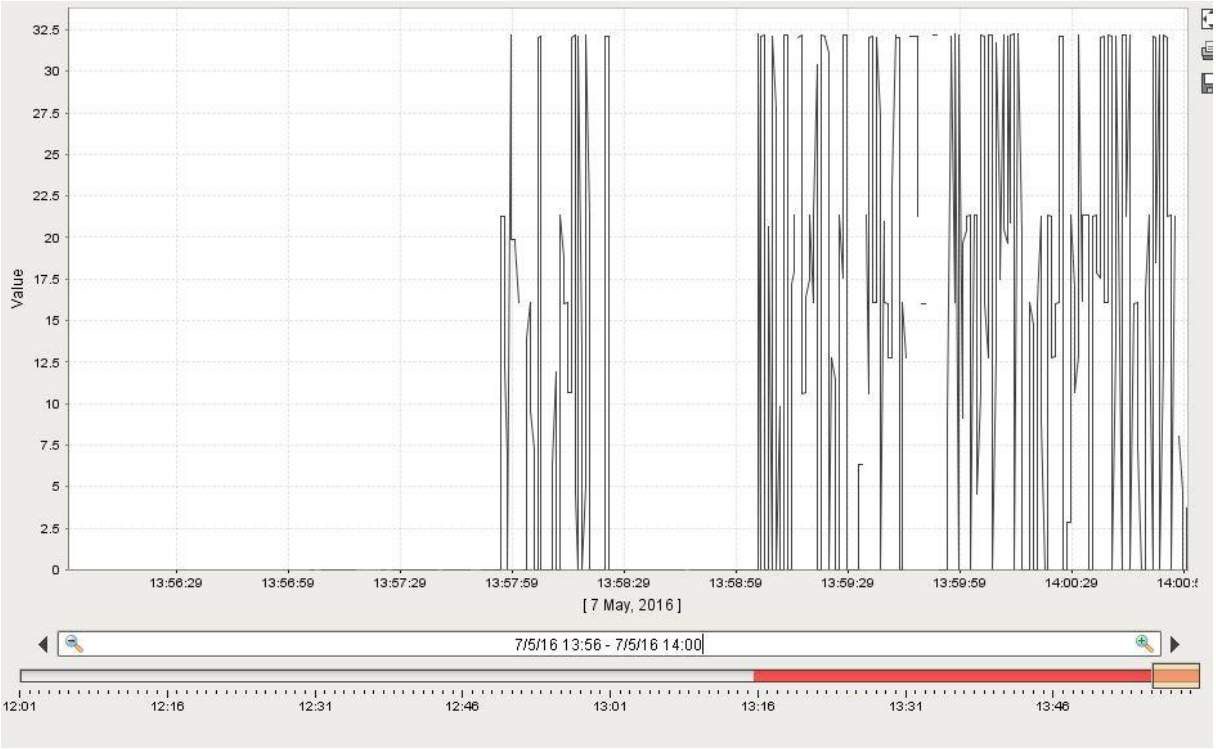
Joonis 4.13. Esimese kolvi liikumise kiirus.



Joonis 4.14. Teise kolvi liikumise kiirus.



Joonis 4.15. Kolmanda kolvi liikumise kiirus.



Joonis 4.16. Neljanda kolvi liikumise kiirus.

Masina liikumiskiiruse graafik (joonis 4.17) näitab masina liikumise kiiruse muutust sõidu jooksul. On näha suurt kiiruse kõikumist, mis ei ole põhjustatud masina liikumise kiiruse kiirest muutumisest vaid väljendab ebapiisavat kiiruse väärtuse stabiliseerimist. Sarnaselt kolvi kiiruse graafikutele on masina liikumise graafikul samuti märgata suurt andmekadu. Andmekadu võib olla tingitud, kas *WiFi* ühenduse probleemidest, mis *SCADA* programmi jälgides kuidagi ei väljendunud.

Korrektseks osaks võib lugeda masina liikumiskiiruse kasvu hetkel mil aeg on 13:59:00. Tõus väljendab hetke, mil masin alustas sõitu. Arvestades, et võistlusmasin kulutas suruõhku väga kiiresti, siis lühike 1 minutiline sõit võib olla korrektne. Võistluse ajal jäi masina maksimaalne liikumise kiirus vahemikku 13 ... 15 km/h.

Joonis 4.17. Masina liikumise kiiruse graafik.

Kuna *SCADA* tarvata võtab graafikule kuvamiseks andmed *MySQL* andmebaasist, võib järeldada, et kolbide positsiooni diagramm ja kolbide jõudude diagrammil on kuvatud andmeid ka varasematest masina liikumistest. Kui võtta arvesse ajavahemikku 13:59:0 kuni 14:00:00, siis andmete kadu ei ole liialt suur.

Kõige õigem on aluseks võtta masina liikumiskiiruse diagramm. Kui masina liikumiskiiruse diagrammiga viia vastavusse kõik teised graafikud, võib üldpilt paremaks muutuda.

KOKKUVÕTE

Eesmärgid, mis Pneumosõidukite automaatjuhtimist projekteerides püstitati, täideti nõuetekohaselt. Automaatjuhtimises ja telemeetria süsteemis esines puudusi, mis küll ei takistanud võistlusel osalemist, kuid vähendasid konkurentsi võimet.

Seadmete valik oli korrektne ja masinale juhttoime avaldamiseks olid olemas kõik vajalikud pneumaatika komponendid. Matemaatiline funktsioon pneumosõiduki rõhu arvutamiseks kiiruse hoidmiseks loodi, kuid *IndraLogic* programmeerimise keskonna poolt pakutud PID regulaator töötas ja oli vastavalt vajadusele seadistatav. Seega matemaatilist funktsiooni ei kasutatud.

Masina mootori töös tõrkeid ei esinenud, seega loodud juhtprogramm töötas hästi. Suruõhu optimaalset kasutust programmiliselt saavutada sai vaid hästi seadistatud PID regulaatoriga püsival kiirusel liikudes. *SCADA* süsteem kogus andmeid ja oli võimalik täitureid distantsjuhtida.

Automaatne rõhu reguleerimise süsteem PID regulaatoriga oli korrektselt tööle seadistamata ja *Technics 1* võistlusmasinal seda kasutada ei saanud. Sarnane süsteem oli programmeeritud ka *Technics 2* võistlusmasinale, kus juhi sõnul automaatreguleerimise süsteem töötas probleemideta. Süsteemi tööpõhimõtteks oli hoida juhi poolt seadistatud kiirust. Nupu vajutusega alustas *PID* regulaator masina liikumise hetkekiirust hoidma. Puuduse põhjuseks oli ajapuudus

Masina esitluse ajal tutvustatud Kaugjuhtimine töötas tõrgeteta. Kaugjuhtimise teel sai reguleerida rõhku, lülitada tööle mootorit ja lugeda andurite väärtuseid. Halvasti seadistatud *PID* reguleerimise tõttu ei töötand kaugjuhtimise teel sisse lülitatud automaatne rõhu reguleerimine mootori kiiruse hoidmiseks.

Telemeetria süsteemi toimimiseks kasutusele võetud *WiFi* ühenduse süsteem töötas tõrgeteta. Signaali tugevus oli piisavalt tugev, seega ühendust ei häirinud ka võistlusalal paiknenud puud. Pneumosõiduki võistlus andmed saadi soovitud kujul tabelis ja graafikul, kuid

vaatamata sellele olid andmed graafikus segasel kujul, seega vajab süsteem seadistamiseks lisa aega.

Andmete kuvamisel graafikul reaalajas oli probleemiks liialt tihe andmepilv. Lahenduseks probleemile tuleb *SCADA* programmis muuta anduri väärtuse muudu piir väärtust, mil uus andmepunkt graafikul näidatakse. Taoline muudatus vähendab andmehulka ja selle tulemusel on graafiku üldpilt loetavam.

SUMMARY

Following research work has been created to build a automatic control system and telemetry for the penumatic vehicle. Every year in Hungary city of Eger students have builted a four or three wheel vehicle witch energy source is 10 L at 200 bar preassure tank. First competition was in year 2008. Estonian University of Life Sciences has been competite three years. Year 2016 university students builted three vehicles. All vehicles have its unique description. Technics 1 vehicle wicht this reseach talks about has radial pneumatic engine, Technics 2 vehicle has much simple engine but aluminium monocoque frame, Technics 3 vehicle is three wheeled and has simple tooht rod motor.

All created achivments has been achived:

1. Technics 1 Pnemobile competited in race
2. All automation systems workd fine
3. Telemetry system collected data
4. Vehicle was remote controlled over SCADA software

Technics 1 automation system compared with other two vehicles is very complex. PLC has same amount of inputs and outputs as simplier automated industry. Problem why Technics 1 vehicle need so much automatic control is about radial motor. Radial motor need exact positioning and every silinder must have seperated air switching system. That means every input of two way cylinder must have its own valve. Problem is that inputs what must have no air input at moment needs to move free.

Radial motor is absolute encoded. Motor has no zeroing position so PLC must know the position even the power is lost. Absolute encoder communicates over SSI interface. Motor has 8 pneumatic solenoid valves and four absolute position sensors of the pneumatic cylinders. Cylinder position sensors are needed to collect position data of the cylinders and calculate cylinder speed.

Telemetry system is wireless. Wireless connection has been establised with Wi-Fi router and acess point. Wi-Fi router is connected top of the pneumatic vehicle and acess point with

laptop. Over ethernet protocols its very easy to establish PLC connection because PLC has onboard ethernet connection. Communication protocol witch is used to establish connection wiht PLC and scada software is Modbus TCP. All the data witch is collected during the race is saved in MySQL database and later has been downloaded in .CSV (Coma separated values) Microsoft Excel supported format.

KIRJANDUS

1. IX. International Aventics Pneumobile Competition (2016), Kättesaadav http://en.pneumobil.hu/pneumobile_2016/announcement_for_the_competition/announcement_and_rules_for_the_competition

2. Product Catalog Pneumatics, Kättesaadav: http://www.aventics.com/pneumatics-catalog/Vornavigation/VorNavi.cfm?Language=EN&Variant=internet&PageID=g53567

3. Electric Drives and Controls Product Catalog (2015), Kättesaadav: http://www.boschrexroth.com/dcc/Vornavigation/VorNavi.cfm?Language=EN&VHist=&PageID=p186794

4. Lisamoodulid (tabel), Kättesaadav: http://www.boschrexroth.com/dcc/Vornavigation/Vornavi.cfm?Language=EN&Variant=internet&VHist=g97568,g96076&PageID=g96669

5. Rexroth Inline Terminal With Eight Analog Input Channel (2005), Kättesaadav: https://md.boschrexroth.com/modules/BRMV2PDFDownload-internet.dll/r911170486_02.pdf?db=brmv2&lvid=1138521&mvid=12115&clid=20&sid=BA257758E4156E2395BA39DE1DEAAAEE.borex-tc&sch=M&id=12115,20,1138521

6. Rexroth Inline Terminal With Two Analog Input Channel (2005), Kättesaadav: https://md.boschrexroth.com/modules/BRMV2PDFDownload-internet.dll/r911170606_01.pdf?db=brmv2&lvid=1138840&mvid=12115&clid=20&sid=BA257758E4156E2395BA39DE1DEAAAEE.borex-tc&sch=M&id=12115,20,1138840

7. Rexroth Inline terminal with one analog output (2014), Kättesaadav: https://md.boschrexroth.com/modules/BRMV2PDFDownload-internet.dll/R911170602_02.pdf?db=brmv2&lvid=1180981&mvid=12115&clid=20&sid=BA257758E4156E2395BA39DE1DEAAAEE.borex-tc&sch=M&id=12115,20,1180981

8. Rexroth Inline terminal for absolute encoders with SSI interface (2009), Kättesaadav: https://md.boschrexroth.com/modules/BRMV2PDFDownload-internet.dll/R911170584_01.pdf?db=brmv2&lvid=1146212&mvid=12115&clid=20&sid=BA257758E4156E2395BA39DE1DEAAAEE.borex-tc&sch=M&id=12115,20,1146212

9. **Manukid, P., Charoen, N.** (2001) Kinematics control of a pneumatic system by hybrid fuzzy PID. – *Mechatronics*, lk 1004.

10. PLC Programming with Rexroth Indralogic 1.0 (2004: 11-12), Kättesaadav: https://md.boschrexroth.com/modules/BRMV2PDFDownload-internet.dll/30503601.pdf?db=brmv2&lvid=89633&mvid=12115&clid=20&sid=BA257758E4156E2395BA39DE1DEAAAEE.borex-tc&sch=M&id=12115,20,89633

11. PLC Programming with Rexroth Indralogic 1.0 (2004: 29-33), Kättesaadav: https://md.boschrexroth.com/modules/BRMV2PDFDownload-internet.dll/30503601.pdf?db=brmv2&lvid=89633&mvid=12115&clid=20&sid=BA257758E4156E2395BA39DE1DEAAAEE.borex-tc&sch=M&id=12115,20,89633

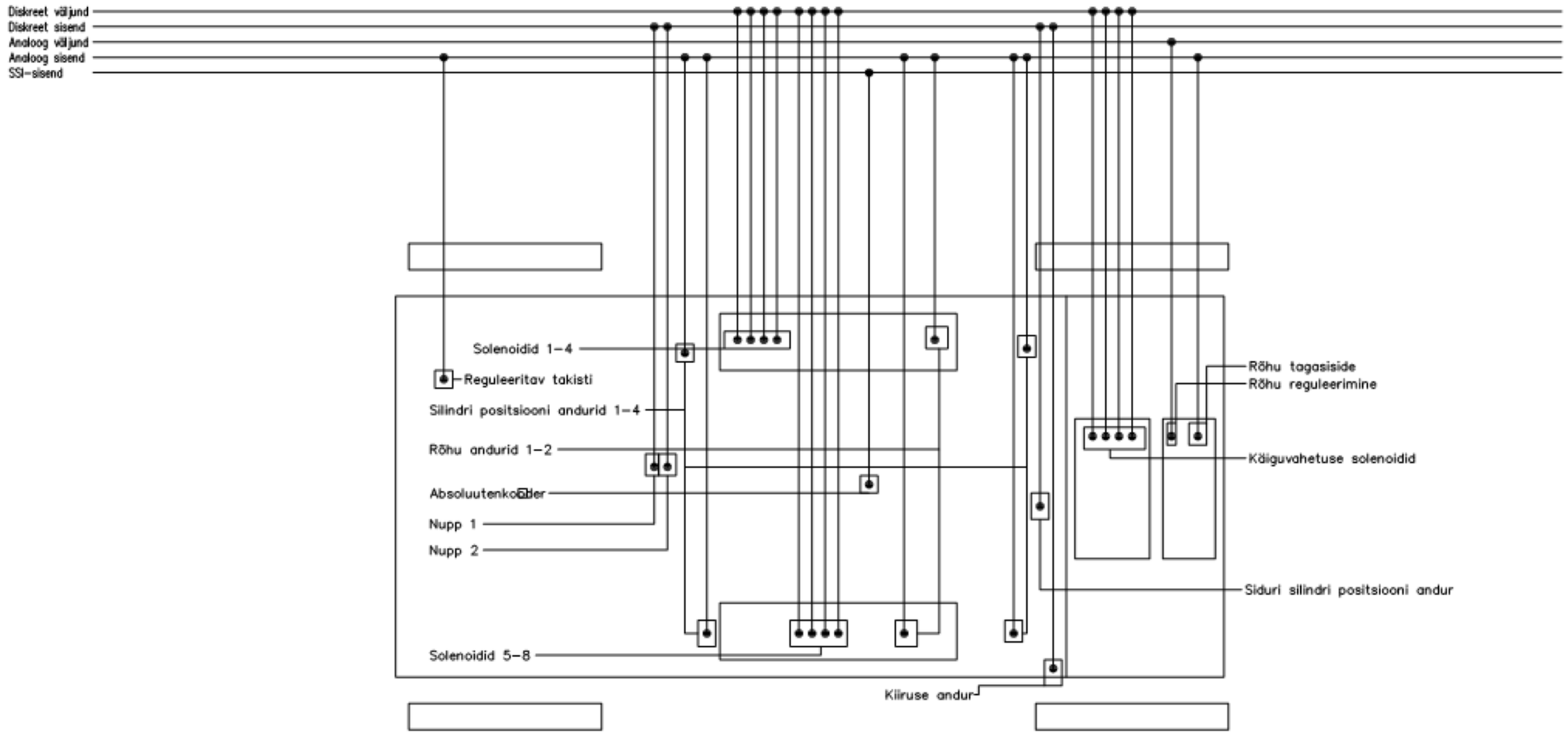
12. AFS60/AFM60 SSI Absolute Encoders (2012), Kättesaadav: https://www.mysick.com/saqqara/im0030507.pdf

13. **Kyoungkwan, A., Shinichi, Y.** (2005) Intelligent switching control of pneumatic actuator using on/off solenoid valves. – *Mechatronics*, lk 689.

14. **Karl-Heinz John. , M. Tiegelkamp** (2010) IEC 61131-3: Programming Industrial Automation System. 390lk

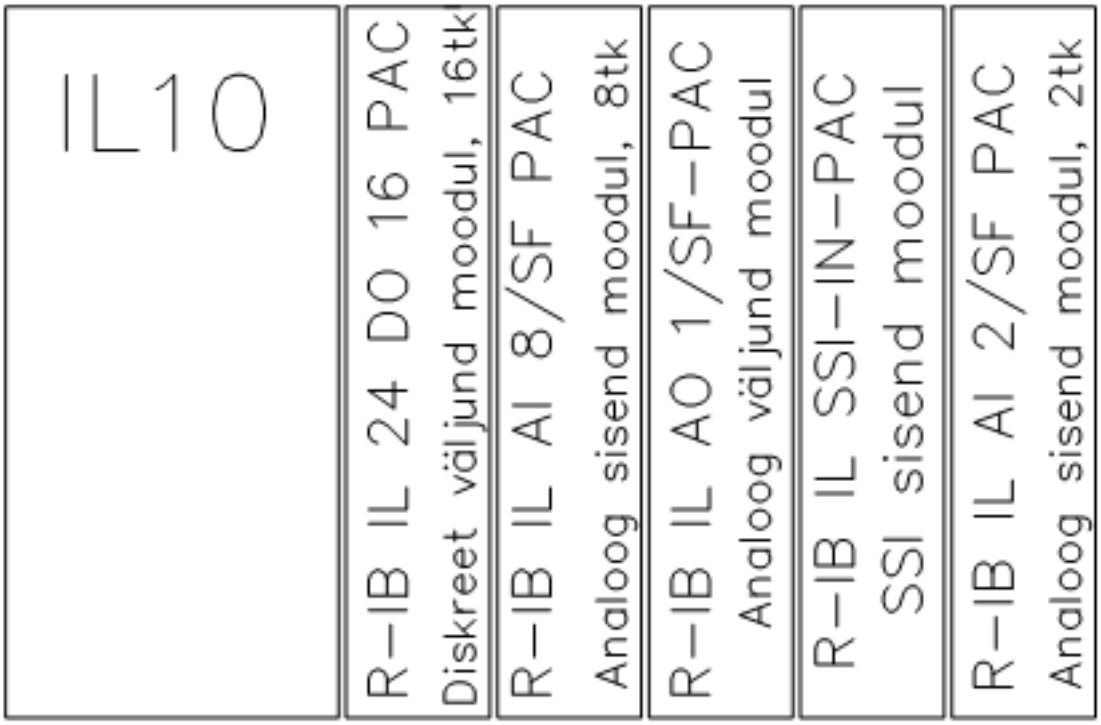
LISAD

LISA A



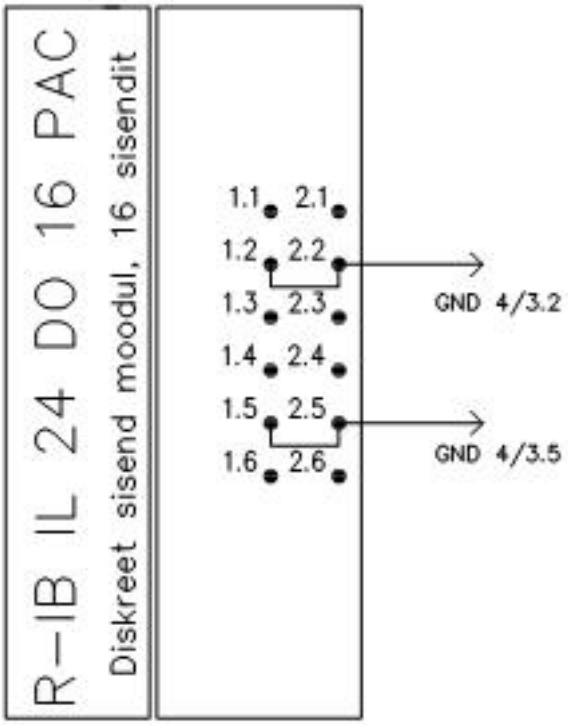
Tellija:		Objekt, seade: Pneumosõiduk, Technics 1
Teostas:	Gert Truu	Nimetus: Sisendite väljundite paiknemine põhiplaanil
Kontrollis:	Erkki Jõgi	
Kinnitas:	Erkki Jõgi	
Eesti Maaülikool EMU Tehnikainstituut		Leht: 1/14 — Tähis: TE 16/090053 A 01 01 S

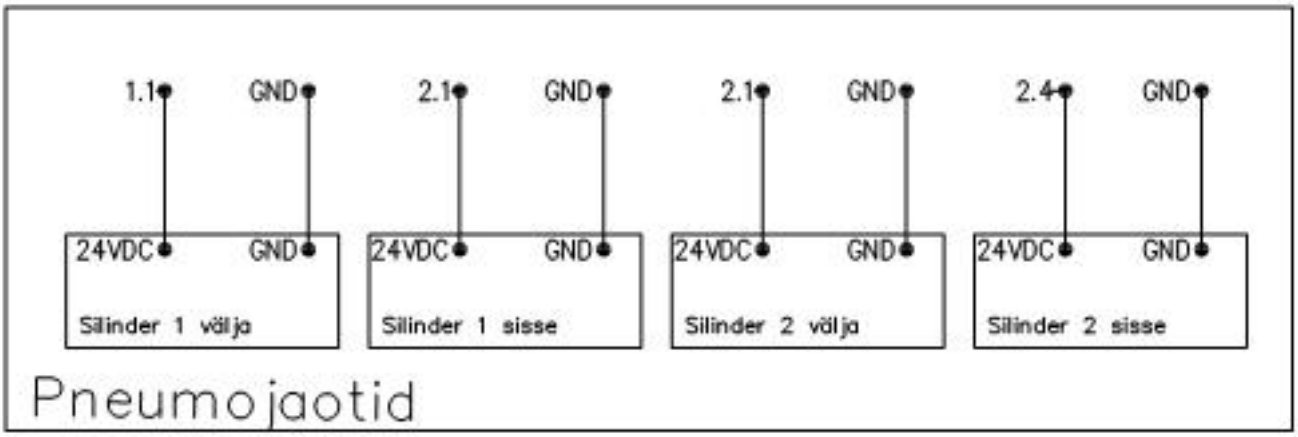
LISA B



Tellija:		Objekt, seade: Pneumosõiduk, Technics 1	
Teostas:	Gert Truu	Nimetus: Loogika kontrolleri külge ühendatud moodulid	
Kontrollis:	Erkki Jõgi		
Kinnitas:	Erkki Jõgi		
Eesti Maaülikool EMU Tehnikainstituut		Leht: 2/14	Tähis: TE 16/090053 B 01 02 S

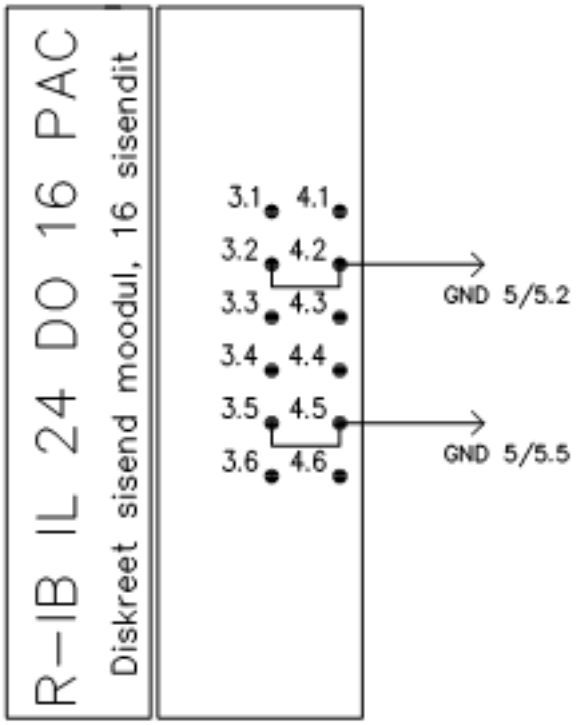
LISA C

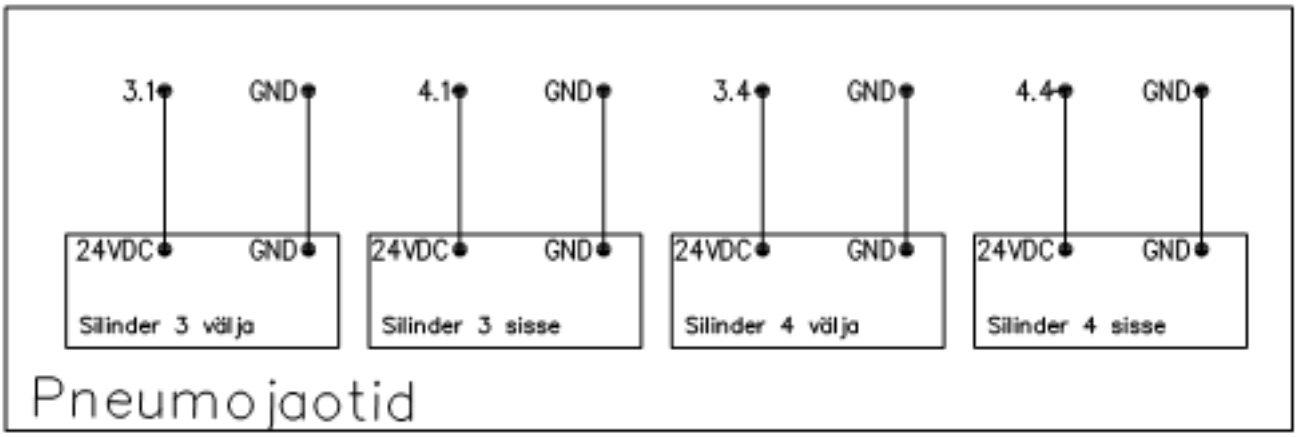




Tellija:		Objekt, seade: Pneumosõiduk, Technics 1	
Teostas:	Gert Truu	Nimetus: Pneumojaotite ühendamise skeem	
Kontrollis:	Erkki Jõgi		
Kinnitas:	Erkki Jõgi		
Eesti Maaülikool EMU Tehnikainstituut		Leht: 3/14	Tähis: TE 16/090053 C 01 03 S

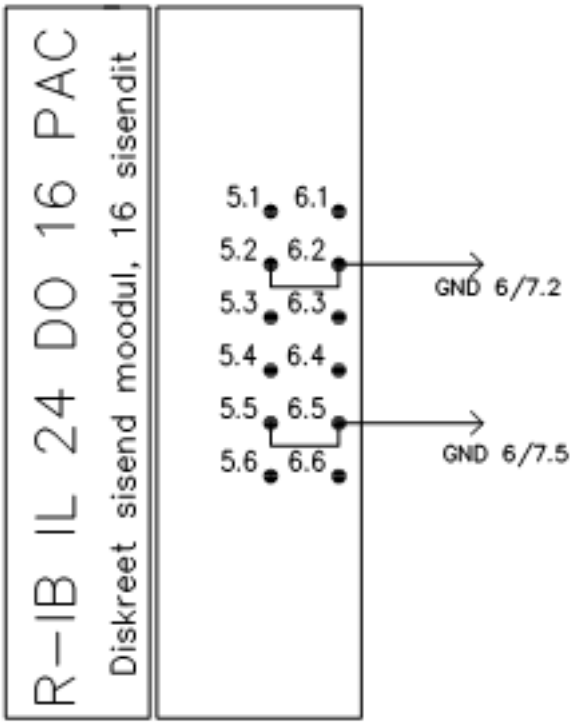
LISA D

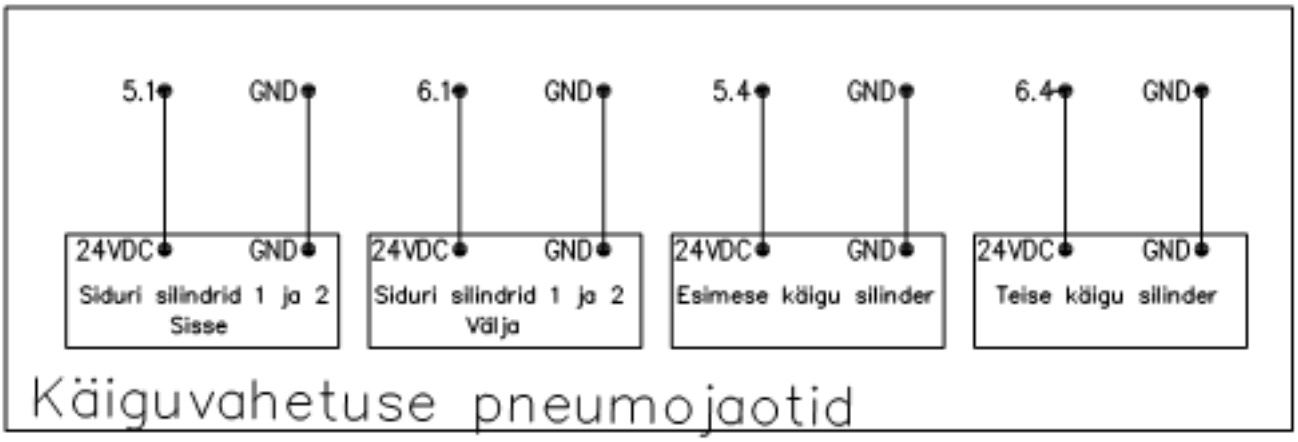




Tellija:		Objekt, seade: Pneumosõiduk, Technics 1	
Teostas:	Gert Truu	Nimetus: Pneumojaotite ühendamise skeem	
Kontrollis:	Erkki Jõgi		
Kinnitas:	Erkki Jõgi		
Eesti Maaülikool EMU Tehnikainstituut		Leht: 4/14	Tähis: TE 16/090053 D 01 04 S

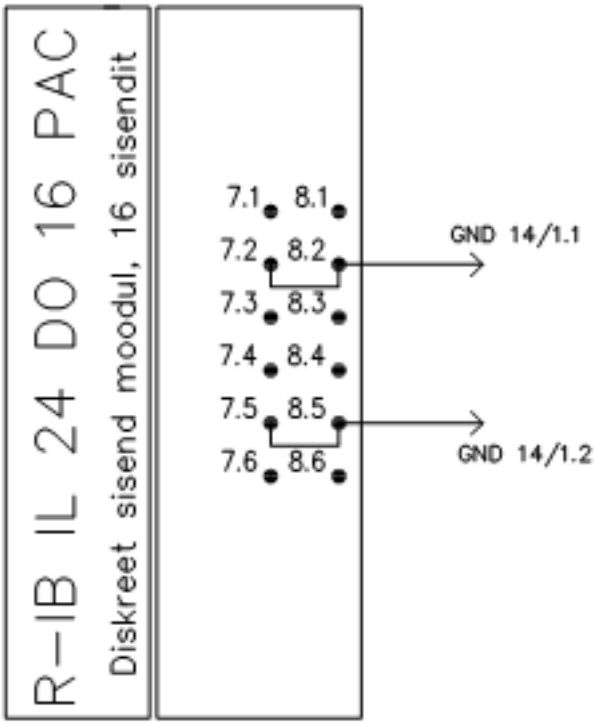
LISA E





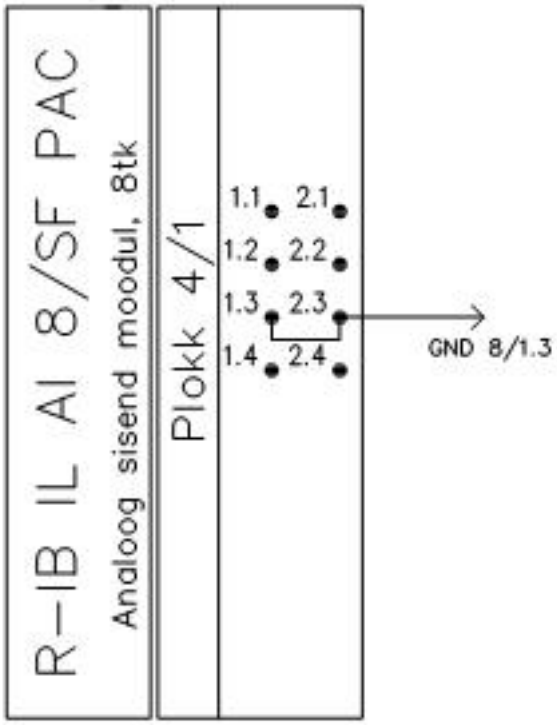
Tellija:		Objekt, seade: Pneumosõiduk, Technics 1	
Teostas:	Gert Truu	Nimetus: Käiguvahetuse süsteem	
Kontrollis:	Erkki Jõgi		
Kinnitas:	Erkki Jõgi		
Eesti Maaülikool EMU Tehnikainstituut		Leht: 5/14	Tähis: TE 16/090053 E 01 05 S

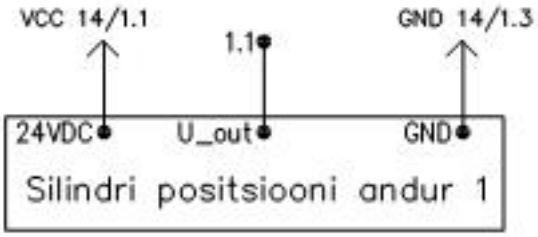
LISA F

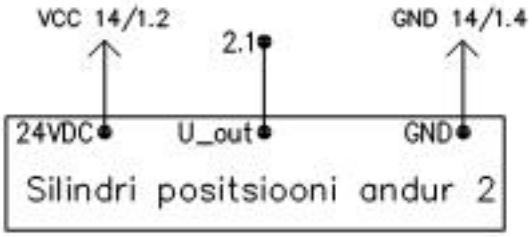


Tellija:		Objekt, seade: Pneumosõiduk, Technics 1	
Teostas:	Gert Truu	Nimetus: Diskreet väljund moodul	
Kontrollis:	Erkki Jõgi		
Kinnitas:	Erkki Jõgi		
Eesti Maaülikool EMU Tehnikainstituut		Leht: 6/14	Tähis: TE 16/090053 F 01 06 S

LISA G

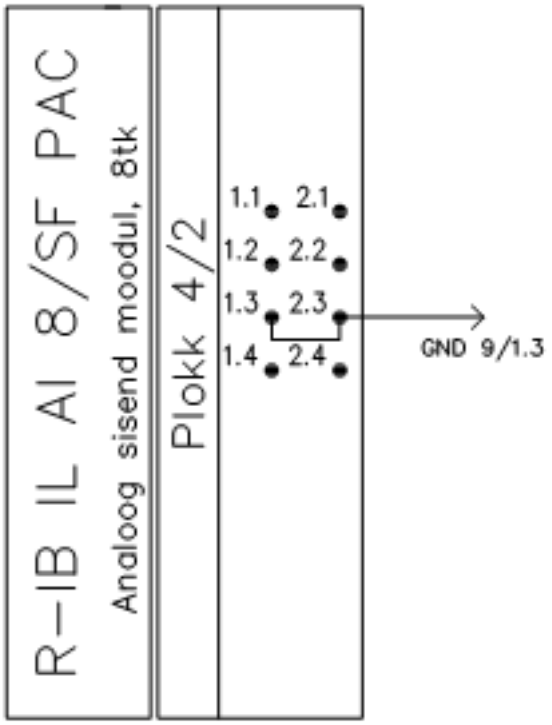


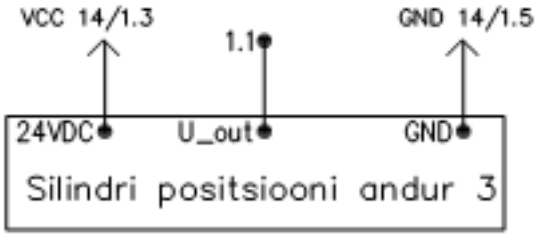


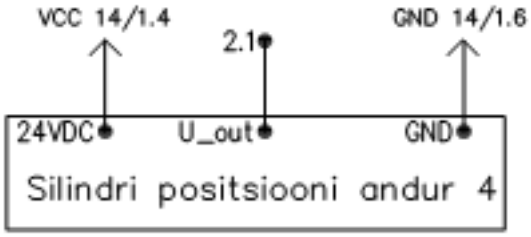


Tellija:		Objekt, seade: Pneumosõiduk, Technics 1	
Teostas:	Gert Truu	Nimetus: Silindri Positsiooni andurid	
Kontrollis:	Erkki Jõgi		
Kinnitas:	Erkki Jõgi		
Eesti Maaülikool EMU Tehnikainstituut		Leht: 7/14	Tähis: TE 16/090053 G 01 07 S

LISA H

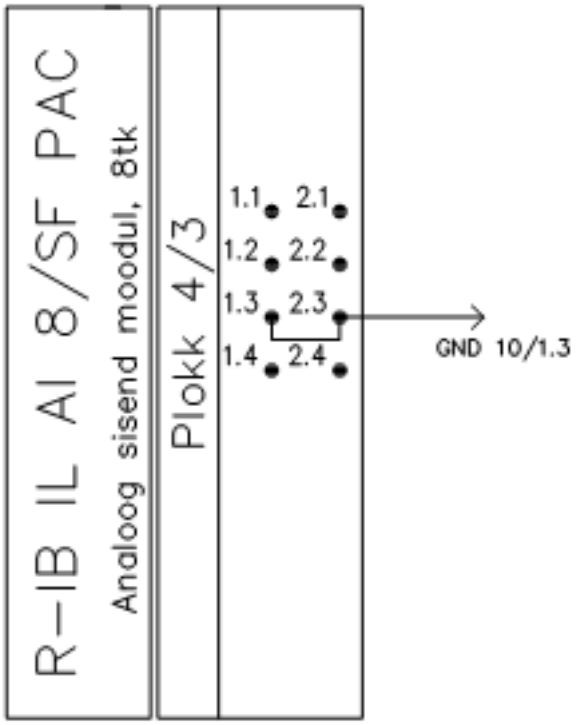


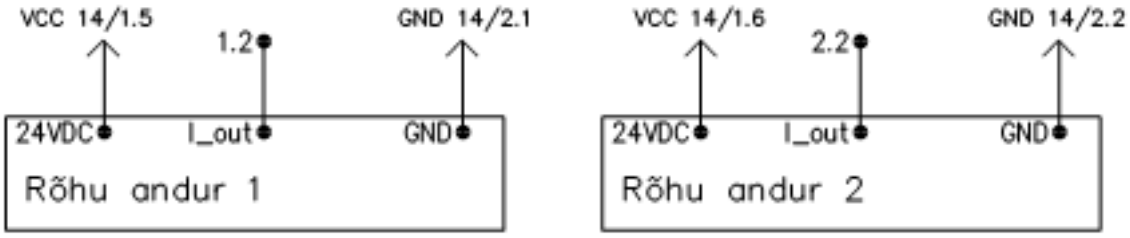




Tellija:		Objekt, seade: Pneumosõiduk, Technics 1
Teostas:	Gert Truu	Nimetus: Silindri positsiooni andurid
Kontrollis:	Erkki Jõgi	
Kinnitas:	Erkki Jõgi	
Eesti Maaülikool EMU Tehnikainstituut		Leht: 8/14 — Tähis: TE 16/090053 H 01 08 S

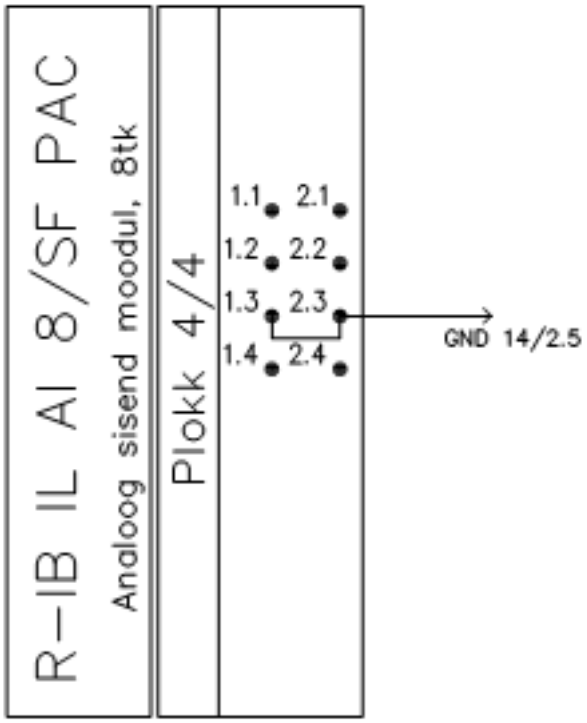
LISA I

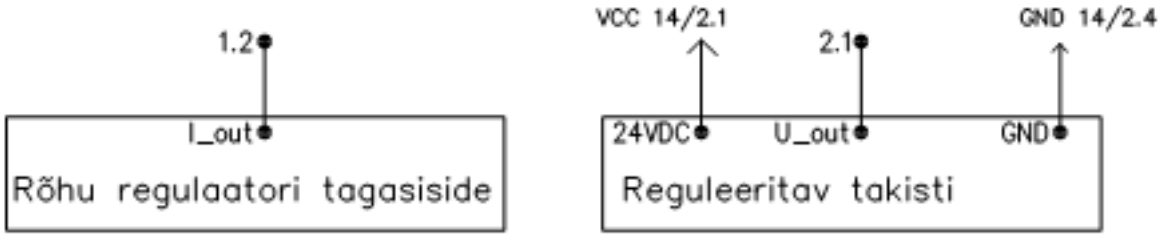




Tellija:		Objekt, seade: Pneumosõiduk, Technics 1	
Teostas:	Gert Truu	Nimetus: Rõhu andurite ühendused	
Kontrollis:	Erkki Jõgi		
Kinnitas:	Erkki Jõgi		
Eesti Maaülikool EMU Tehnikainstituut		Leht: 9/14	Tähis: TE 16/090053 I 01 09 S

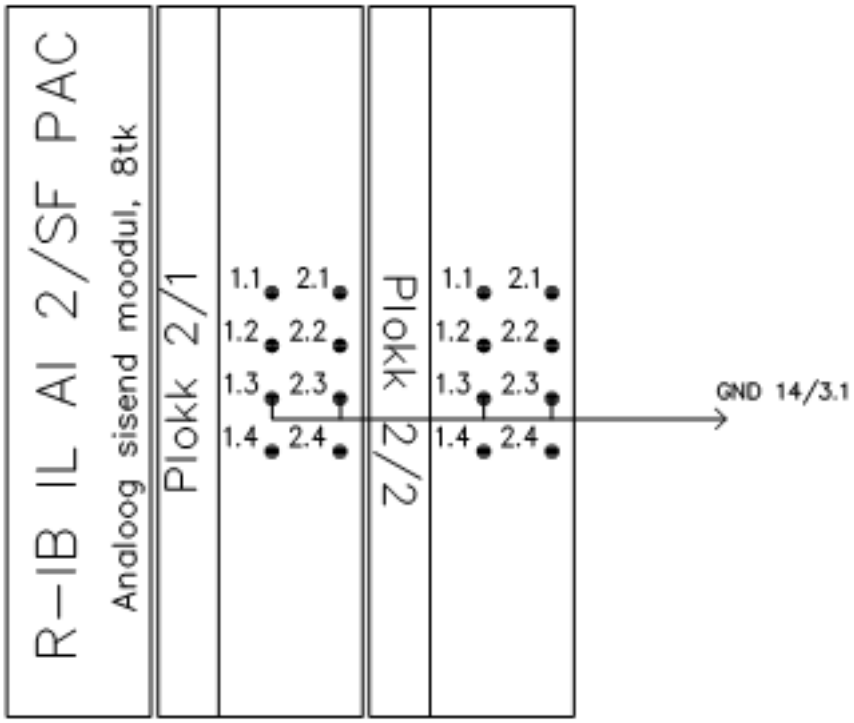
LISA J

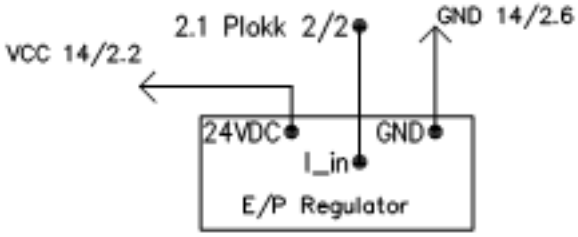




Tellija:		Objekt, seade: Pneumobile, Technics 1
Teostas:	Gert Truu	Nimetus: Rõhu reguleerimise süsteem
Kontrollis:	Erkki Jõgi	
Kinnitas:	Erkki Jõgi	
Eesti Maaülikool EMU Tehnikainstituut		Leht: 10/14 — Tähis: TE 16/090053 J 01 10 S

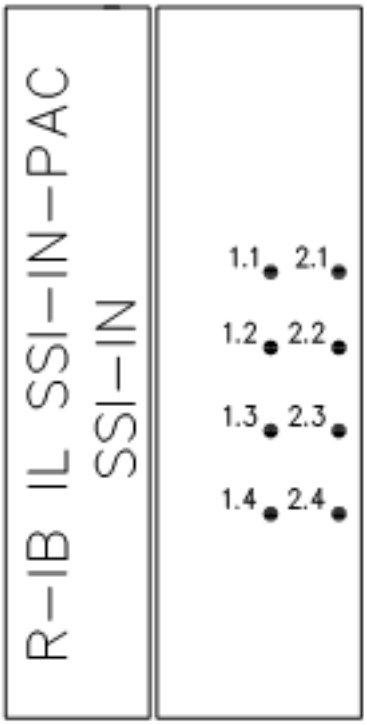
LISA K



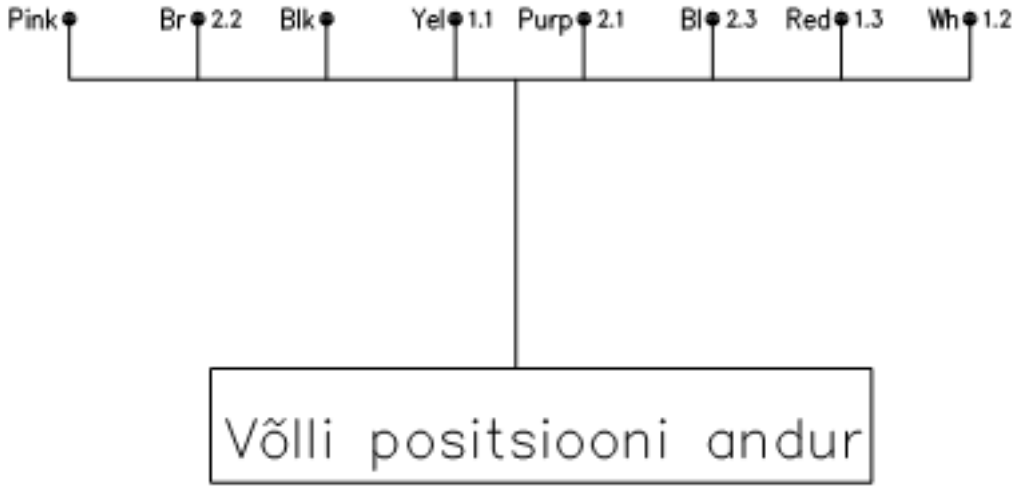


Tellija:		Objekt, seade: Pneumosõiduk, Technics 1	
Teostas:	Gert Truu	Nimetus: Analoog sisend moodul	
Kontrollis:	Erkki Jõgi		
Kinnitas:	Erkki Jõgi		
Eesti Maaülikool EMU Tehnikainstituut		Leht: 11/14	Tähis: TE 16/090053 K 01 11 S

LISA L

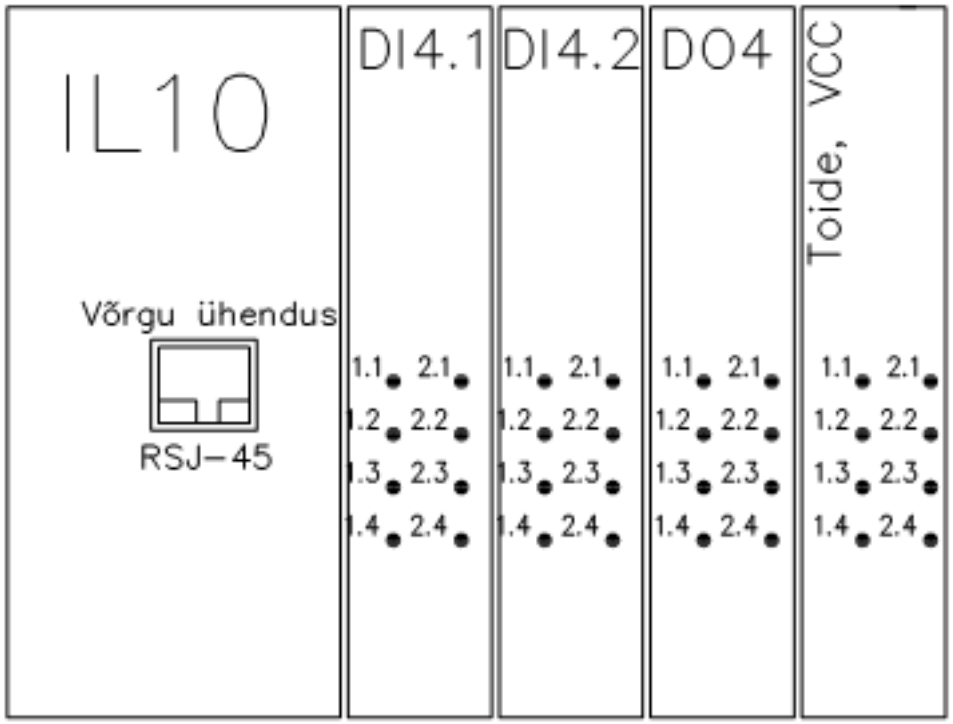


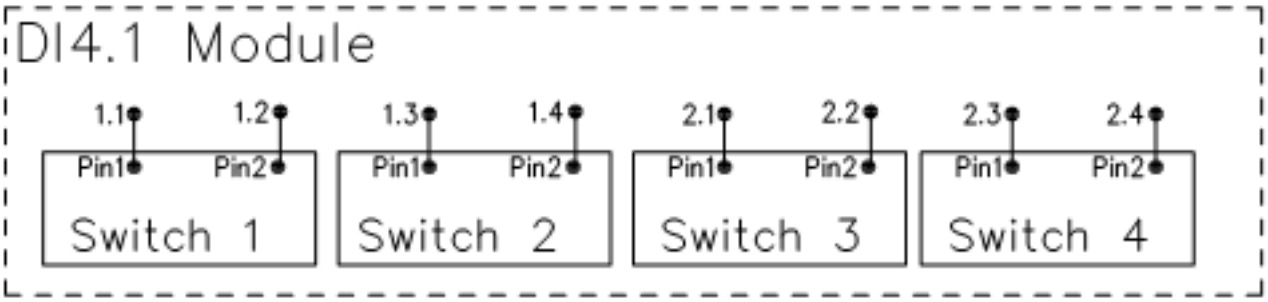
Roosa (Pink) -- SET
Pruun (Br) -- Data-
Must (Blk) -- CW/CCW
Kollane (Yel) -- Clock+
Lilla (Lilac) -- Clock -
Sinine (Bl) -- GND
Punane (Red) -- VCC
Valge (Wh) -- Data+

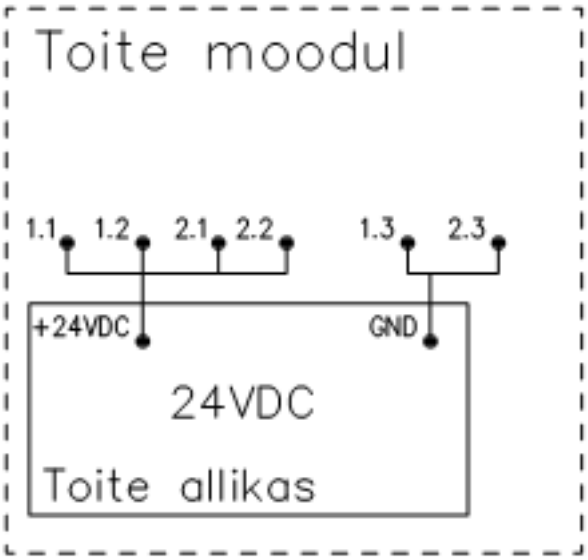


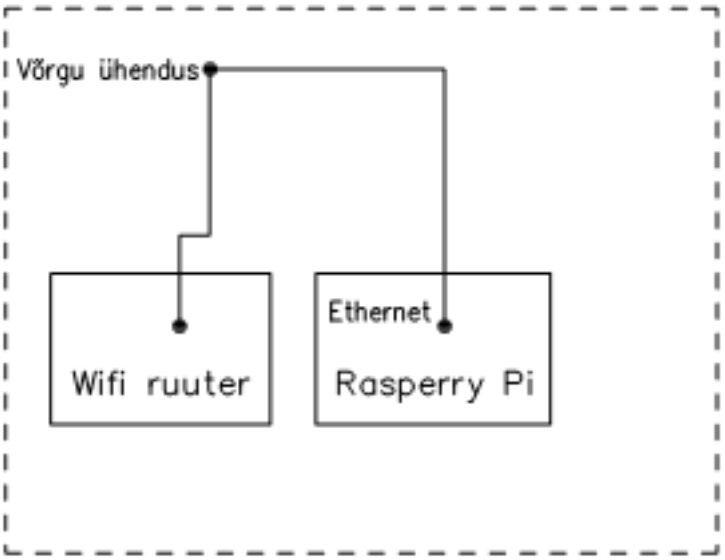
Tellija:		Objekt, seade: Pneumosõiduk, Technics 1	
Teostas:	Gert Truu	Nimetus: Võlli positsiooni anduri ühendamise skeem	
Kontrollis:	Erkki Jõgi		
Kinnitas:	Erkki Jõgi		
Eesti Maaülikool EMU Tehnikainstituut		Leht: 12/14	Tähis: TE 16/090053 L 01 12 S

LISA M



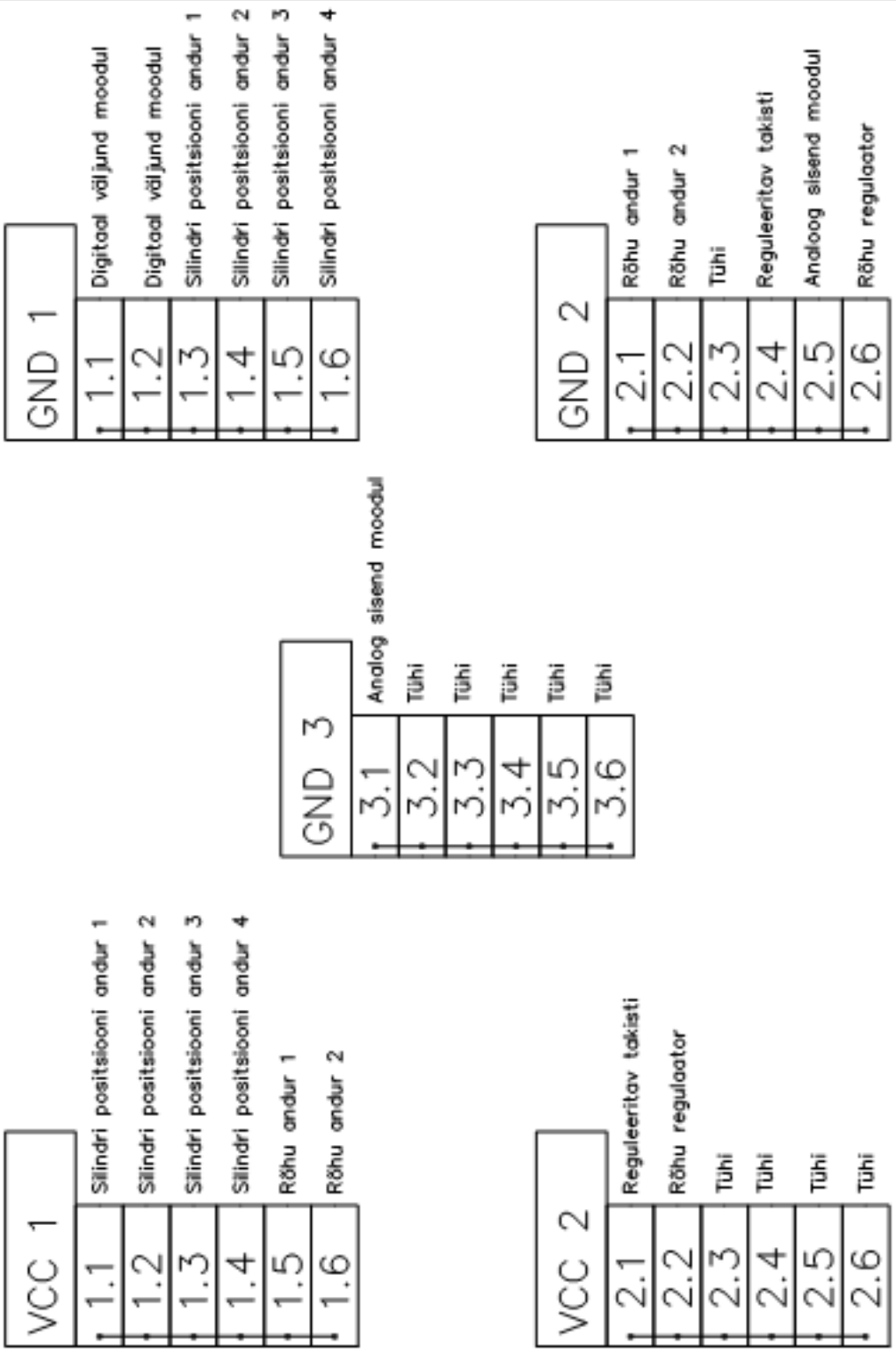






Tellija:		Objekt, seade: Pneumosõiduk, Technics 1	
Teostas:	Gert Truu	Nimetus: Loogikakontrolleri ühendused	
Kontrollis:	Erkki Jõgi		
Kinnitas:	Erkki Jõgi		
Eesti Maaülikool EMU Tehnikainstituut		Leht: 13/14	Tähis: TE 16/090053 M 01 13 S

LISA N



Tellija:		Objekt, seade: Pneumosõiduk, Technics 1	
Teostas:	Gert Truu	Nimetus: Riviklemmide ühendused	
Kontrollis:	Erkki Jõgi		
Kinnitas:	Erkki Jõgi		
Eesti Maaülikool EMU Tehnikainstituut		Leht: 14/14	Tähis: TE 16/090053 N 01 13 S

LISA O Globaal muutujad

```
VAR_GLOBAL

 C1 AT %QB6                                          :BYTE;
(*
C2.0 = käbi välja
C2.1 = käbi sisse
C2.2 = ülemine pidur
C2.3 = alumine pidur
*)
C2                          AT %QB7                 :BYTE
Clutch                      AT %IX0.0               :BOOL;
G1                          AT %QX7.2               :BOOL := FALSE;
G2                          AT %QX7.3               :BOOL := FALSE;
ManualAuto                  AT %IX0.7               :BOOL;
        Lin_Pos_sens                        : ARRAY [0..3] OF REAL;
        Lin_Pos_sens2                       : ARRAY [0..3] OF WORD;
        Preassure                           :ARRAY[0..2]  OF REAL;
        PSI_OUT                                         :WORD;
        PotVal                                          :REAL;
        FlowRate                                        :REAL;
        Gear                                            :REAL:=0;
        Gear_F                                          :REAL;
        LastGear                                        :WORD:=0;
        Temp                                 : ARRAY [0..1]  OF REAL;
        A_OUT0              AT %QW8                     :WORD;
        A_OUT1              AT %QW10                    :WORD;
        A_IN0               AT %IW2                     :WORD;
        A_IN1               AT %IW4                     :WORD;
        AnalogVAL           AT %IW4                     :WORD;
        A2_OUT0             AT %QW16                    :WORD;
        A2_OUT1             AT %QW18                    :WORD;
        A2_IN0              AT %IW10                    :WORD;
        A2_IN1              AT %IW12                    :WORD;
        AO_OUT0             AT %QW2                     :WORD;
        P_S_EP_Write                                    :WORD;
        ENC_O0              AT %QW12                    :WORD;
        ENC_O1              AT%QW14                     :WORD;
        ENC_I0              AT %IW6                     :WORD;
        ENC_I1              AT %IW8                     :WORD;
        ENC_I1_R            AT %IW8                     :REAL;
        ENCODER                                         :REAL;
       Overide                                          :WORD;
       Overide_F                                        :WORD;
       Drive                                            :REAL;
       Drive_F                                          :REAL;
       PSI_read                                         :REAL;
       PSI                                              :REAL;
```

```
        Kiirused                                    : ARRAY [0..5]  OF   REAL;
        runStop                                                      :BOOL;
        runStop_F                                                    :BOOL;
        runStop_F2                                                   :BOOL;
        ManualAuto_F                                                 :BOOL;
        PID_F                                                        :BOOL;
        PID_S                                                        :BOOL;

        airflow                                                     :REAL;
        airTank                                                     :WORD := 65;
        SpeedSet                                                    :WORD;
        SpeedSet2                                                   :WORD;
        SpeedSet_F                                                  :REAL;
        F1                                                          :REAL;
        F2                                                          :REAL;
        F3                                                          :REAL;
        F4                                                          :REAL;
        T1                                                          :REAL;
        V2                                                          :REAL;

END_VAR
```

LISA P Põhiprogramm

(Muutujate defineerimise algus)

```
PROGRAM PLC_PRG
VAR

(*Funktsioonid*)
        Motor: Motor;
        GearChange:GearChange;
        Kiirused: MotorMove;
        AnalogRead:AnalogRead;
        Encoder:EncoderRead;
        AnalogOUT: AnalogOutput;
 (*Funktsioonid lõpp*)
END_VAR
(Muutujate defineerimise lõpp)
```

(Programmi algus)

```
AnalogRead;
Encoder;
AnalogOUT;
Kiirused;
GearChange;
IF PotVal > 75 OR AnalogOUT.L = 1 OR GearChange.Change = TRUE OR RunStop_F =
TRUE THEN
Motor;
runStop_F2 := TRUE;
ELSIF PotVal < 75 OR GearChange.Change = FALSE OR RunStop_F = FALSE THEN
runStop_F2 := FALSE;
F1:=0;
F2:=0;
F3:=0;
F4:=0;
END_IF
(*Tagasiside kuvamine*)
IF runStop = TRUE THEN
runStop_F := TRUE;
ELSE
runStop_F := FALSE;
END_IF
IF ManualAuto = TRUE THEN
ManualAuto_F := TRUE;
ELSE
ManualAuto_F := FALSE;
END_IF
IF PID_S = TRUE THEN
PID_F := TRUE;
ELSE
PID_F := FALSE;
END_IF
```

```
IF drive = 1 THEN
drive_F := 1;
ELSIF drive = 2 THEN
drive_F := 2;
ELSE
drive_F := 0;
END_IF
(*Tagasiside kuvamise lõpp*)
```
(Programmi lõpp)

LISA Q Pneumojaotite lülitamise funktsiooniplokk

(Muutujate defineerimine)

```
FUNCTION_BLOCK Motor
VAR_INPUT
END_VAR
VAR_OUTPUT
        Period:INT;
END_VAR
VAR
END_VAR
```

(Muutujate defineerimise lõpp)

(Põhiprogrammi algus)

```
IF ENCODER>355 AND ENCODER<=359.9 THEN
C1:= 72;
F1 := 0.0;
F2 := (Preassure[1] * 100000.0) * (((0.050*0.050) * 3.14 / 4) - ((0.020*0.020) * 3.14 /4));
F3 := 0.0;
F4 := (Preassure[2] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
ELSIF ENCODER>0 AND ENCODER<=5 THEN
C1:= 72;
F1 := 0.0;
F2 := (Preassure[1] * 100000.0) * (((0.050*0.050) * 3.14 / 4.0) - ((0.020*0.020) * 3.14 /4.0));
F3 := 0.0;
F4 := (Preassure[2] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
ELSIF ENCODER>=6 AND ENCODER<=85 THEN
C1:=105;
F1 := (Preassure[1] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
F2 := (Preassure[1] * 100000.0) * (((0.050*0.050) * 3.14 / 4.0) - ((0.020*0.020) * 3.14 /4.0));
F3 := (Preassure[2] * 100000.0) * (((0.050*0.050) * 3.14 / 4.0) - ((0.020*0.020) * 3.14 /4.0));
F4 := (Preassure[2] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
ELSIF ENCODER>=86 AND ENCODER<=95 THEN
C1:=33;
F1 := (Preassure[1] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
F2 := 0.0;
F3 := (Preassure[2] * 100000.0) * (((0.050*0.050) * 3.14 / 4) - ((0.020*0.020) * 3.14 /4.0));
F4 := 0.0;
ELSIF ENCODER>=96 AND ENCODER<=175 THEN
C1:=165;
F1 := (Preassure[1] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
F2 := (Preassure[1] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
F3 := (Preassure[2] * 100000.0) * (((0.050*0.050) * 3.14 / 4.0) - ((0.020*0.020) * 3.14 /4.0));
F4 := (Preassure[2] * 100000.0) * (((0.050*0.050) * 3.14 / 4.0) - ((0.020*0.020) * 3.14 /4.0));
ELSIF ENCODER>=176 AND ENCODER<=185 THEN
C1:=132;
F1 := 0;
F2 := (Preassure[1] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
F3 := 0;
F4 := (Preassure[2] * 100000.0) * (((0.050*0.050) * 3.14 / 4.0) - ((0.020*0.020) * 3.14 /4.0));
```

```
ELSIF ENCODER>=186 AND ENCODER<=270 THEN
C1:=150;
F1 := (Preassure[1] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
F2 := (Preassure[1] * 100000.0) * (((0.050*0.050) * 3.14 / 4.0) - ((0.020*0.020) * 3.14 /4.0));
F3 := (Preassure[2] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
F4 := (Preassure[2] * 100000.0) * (((0.050*0.050) * 3.14 / 4.0) - ((0.020*0.020) * 3.14 /4.0));
ELSIF ENCODER>=271 AND ENCODER<=280 THEN
C1:=18;
F1 := (Preassure[1] * 100000.0) * (((0.050*0.050) * 3.14 / 4.0) - ((0.020*0.020) * 3.14 /4.0));
F2 := 0.0;
F3 := (Preassure[2] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
F4 := 0.0;
ELSIF ENCODER>=281 AND ENCODER<=355 THEN
C1:=90;
F1 := (Preassure[1] * 100000.0) * (((0.050*0.050) * 3.14 / 4.0) - ((0.020*0.020) * 3.14 /4.0));
F2 := (Preassure[1] * 100000.0) * (((0.050*0.050) * 3.14 / 4.0) - ((0.020*0.020) * 3.14 /4.0))
;
F3 := (Preassure[2] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
F4 := (Preassure[2] * 100000.0) * ((0.050*0.050) * 3.14 / 4.0);
END_IF
```

(Põhiprogrammi lõpp)

LISA R Käigu vahetamise süsteem

(Muutujate defineerimise algus)

```
END_VAR
VAR_OUTPUT
END_VAR
VAR
        TONInst         :TON;
        TONInst2        :TON;
        TOF2            :TOF;
        TS              :BOOL:=FALSE;
        TS2             :BOOL;
        Q1              :BOOL:=FALSE;
        Q2              :BOOL:=FALSE;
        LG              :BOOL;
        Step            :INT:=0;
        Change          :BOOL:= FALSE;
        VarBool1        :BOOL;
        VarBool2        :BOOL;
        JF              :BOOL;
        Gear_set        :INT;
END_VAR
```

(Muutujate defineerimise lõpp)

(Põhiprogrammi algus)

```
IF Gear_set = 0 AND Clutch = TRUE  THEN
Gear_F := 0;
ELSIF Gear_set = 1 THEN
Gear_F := 1;
ELSIF Gear_set = 2 THEN
Gear_F := 2;
ELSIF Gear_set = 3 THEN
Gear_F := 3;
END_IF

CASE Step OF
0:
IF %IX0.3 = TRUE THEN
Gear_set := 0;
Step := Step +1;
ELSIF Kiirused[0] <= 5 AND Change = FALSE AND Gear_set <> 1 THEN
Step := Step +2;
Gear_set := 1;
Change := TRUE;
ELSIF Kiirused[0] > 9 AND Kiirused[0] <= 15 AND Change = FALSE AND Gear_set <> 2
THEN
Step := Step +2;
Gear_set := 2;
```

```
Change := TRUE;
ELSIF Kiirused[0] > 15 AND Change = FALSE AND Gear_set <> 3 THEN
Step := Step +2;
Gear_set := 3;
Change := TRUE;
END_IF

1:
C2.0 := TRUE;
C2.1 := FALSE;
G1:=FALSE;
G2:=FALSE;

IF Clutch = FALSE THEN
Step := 0;
Change := FALSE;
END_IF

2:
IF Change = TRUE  THEN
VarBool1 := TRUE;

END_IF

TONInst(IN := VarBOOL1, PT:= T#500ms);
TONInst.Q;
TONInst.ET;
IF TONInst.Q = TRUE AND Gear_set = 1 THEN
Step:= 3;
TONInst(IN := FALSE);
VarBool1 := FALSE;
ELSIF TONInst.Q = TRUE AND Gear_set = 2 THEN
Step := 4;
TONInst(IN := FALSE);
VarBool1 := FALSE;
ELSIF TONInst.Q = TRUE AND Gear_set = 3 THEN
Step := 6;
TONInst(IN := FALSE);
VarBool1 := FALSE;
END_IF

3:(*Esimene käik*)
C2.0 := FALSE;
C2.1 := TRUE;
G1:=FALSE;
G2:=FALSE;
IF Clutch = TRUE THEN
Change := FALSE;
Step := 8;
END_IF
```

```
4: (*Teine käik*)
C2.0 := TRUE;
C2.1 := FALSE;
IF Clutch = FALSE THEN
Step := 5;
END_IF

5:
G1 := TRUE;
G2 := FALSE;
IF 1 = TRUE AND G2 = FALSE THEN
Change := FALSE;
Step := 8;
END_IF

6:(*Kolmas käik*)
C2.0 := TRUE;
C2.1 := FALSE;
IF Clutch = FALSE THEN
Step := 7;
END_IF

7:
G1 := FALSE;
G2 := TRUE;
IF G1 = FALSE AND G2 = TRUE THEN
Change := FALSE;
Step := 8;
END_IF

8:
Change := FALSE;
IF Change = FALSE  THEN
VarBool2 := TRUE;
END_IF
TONInst2(IN := VarBOOL2, PT:= T#5s);
TONInst2.Q;
TONInst2.ET;
IF TONInst2.Q = TRUE THEN
Step:= 0;
TONInst2(IN := FALSE);
VarBool2 := FALSE;
END_IF
END_CASE
```

(Põhiprogrammi lõpp)

LISA S Mootori ja masina kiiruse mõõtmine

(Muutujate defineerimise algus)

```
FUNCTION_BLOCK MotorMove
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
VAR
	Sagedus1:FREQ_MEASURE;
	Sagedus2:FREQ_MEASURE;
	Sagedus3:FREQ_MEASURE;
	Sagedus4:FREQ_MEASURE;
	Sagedus5:FREQ_MEASURE;
	Sagedus6:FREQ_MEASURE;
	avg1:STATISTICS_REAL;
	avg2:STATISTICS_REAL;
	avg3:STATISTICS_REAL;
	avg4:STATISTICS_REAL;
	avg5:STATISTICS_REAL;
	avg6:STATISTICS_REAL;
	kmh:R_TRIG;
	rpm:R_TRIG;
	cylpos1:R_TRIG;
	cylpos2:R_TRIG;
	cylpos3:R_TRIG;
	cylpos4:R_TRIG;
	Timer: CTU;
	First: BOOL:= TRUE;
	delay:TOF;
	rev: REAL;
	TONInst: TON;
	VarBOOL1: BOOL;
	counter: INT;
	hold:REAL;
	Step:INT := 1;
	VarBool2: BOOL;
	Timer2:TOF;
	Timer3:TOF;
END_VAR
```

(Muutujate defineerimise lõpp)

(Põhiprogrammi algus)

```
kmh(CLK:=%IX0.1);
rpm(CLK:=ENC_I1.7); (*ENC_I1.7*)
cylpos1(CLK:=Lin_Pos_sens2[0].10);
cylpos2(CLK:=Lin_Pos_sens2[1].10);
cylpos3(CLK:=Lin_Pos_sens2[2].10);
cylpos4(CLK:=Lin_Pos_sens2[3].10);
```

```
Sagedus1(IN:= kmh.Q); (*KMH*)
Sagedus2(IN:= rpm.Q); (*RPM*)
Sagedus3(IN:= cylpos1.Q, PERIODS := 1); (*CylPos1*)
Sagedus4(IN:= cylpos2.Q, PERIODS := 1); (*CylPos2*)
Sagedus5(IN:= cylpos3.Q, PERIODS := 1); (*CylPos3*)
Sagedus6(IN:= cylpos4.Q, PERIODS := 1); (*CylPos4*)
avg1(IN := Sagedus1.OUT );(* km/hr*)
avg2(IN := Sagedus2.OUT * 60.0 / 16.0); (* rpm*)
avg3(IN := Sagedus3.OUT * 10.24); (* mm/s*)
avg4(IN := Sagedus4.OUT * 10.24);(* mm/s*)
avg5(IN := Sagedus5.OUT * 10.24);(* mm/s*)
avg6(IN := Sagedus6.OUT * 10.24);(* mm/s*)
Kiirused[0] := Sagedus1.OUT / 2 * 1.6 * 3.6;
Kiirused[1] := avg2.AVG;
Kiirused[2] := avg3.AVG;
Kiirused[3] := avg4.AVG;
Kiirused[4] := avg5.AVG;
Kiirused[5] := avg6.AVG;
IF avg1.COUNTER > 100 AND avg2.COUNTER > 100 AND avg3.COUNTER > 100 AND
avg4.COUNTER > 100 AND avg5.COUNTER > 100 AND avg6.COUNTER > 100 THEN
avg2(RESET:=TRUE);
avg3(RESET:=TRUE);
avg4(RESET:=TRUE);
avg5(RESET:=TRUE);
avg6(RESET:=TRUE);
ELSE
avg2(RESET:=FALSE);
avg3(RESET:=FALSE);
avg4(RESET:=FALSE);
avg5(RESET:=FALSE);
avg6(RESET:=FALSE);
END_IF
IF kmh.Q = FALSE THEN VarBool1 := TRUE;
ELSE
VarBool1 := FALSE;
END_IF
TONInst(IN := VarBOOL1, PT:= T#2s);
TONInst.Q;
TONInst.ET;
IF TONInst.Q = TRUE THEN
Sagedus1(RESET :=TRUE);
ELSE
Sagedus1(RESET:= FALSE);
END_IF
Timer2(IN:=cylpos1.Q, PT := t#1s);
IF Timer2.ET = t#1s THEN
Sagedus3(RESET :=TRUE);
Sagedus4(RESET :=TRUE);
Sagedus5(RESET :=TRUE);
Sagedus6(RESET :=TRUE);
```

```
ELSE
Sagedus3(RESET:= FALSE);
Sagedus4(RESET:= FALSE);
Sagedus5(RESET:= FALSE);
Sagedus6(RESET:= FALSE);
END_IF
Timer3(IN:=rpm.Q, PT := t#500ms);
IF Timer2.ET = t#500ms THEN
Sagedus2(RESET :=TRUE);
ELSE
Sagedus2(RESET:= FALSE);
END_IF
(*Force calculations*)
V2 := (Kiirused[2]) * (((50.0*50.0) * 3.14) / 4) ; (*mm^3/s*)
```

(Põhiprogrammi lõpp)

LISA T Analoog sisend

(Muutujate defineerimise algus)

```
FUNCTION_BLOCK AnalogRead
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
VAR
        First    :BOOL:=FALSE;
        Step     :INT;
        Step2    :INT :=0;
END_VAR
```

(Muutujate defineerimise lõpp)

(Põhiprogrammi algus)

```
IF First=FALSE THEN
        Step:=0;
END_IF

CASE Step OF
0:  A_OUT0:=1000;   (*config 1. channel*)
          A_OUT1:=10;   (*4-20mA , 16 average, IL format *)
                IF A_IN0=A_OUT0 AND
                        A_IN1=A_OUT1
                THEN
                        Step:=Step+1;
                        First:=TRUE;
                END_IF
1:  A_OUT0:=1100;   (*config 2. channel*)
          A_OUT1:=20;   (*0-10V , 16 average, IL format *)
                IF A_IN0=A_OUT0 AND
                        A_IN1=A_OUT1
                THEN
                        Step:=Step+1;
                END_IF
2:  A_OUT0:=1200;   (*config 3. channel*)
          A_OUT1:=30;   (*0-10V , 16 average, IL format *)
                IF A_IN0=A_OUT0 AND
                        A_IN1=A_OUT1
                THEN
                        Step:=Step+1;
                END_IF
3:  A_OUT0:=1300;   (*config 4. channel*)
          A_OUT1:=40;   (*0-10V , 16 average, IL format *)
                IF A_IN0=A_OUT0 AND
                        A_IN1=A_OUT1
                THEN
```

```
                        Step:=Step+1;
                END_IF
4:  A_OUT0:=1400;   (*config 5. channel*)
        A_OUT1:=50;   (*4-20mA , 4 average, IL format *)
                IF A_IN0=A_OUT0 AND
                        A_IN1=A_OUT1
        THEN
                        Step:=Step+1;
                END_IF
5:  A_OUT0:=1500;   (*config 6. channel*)
        A_OUT1:=60;   (*4-20mA , 4 average, IL format *)
                IF A_IN0=A_OUT0 AND
                        A_IN1=A_OUT1
        THEN
                        Step:=Step+1;
                END_IF
6:  A_OUT0:=1600;   (*config 7. channel*)
        A_OUT1:=70;   (*4-20mA , 4 average, IL format *)
                IF A_IN0=A_OUT0 AND
                        A_IN1=A_OUT1
                THEN
                        Step:=Step+1;
                END_IF
7:  A_OUT0:=1700;   (*config 8. channel*)
        A_OUT1:=80;   (*4-20mA , 4 average, IL format *)
                IF A_IN0=A_OUT0 AND
                        A_IN1=A_OUT1
        THEN
                        Step:=Step+1;
                END_IF

8:
Step:=Step+1;

9:  A_OUT0:=0000;   (*read 1. channel*)
        A_OUT1:=0000;
                IF A_IN0=A_OUT0
        THEN
                Preassure[0]:= WORD_TO_REAL(AnalogVAL)*Scale1;
                        Step:=Step+1;
                END_IF
10:  A_OUT0:=0100;   (*read 2. channel*)
        A_OUT1:=0100;
                IF A_IN0=A_OUT0
        THEN
                Lin_Pos_sens [0]:=AnalogVAL*0.005;
                Lin_Pos_sens2[0]:=A_IN1;
                        Step:=Step+1;
                END_IF
11:  A_OUT0:=0200;   (*read 3. channel*)
```

```
            A_OUT1:=0200;
                IF A_IN0=A_OUT0
            THEN
                  Lin_Pos_sens [1]:=AnalogVAL*0.005;
                  Lin_Pos_sens2[1]:=A_IN1;
                      Step:=Step+1;
                END_IF
12:  A_OUT0:=0300;   (*read 4. channel*)
            A_OUT1:=0300;
                IF A_IN0=A_OUT0
            THEN
                  Lin_Pos_sens [2]:=AnalogVAL*0.005;
                  Lin_Pos_sens2[2]:=A_IN1;
                      Step:=Step+1;
                END_IF
13:  A_OUT0:=0400;   (*read 5. channel*)
            A_OUT1:=0400;
                IF A_IN0=A_OUT0
            THEN
                      Lin_Pos_sens [3]:=AnalogVAL*0.005;
                      Lin_Pos_sens2[3]:=A_IN1;
                      Step:=Step+1;
                END_IF
14:  A_OUT0:=0500;   (*read 6. channel*)
            A_OUT1:=0500;
                IF A_IN0=A_OUT0
            THEN
                  PotVal:=AnalogVAL;
                      IF PotVal < 70 THEN PotVal := 70;
                      ELSIF PotVal >14000 THEN PotVal := 14000; END_IF

                      Step:=Step+1;
                END_IF
15:  A_OUT0:=0600;   (*read 7. channel*)
            A_OUT1:=0600;
                IF A_IN0=A_OUT0
            THEN
                  Temp[0]:=1.2209 * EXP(0.0001 * AnalogVAL);
                      Step:=Step+1;
                END_IF
16:  A_OUT0:=0700;   (*read 8. channel*)
            A_OUT1:=0700;
                IF A_IN0=A_OUT0
            THEN
                      Temp[1]:=1E-06*(AnalogVAL*AnalogVAL) - (0.0679 * AnalogVAL)
+ 839.48;
                      Step:=9;
                END_IF
END_CASE
```

```
CASE Step2 OF
0:
A2_OUT0:=0800;   (*config 2. channel*)
A2_OUT1:=0800;   (*0-10V , 16 average, IL format *)
Step2:=Step2+1;

1:
A2_OUT0:=0800;   (*config 2. channel*)
A2_OUT1:=0800;   (*0-10V , 16 average, IL format *)
Step2:=Step2+1;
2:

Preassure[1] := A2_IN0 * 0.00033;
Preassure[2] := A2_IN1 * 0.00033;

END_CASE

IF Preassure[0] < 0.5 OR Preassure[1] < 0.5 OR Preassure[2] < 0.5 THEN

Preassure[0]:=0;
Preassure[1]:=0;
Preassure[2]:=0;
END_IF
```

(Põhiprogrammi lõpp)

LISA U Enkooder

(Muutujate defineerimise algus)

```
FUNCTION_BLOCK EncoderRead
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
VAR
        First    :BOOL:=FALSE;
        Step     :INT:=0;
        ENCODER_POS: INT;
        constrain:LIN_TRAFO;
END_VAR
```

(Muutujate defineerimise lõpp)

(Põhiprogrammi algus)

```
(*O0 bit 15-9 control code*)
IF First=FALSE THEN
        Step:=0;
END_IF

CASE Step OF
 (*config 1. encoder*)
0: ENC_O0:=400;
  IF ENC_I0 = 0
                THEN
                Step:=Step+1;
                First:=TRUE;
                END_IF

1:  ENC_O0:=200;
          ENC_O1:=16#0551;
                IF ENC_O0= ENC_I0
                THEN
                        Step:=Step+1;

                END_IF

2: IF ENC_I0 = 20 THEN
ENC_O1:=16#5555;
ENC_O0:=16#200   (*read encoder*)
ELSE
Step:=0;
END_IF
(*ENC_O1:=5552;*)
END_CASE
```

```
constrain(IN:= WORD_TO_REAL(ENC_I1), IN_MIN := 0, IN_MAX := 4096, OUT_MIN := 0, OUT_MAX := 359.9);
ENCODER := constrain.OUT;
```

(Põhiprogrammi lõpp)

LISA V Analoog väljund

(Muutujate defineerimise algus)

```
FUNCTION_BLOCK AnalogOutput
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
VAR
        PID:PID;
        set:REAL;
        K_P: REAL:=10;
        T_N: REAL:=0.01;
        T_V: REAL:=0;
        constrain:LIN_TRAFO;
        R1:R_TRIG;
        L: REAL:=0;
END_VAR
```

(Muutujate defineerimise lõpp)

(Põhiprogrammi algus)

```
constrain(IN := PotVal, IN_MIN:= 70, IN_MAX := 14000, OUT_MIN:= 0, OUT_MAX :=
65535);
R1(CLK := %IX0.2);
IF R1.Q AND L = 0 AND ManualAuto = FALSE THEN
L := 1;
Set := Kiirused[1];
ELSIF ((R1.Q AND L = 1) OR %IX0.5 = TRUE) AND ManualAuto = FALSE THEN
L := 0;
ELSIF ManualAuto = TRUE AND PID_S = TRUE THEN
L := 2;
ELSIF ManualAuto = TRUE AND PID_S = FALSE THEN
L := 3;
ELSIF (L = 2 OR L = 3) AND ManualAuto = FALSE THEN
L := 0;
END_IF
IF PID_F = TRUE THEN
SpeedSet_F := SpeedSet2;
ELSIF PID_F = FALSE THEN
SpeedSet_F := 0;
END_IF
IF PLC_PRG.GearChange.Change = TRUE THEN
AO_OUT0 :=8000;
ELSIF L = 1 AND PLC_PRG.GearChange.Change = FALSE AND ManualAuto = FALSE
THEN
PID;
```

```
PID( ACTUAL := Kiirused[1], SET_POINT := set, KP := K_P, TN := T_N, TV := T_V, Y_MIN := 0, Y_MAX := 65535, RESET:=FALSE);
AO_OUT0 :=REAL_TO_WORD(PID.Y);
ELSIF L = 0 AND PLC_PRG.GearChange.Change = FALSE AND ManualAuto = FALSE THEN
AO_OUT0 :=REAL_TO_WORD(constrain.OUT);
ELSIF ManualAuto = TRUE AND PID_F = FALSE AND PLC_PRG.GearChange.Change = FALSE AND L = 3 AND RunStop_F = TRUE THEN
AO_OUT0 :=P_S_EP_Write * REAL_TO_WORD(6553.5);
ELSIF ManualAuto = TRUE AND PID_F = TRUE AND PLC_PRG.GearChange.Change = FALSE AND L = 2 AND RunStop_F = TRUE THEN
PID;
PID( ACTUAL := Kiirused[1], SET_POINT := SpeedSet_F , KP := K_P, TN := T_N, TV := T_V, Y_MIN := 0, Y_MAX := 65535, RESET:=FALSE);
AO_OUT0 :=REAL_TO_WORD(PID.Y);
ELSE
AO_OUT0:=0;
END_IF
```

(Põhiprogrammi lõpp)

LISA S Lihtlitsents lõputöö salvestamiseks ja üldsusele kättesaadavaks tegemiseks ning juhendaja(te) kinnitus lõputöö kaitsmisele lubamise kohta

Mina, _____ ,
(*autori nimi*)
sünniaeg _____,

1. annan Eesti Maaülikoolile tasuta loa (lihtlisentsi) enda loodud lõputöö

_____

_____

_____,
(*lõputöö pealkiri*)

mille juhendaja(d) on _____ ,

1.1 salvestamiseks säilitamise eesmärgil,
1.2.digiarhiivi DSpace lisamiseks ja
1.3. veebikeskonnas üldsusele kättesaadavaks tegemiseks

kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile;

3. kinnitan, et lihtlisentis andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Lõputöö autor _____
(*allkiri*)

Tartu, _____
(*kuupäev*)

Juhendaja(te) kinnitus lõputöö kaitsmisele lubamise kohta

Luban lõputöö kaitsmisele.

_____ _____
(*juhendaja nimi ja allkiri*) (*kuupäev*)