TARTU ÜLIKOOL
Arvutiteaduse instituut

Infotehnoloogia mitteinformaatikutele õppekava

Indrek Pomerants

Automaatjuhtimissüsteemi testimine kontrolleri UC20-WL2000-AC näitel

Magistritöö (15 EAP)

Juhendajad: Ivar Koppel, PhD
Kaido Jaanus, MSc

Tartu 2021

Automaatjuhtimissüsteemi testimine kontrolleri UC20-WL2000-AC näitel

Lühikokkuvõte:

Magistritöö eesmärk on luua toimiv lahendus automaatjuhtimissüsteemi testimiseks, mida ettevõte saab kasutada süsteemide arendamiseks, uute töötajate väljaõpetamiseks ja turundusüritustel osalemiseks.

Töös on kirjeldatud automaatjuhtimissüsteemi automaatikakontrolleri valiku tingimusi ja programmeerimist. Hinnatakse valminud testautomaatjuhtimissüsteemi vastavust esitatud nõuetele ja kontrolleri sobivust tööstusautomaatikalahendusi pakkuva ettevõtte vajadustega. Töös on keskendutud eelkõige kontrolleri erinevate funktsionaalsuste rakendamisele. Vähesel määral on käsitletud ka teisi olulisi komponente, mis on süsteemi toimimiseks hädavajalikud. Töös on antud soovitusi edasiseks kasutus- ja arendustegevusteks.

Analüüsi käigus hinnati kontroller mitte sobivaks ettevõtte levinud projektlahendustesse. Seade võib leida potentsiaalset kasutust olukorras, kus on tarvis teostada e-kirjaga teavitusi, lihtsas tehnoloogilises lahenduses või kui on vajadus kontrolleri mõne kindla funktsionaalsuse rakendamiseks.

Valminud lõputööd saab kasutada lähteülesande ja juhendmaterjalina ka mõne muu kontrolleri või riistvara testimiseks.

Võtmesõnad:
PLC, Programmeeritav loogikakontroller, PAC, automaatikakontroller, FBD, funktsiooniplokkskeem programmeerimine, Node-RED, automaatjuhtimissüsteem.

CERCS:

T125 Automatiseerimine, robootika, juhtimistehnika

Automation Control System Testing On The Example Of UC20-WL2000-AC

Abstract:

The master's thesis aims to create a functional solution for testing an automation control system that the company can use for development, training, and marketing purposes.

Master’s thesis describes the terms for choosing a programmable automation controller and its programming. A composed test automation control system is evaluated if it is according to the collected requirements and if it corresponds to the company's needs that offer automation solutions for the industry. The thesis focuses primarily on different applications and their functionalities that the controller provides. Other components that are necessary for the system to function have been described. Suggestions for the test system use case and further development have been given.

The analysis evaluates the controller to be not suitable for the company standard project solutions. The device could find potential use in applications where email notifications are

required, in simple technological solutions, or there is a use case for specific functionality that the controller provides.

The thesis provides the company a reference and guidance material to test different controllers or other hardware.

Keywords:
PLC, programmable logic controller, PAC, programmable automation controller, FBD, Function Block diagram, Node-RED, automation control system

CERCS:

T125 Automation, robotics, control engineering

Sisukord

Kasutatud mõisted ja terminid

4G	Mobiilse andmesidevõrgu neljas generatsioon, mis sätestab nõuded mobiilside kiirusele liikuvatele seadmetes kuni 100Mbit/s ja statsionaarsetele seadmetele kuni 1Gbit/s [1].
AHP	(*Analytic Hierarchy Process*) nõuete prioriteetide määramiseks mõeldud analüütiline meetod, kus nõudeid võrreldakse omavahel ja hinnatakse vastavalt ettenähtud kriteeriumitele.
CODESYS	Tootjaneutraalne automaatikakontrollerite programmeerimisplatvorm, mis toetab programmeerimiskeeli vastavalt standardile IEC 61131-3 [2].
CSS	(*Cascading Style Sheets*) märgistuskeel veebilehtede kujundamiseks [3].
Debug	Silumine ehk programmi vigade avastamine ja kõrvaldamine [3].
DNS	(*Domain Name Server*) domeeninimeserver. TCP/IP võrgu komponentide, teenuste ja ressursside nimede süsteem [4]. Teisendab nimesid IP-aadressideks ja vastupidi [4].
EEPROM	Püsimälu, mis on elektriliselt ümberprogrammeeritav [3].
Ethernet	Kohtvõrgu standard [3].
FBD	(*Function Block Diagram)* funktsiooniplokkskeem. Kontrolleri programmeerimiseks mõeldud graafiline programmeerimiskeel [5]. Programm koosneb põhiloogikaelementidest (tarkvara tootja loodud) ja kasutajal loodud elementidest [5].
Firmware	Püsivara – seadme püsimällu salvestatud, käskude ja andmete kogum [3].
Gateway	Lüüs – erinevaid arhitektuure ja protokolle kasutatavaid võrke ühendav funktsionaalüksus, mis teisendab protokolle vastavalt võrkudevahelisele andmete liikumisele [3].
Help	Spikker [3].
HTML5	(*Hyper Text Markup Language*) veebis kasutatava markeerimiskeele viies versioon.
HTTP	(*HyperText Transfer Protocol)* veebis andmete edastuseks ja kuvamiseks kasutatud protokoll. Andmeid edastatakse teksti kujul ja krüpteerimata [3].
HTTPS	(*Secure HyperText Transfer Protocol)* veebis andmete edastuseks ja kuvamiseks kasutatud protokoll [3]. Andmeid edastatakse krüpteeritud kujul [3].
IL	(*Instruction List)* käsulist, tekstikujuline madalataseme keel kontrolleri programmeerimiseks [6]. Vastab standardile IEC 61131-3.
Industry 4.0	Neljas tööstusrevolutsioon.
IoT	*(Internet of Things)* asjade internet.
IP-aadress	IP võrku ühendatud seadme identifikaator [3].

IPv4	(*Internet Protocol versioon 4*) neljas versioon IP protokollist [3].
JavaScript	Tekstiline programmeerimiskeel, mis on kasutatud põhiliselt veebilahenduste programmeerimiseks [7].
LAN	Kohtvõrk [3].
LD	(*Ladder Diagram)* redelskeem või kontaktaseskeem. Kontrolleri programmeerimiseks mõeldud graafiline programmeerimiskeel, mis sarnaneb elektriskeemiga [5]. Vastab standardile IEC 61131-3.
LTE	(*Long-term Evolution*) andmesidestandard, mis võimaldab andmeedastuskiirust kuni 100Mbit/s (esialgne versioon), kuid ei vasta päris 4G standardis esitatud nõuetele [1]. Edasiarendused LTE tehnoloogias (LTE-A) võimaldavad suuremaid andmesidekiiruseid [1].
M2M	(*Machine to machine*) masinate vaheline suhtlus.
Modbus RTU	Modbus jadasideprotokoll, mis on levinud tööstuses seadmete vaheliseks andmevahetuseks.
MITM	(*Man in the middle attack*) ründe läbiviija saab salaja lugeda pooltevahelisi sõnumeid, neid lisada ja muuta [4].
Modbus TCP/IP (Modbus TCP)	Tööstuses levinud andmesideprotokoll, mis omakorda kasutab TCP/IP protokolli.
MoSCow	Nõuete prioriteetide määramiseks mõeldud tehnika. Termin koosneb akronüümidest: M – *must have* (peab olema); S – *should have* (peaks olema); C – *could have* (võiks olla); W – *won't have* (ei ole). Nõudeid hinnatakse ja kategoriseeritakse vastavalt.
Muutmälu	vt RAM.
Node	Sõlm, Node-RED arenduskeskkonna mõistes on tegemist mingit funktsiooni omava elemendiga. Sõlmel saab olla maksimaalselt üks sisend ja võib olla null või rohkem väljundit [8].
Node-RED	Node-RED on voopõhine sündmustest juhitud programmeerimistööriist, mis on arendatud Node.js peal [9]. Node.js on JavaScript käitusajaobjekt (*runtime*), mis on arendatud Chrome'i V8 JavaScript mootorile [9].
NTP	(*Network Time Protocol*) võrguaja protokoll.
OPC UA	(*Open Platform Communication Unified Arhitecture*) on avatud standard, mis kirjeldab infovahetust tööstuslikus andmesides, mis võimaldab seadmelt seadmele (*machine to machine* ehk M2M) suhtlust näiteks SCADA platvormi ja kontrolleri vahel [10].
Overhead	Üldkulu, mille alla loetakse operatsioonisüsteemi ja toetavate utiilide tööks kuluvat aega [3].
PAC	(*Programmable Automation Controller)* programmeeritav automaatikakontroller.
Payload	Andmepaketi osa, mis sisaldab sõnumi sisu [3].
PING	(*Packet InterNet Groper*) kasutatakse TCP/IP võrgus sihtkohtade kättesaadavuse kontrolliks [3].

PLC	(*Programmable Logic Controller*) programmeeritav loogikakontroller.
Port	TCP/IP ja UDP võrgus loogilise kanali lõpp-punkt [3][4].
PT100	Takistustermomeeter, mille elektriline takistus kasvab temperatuuri tõustes. Temperatuuril 0°C on takistuse väärtuseks 100Ω [11].
Püsimälu	vt EEPROM.
RAM	Muutmälu, mis kuulub hävimälude hulka [3]. Andmed kustuvad toitepinge kadumisel [3].
Real-time	Süsteem mis reageerib piisavalt kiiresti sisendsignaalile ja on võimeline juhtima protsessi nõutud kiirusega [3]. „Sisenevad sündmused tuleb töödelda ettemääratud ajavahemikus“ [4]. Töötlus sooritatakse välise protsessi toimumise ajas selliselt, et saadud tulemust oleks võimalik kasutada protsessi juhtimiseks, seireks või õigeaegseks reageerimiseks [4].
Rebuut	(*Reboot*) seadme operatsioonisüsteemi taaskäivitamine [3].
RJ45	(*Registered Jack*) standardiseeritud pistikühendus, mida kasutatakse Ethernet võrkudes [12].
Runtime	Käitusaegne ehk programmi täitmise aegne [4].
Sampling rate	Siskreetimissagedus. Analoog-digitaalmuunduri töösagedus [3]. Siinjuhul iseloomustab kui kiiresti töödeldakse moodulisse sisenevat analoogsignaali. Mida kiiremini, seda täpsemalt jälgib sisendsignaali kuju.
SCADA	(*Supervisory Control and Data Acquisition)* kaugjälgimissüsteem, mille eesmärk on kontrolleritest saadava informatsiooni visualiseerimine protsessi jälgimiseks ja juhtimiseks.
SMTP	(*Simple Mail Transfer Protocol*) lihtne meiliedastusprotokoll, mis on ette nähtud e-kirjade saatmiseks ja vastuvõtmiseks serverite vahel [3].
ST	(*Structured Text*) struktureeritud tekst. Kontrolleri programmeerimiseks mõeldud tekstiline programmeerimiskeel, mis omab sarnasusi kõrgtaseme keeltega nagu C++ ja Pascal [5].
TCP/IP	(*Transmission Control Protocol/Internet Protocol)* „edastusohje protokollistik internetiprotokolli peal, interneti protokollistik [3].“
TLS	(*Transport Layer Security*) võimaldab klient-server rakendustel turvaliselt üle interneti suhelda [3].
USB	(*Universal Serial Bus*) universaalne jadasiin. Standardiseerib väliste seadmete ja arvutite vahelist ühendamist [3].
VPN	(*Virtual Private Network*) avalikku telekommunikatsiooni infrastruktuuri kasutav privaatvõrk [3]. Kasutatakse parema turvalisuse ja privaatsuse säilitamise eesmärgil [3].

Sissejuhatus

Tööstusautomaatika valdkonna eesmärgiks on tööstussektori protsesside automatiseerimine, mis väljendub süsteemide efektiivsemas juhtimises, rikete, seisakute ja avariiolukordade ennetamises ning reageerimisaja vähendamises, kvaliteedi kontrollis ja protsessi jälgimises. Sektori kasvule on mõnekümne aasta jooksul kaasa aidanud kiire kommunikatsioonitehnoloogia areng ja ettevõtjate soov automatiseerituse taset tootmises tõsta.

Tööstusautomaatika lahendusi pakkuv süsteemiintegraator peab olema kursis kiiresti arenevate automaatikakontrollerite, täiturmehhanismide ja andurite tehnoloogiatega. Ettevõtte vajab lahendust uute tehnoloogiate testimise ja kasutuselevõtu kiirendamiseks, mis aitaks lihtsustada uute ideede katsetamist kontrollitud oludes ja töötajate integreerimist ettevõtte tootmis- ja arendusprotsessi.

Magistritöö eesmärk on luua toimiv lahendus automaatjuhtimissüsteemi testimiseks, mida ettevõte saab kasutada süsteemide arendamiseks, uute töötajate väljaõpetamiseks ja turundusüritustel osalemiseks. Lahenduse väljatöötamiseks kasutatakse kontrollerit, mida ettevõte ei ole varem oma projektides rakendanud. Valitud kontroller aitab paremini välja tuua uue tehnoloogia kasutuselevõtuga seotud probleeme ja võimaldab töö lõpus anda hinnangut seadme sobivusele ettevõtte vajadustega. Koostatud tööülesannet saab kasutada uute kontrollerite testimiseks ja võimaldab võrrelda valminud lahenduste vastavust nõuetele.

Magistritöös kirjeldatakse automaatjuhtimissüsteemi kontrolleri valiku tingimusi ja programmeerimist ning hinnatakse valminud testautomaatjuhtimissüsteemis kasutatud kontrolleri vastavust esitatud nõuetele ja ettevõtte vajadustele. Töö keskendub eelkõige kontrolleri erinevate funktsionaalsuste rakendamisele, mille käigus kasutatakse kahte erinevat programmeerimiskeskkonda ja keelt ning vähesel määral on käsitletud ka teisi olulisi komponente, mis on süsteemi toimimiseks hädavajalikud.

Autoril puudub eelnev kogemus töös kasutatud automaatikakontrolleri ja selle pakutud tarkvaralahenduste ja programmeerimiskeeltega. Töös on kirjeldatud:

1. Ettevõtte vajaduste kaardistamine - kuidas on seni kontrollerit testitud ja missuguseid lahendusi pakutakse. Seotud osapoolte nõuded uuele loodavale süsteemile koos tööülesande püstitusega. Tööülesanne kirjeldab ära tehnoloogilise protsessi, mida kontrolleriga juhitakse.
2. Riistvara valik – kirjeldatud on lahenduse jaoks olulisi seadmeid ja nende valikupõhimõtteid.
3. Võrgu arhitektuur – üldise struktuuri, selle erinevate kihtide, turvalisuse ja töös kasutatava lahenduse kirjeldus.
4. Kontroller – seadme ja arenduskeskkonna tutvustus koos ligipääsu ja püsivara uuenduse kirjeldusega
5. Kontrolleri funktsiooniplokkskeem rakendus – kirjeldatud on kasutatud juhendmaterjali ja põhimõtteid ning tehnoloogilise protsessi juhtimiseks loodud programmi.
6. Programmeerimine Node-RED keskkonnas – tutvustatud on keskkonda ja selle abil loodud programmi, mille eesmärk on alarmide edastus e-kirja teel.

7. Kasutajaliidese loomine – tegemist on kontrolleri sisese visualiseerimiskeskkonnaga. Kirjeldatud on selle erinevaid elemente ja võimalusi.
8. Hinnang lahendusele – töö viimases osas on analüüsitud ja hinnatud kasutatud keskkondasid, valminud lahenduse vastavust esitatud nõuetega ning kontrolleri sobivust ettevõtte vajadustega.

Valminud programm, muutujate loetelu, kasutuslood ja erinevate töölehtede vaated on esitatud lisades.

1. Ettevõtte vajaduste kaardistamine

Ettevõte Entronik OÜ tegeleb tööstusautomaatika lahenduste pakkumisega alates projekteerimisest lõpetades automaatsüsteemi visualiseerimisega. Põhilisteks tööstusvaldkondadeks on joogi- ja reoveetöötlus, katlamajad, viljakuivatid ja erinevat tootmisliinid ja prototüüplahendused. Projektist sõltuvalt on kasutusel erinevate tootjate andurid, kontrollerid ja muu riistvara, mida on tihti vaja eelseadistada ja testida enne kasutuselevõttu.

Järgnevates peatükkides on lähemalt kirjeldatud hetkeolukorda, süsteemile esitatud nõuete kogumist, kaardistamist ja tööülesande koostamist.

1.1. Hetkeolukorra kirjeldus

Seni on ettevõttes automaatjuhtimissüsteemi testiks ja arenduseks kasutatud kontrolleri arendustarkvarasse sisse ehitatud simulatsioonikeskkonda, mis ei vaja kontrolleri olemasolu. Kui on olnud vajadus kontroller ühendada arvuti ja arenduskeskkonnaga, siis tehakse seda kontori laual. Funktsionaalselt selline lahendus toimib, kuid võib ohtu kujutada teistele inimestele ja seadmetele. Programmeerija ise või keegi teine võib tahtmatult süsteemi terviklikkus rikkuda, näiteks lükatakse seade laua pealt maha või ühendub mõni juhe lahti, mille tagajärjel võib kontroller saada kahjustada või halvemal juhul ohtu sattuda inimelu. Kontrolleri ja muu riistvara testimine kontrollitud oludes võimaldab:

1. Avastada varases staadiumis defektne toode;
2. Uuendada riistvara püsivara (*firmware*);
3. Teostada riistvara eelseadistust (näiteks IP aadressid);
4. Testida riistavaral tarkvaralist funktsionaalsust.

Ettevõtted, kes tegutsevad sarnases valdkonnas, loovad oma testsüsteemi ise vastavalt oma vajadustele, näiteks Clever Automatics OÜ. Asutused, kes tegelevad põhiliselt ühe tootja seadmetega kasutavad ka vastava tootja stardikomplekte näiteks ABB Power Grids Estonia AS või loovad oma tootja komponentidest tooteid tutvustava stendi näiteks Schneider Electric Eesti AS.

Erinevad kontrolleri tootjad pakuvad valmis stardikomplekte oma seadmete tutvustamiseks, näiteks Phoenix Contact [13] või Siemens [14], kuid tegemist on kitsa tootepõhise lahendusega.

Esimesel juhul on tegemist kompaktse stendiga, kus seadmete vahetamine mingi muu tootega on keeruline või ei ole teostatav. Süsteemis on võimalus testida digi- ja analoogsignaale.

Teisel juhul on tegemist kontrolleri ja tarkvara komplektiga, mis ei sisalda seadmete kinnitus ega toitelahendust. Süsteem võimaldab testida digisignaale, kuid ei sisalda vahendeid analoogsignaali genereerimiseks, kuigi seadmel on kaks analoogsignaali sisendit.

Pakutavad süsteemid tulevad tihti koos näidisprojektiga, kuid ei sisalda ise õppimiseks näidisülesandeid. Seadmel on olemas kõik elementaarne, mis on vajalik alustamiseks, kuid

põhjalikumaks tutvumiseks ei pruugi sellest piisata, mis teeb stardikomplekti süvitsi testimise keerukaks ilma seda täiustamata. Stardikomplekti täiustamisel tuleb leida ise lahendus komponentide paigutusele, elektritoitele ja valminud funktsionaalsem lahendus ei pruugi olla ohutu. Ettevõttel on vajadus universaalsema ja funktsionaalsema lahenduse järele, kui see, mida mainitud tootjad hetkel pakuvad.

Töötajatel on praktiline vajadus lihtsalt ja ohutult teostada automaatjuhtimissüsteemi komponentide esmast testimist ja seadistamist. Vaja on aja- ja kuluefektiivsemat lahendust automaatjuhtimissüsteemi arenduseks ja uute töötajate integreerimiseks ettevõtte protsessidesse. Esitatud kirjelduse järgi võib defineerida lahendust kui stendi, mis tagab seadmete ohutu kasutamise. Katsestendi funktsionaalsust on võimalik muuta läbi uute seadmete lisamise ja kontrolleri programmeerimise. Töö käigus luuakse ettevõtte vajadusi arvestav automaatjuhtimissüsteemi testimislahendus.

Järgnevalt on lähemalt kirjeldatud loodavale süsteemile esitatud nõudeid, nende kogumist ja grupeerimist.

1.2. Nõuded süsteemile

Esmalt on kindlaks tehtud seotud osapoolte nõuded loodavale lahendusele. Süsteem koosneb nii riistvarast kui tarkvaralisest funktsionaalsusest, mille nõudeid esitavad sellega seotud osapooled. Süsteemiga seotud osapooled on esitatud tabelis 1.1.

Tabel 1.1. Süsteemi seotud osapooled

Seotud osapool	Kirjeldus
Ettevõtte tegevjuht	Ettevõtte juhina määrab ettevõtte arengusuuna ja tasub testimise ja arendusega seotud kulud. On huvitatud kasutama stendi kui ettevõtte turunduse abivahendit messidel ja vahendit uute töötajate koolitamiseks.
Automaatikainsener	Tema ülesanne on kontrollerite ja andurite programmeerimine, seadistamine ja testimine ja sellega seotud insenertehniliste probleemide lahendamine. On huvitatud testimis ja arendustegevuse efektiivsemaks muutmisest.
SCADA insener	Tema ülesanne on üles seada tarkvara ja luua arvutisse kasutajaliides operaatori jaoks, mille abil süsteemi juhtida. Magistritöös on tema osaks kasutajaliidesele esitatud nõuded.

SCADA süsteem on oluline osa terviklahendusest, kuid selle ehitus ja ülesseadmine ei ole töö mahus. Nõuete kaardistamisel on arvesse võetud kõikide seotud osapoolte esitatud süsteemi nõudeid. Tegevjuhi poolt esitatud nõuded kaardistati kolme erineva koosoleku ja arutelu tulemusel. Automaatikainseneri nõuded tuvastati telefonivestluse käigus ja SCADA inseneri rolli täidab töö autor. Nõuete kaardistamiseks on koostatud kasutuslood, mis on esitatud lisas I.

Tegevjuhi nõudeid analüüsides, saab järeldada, et tegemist on mittefunktsionaalsete nõuetega. Automaatika- ja SCADA inseneri poolt on esitatud enamasti funktsionaalsed nõuded.

Järgnevalt on vaja nõuded grupeerida tähtsuse järgi. Meetodi valikul on lähtutud eelnevas õppetöös praktiseeritust:

1. AHP – on kasutatav väikse nõuete arvu korral (alla 10). Töös on kasutuslugude kaudu esitatud nõuete arv suurem ja AHP protsess oleks liialt ajamahukas, et kõiki nõudeid omavahel võrrelda ja hinnata.
2. Nõuete järjestamine tähtsuse järjekorda – Töös ei ole oluline nõuete täpne järjestamine, mis võtaks liialt palju aega. Piisab nõuete grupeerimisest.
3. MoScoW – Meetod sobib nõuete grupeerimiseks. Nõudeid ei pea omavahel võrdlema ja järjestama, mistõttu on see meetod siinjuhul kõige sobivam.

Analüüsi käigus leiti, et erinevate osapoolte seatud nõuded olid osaliselt kattuvad. Grupeeritud nõuded vastavalt prioriteedile on esitatud kokkuvõtvalt tabelis 1.2. Punasega on esitatud mittefunktsionaalsed nõuded.

Tabel 1.2. Nõuete grupeerimine kasutades MoScoW meetodit

Peab olema (*Must have*)	**Peaks olema *(Should have)***
Ohutu inimestele ja seadmetele. Abistama uue töötaja integreerimisel ettevõttesse Kontroller peab omama Ethernet TCP/IP võrguliidest. Kontrolleri programmeerimiskeel peab vastama IEC 61131-3 standardile. Analoog signaali mõõtmise võimekus (4…20mA ja 0…10V) Digitaalse signaali mõõtmise võimekus	Seadmete vahetamine stendis peab olema lihtne Stend annab ülevaate ettevõttes kasutatavatest tehnoloogiatest. Alarmide edastamine toimuma e-kirja teel. Alarmide kuvamine kasutajaliideses avalehel. E-kirja saaja muutmine läbi kasutajaliidese Modbus TCP/IP andmeside kasutamine kontrolleri ja mõne teise seadme vahel
Võiks olla *(Could have)*	**Ei ole *(Won't have)***
Lihtsasti teisaldatav, kompaktne, alusele kinnituv ja seinale kinnituv stend Seadmed ja juhtmed peavad olema tähistatud Alarmi seadeväärtuste muutmine kasutajaliidesest Tarkvara peab võimaldama erinevate kasutajate ja rollide loomist. Alarmide e-kirjaga edastamise desaktiveerimine. Üksikute alarmide desaktiveerimine	Keskkond peab võimaldama raportite koostamist.

Tabelis 1.2 märgitud nõuded grupeeriti tähtsuse järgi vastavalt otstarbele ja võetakse arvesse tööülesande kirjeldamisel ja lahenduse loomisel. Lahenduse elluviimisel lähtuti eesmärgist rahuldada võimalikult palju nõudeid, alustades prioriteetsematest.

Kaardistatud ja grupeeritud nõudeid saab võtta arvesse tööülesande kirjeldamisel, mida on käsitletud järgnevas peatükis.

1.3. Tööülesanne

Eelnevas peatükis kirjeldatud nõuete kokkusidumiseks üheks toimivaks lahenduseks, on koostatud tööülesanne. Tööülesanne annab üldise ettekujutuse, millised on lahenduse põhifunktsionaalsused.

Valminud lahendus peab omama loogilist funktsiooni ja toimima ühe tervikuna, täites võimalikult suures osas esitatud nõudeid. Magistritöös on olulisel kohal testitav kontroller, ja seetõttu on ülesanne koostamisel jälgitud, et kasutust leiaks mitmed erinevad seadme funktsionaalsused. Kontrolleri põhjalikum rakendamine aitab paremini hinnata lõpplahenduses kasutatud kontrolleri sobivust ettevõtte vajadustega.

Kirjeldus – süsteemi temperatuuri mõõdab analoogtemperatuuriandur ja keskkonda jahutab ventilaator. Süsteemi on võimalik lülitada füüsilisest nupust käsijuhtimisele (ventilaator töötab koguaeg), automaatjuhtimisele (ventilaatori tööd juhib kontroller) ja välja lülitada. Ventilaatori töö korral põleb roheline indikaatorlamp.

Kui süsteem on automaatjuhtimisel, saab erinevaid parameetreid ja seeläbi ventilaatori tööd juhtida läbi kontrolleris oleva kasutajaliidese. Kui süsteemi temperatuur tõuseb üle või langeb alla alarmi seadeväärtuse, edastatakse alarmteade e-kirjaga. E-kirja edastust üldiselt ja üksikuid alarme peab saama desaktiveerida. E-posti edastamiseks peab kontrolleril olema ühendus internetiga.

Kasutajaliideses peab olema kuvatud hetke temperatuur, temperatuurigraafiku ajalugu, süsteemi olekud (automaatjuhtimisel, käsijuhtimisel, ventilaator töös) ja erinevad alarmid. Läbi kasutajaliidese peab saama muuta alarmide seadeparameetreid ja ventilaatori tööparameetreid.

Eesmärk – hoida süsteemi temperatuuri kindlas vahemikus.

Süsteemi riistvaralised elemendid:

1. Ventilaator;
2. Temperatuuriandur;
3. Mitmepositsiooniline nupplüliti;
4. Roheline indikaatorlamp;
5. Kontroller;
6. Võrguseadmed;
7. Muud komponendid (juhtmed, kaablid, karbikud jm).

Märkused – süsteemi temperatuuri muutuse simuleerimine peab olema võimalik käepäraste vahenditega, näiteks käega soojendamine.

Tööülesandes kirjeldatud kontrollerile mõeldud tegevused võib jagada kaheks:

1. Põhiülesanne – Protsessi juhtimine, ventilaatori töö juhtimine.
2. Kõrvalülesanded – Tegevused mis ei ole otseselt vajalikud põhiülesande täitmiseks:
    a. Kasutajaliides – Võimaldab lihtsamini määrata tööparameetreid ja jälgida süsteemi tööd, kuid kõik vajaliku saab kirjutada ka otse programmi.
    b. Alarmide edastus – Edastab teavitusi protsessi ebanormaalsetest olukordadest, kuid selle kaudu ei mõjutata protsessi.

Lahenduse loomisel on oluline keskenduda põhiülesande funktsionaalsuse tagamisele.

Järgnevas peatükis kirjeldatakse toimiva lahenduse jaoks olulisi riistvaralisi komponente.

2. Riistvara valik

Riistvara valikul on vaja arvestada ettevõtte tegevusvaldkonnaga, kasutatavate tehnoloogiate ja nende maksumusega. Lahendus peab täitma seotud osapoole seatud nõudeid ja eesmärke ning seda võimalikult soodsalt. Järgnevates peatükkides on kirjeldatud riistvara komponente ja nende valiku tingimusi.

2.1. Riistvarakomponendid

Töö raames loodud süsteem kogub ümbritsevast keskkonnast informatsiooni, töötleb seda ja reageerib vastavalt loodud algoritmile. Lahendus eeldab vähemalt järgnevate riistvaraliste seadmete olemasolu:

1. Kontroller – kogub ja töötleb informatsiooni vastavalt programmeeritud algoritmile;
2. Andur – mõõdab keskkonna parameetrit ja edastab selle kontrollerisse;
3. Täiturmehhanism – mehhaaniline seade, mida kontroller saab juhtida [15].

Järgnevates peatükkides on lühidalt kirjeldatud riistvarakomponente. Nende valikul on silmas peetud ettevõtte vajadusi ja juba kasutusel olevaid tehnoloogiaid.

2.1.1 Andurid

Tööstuses kasutusel olevad andurid võib jagada väljundi järgi kolme kategooriasse:

1. Digitaalse väljundsignaaliga – Digitaalse väljundiga anduri väljundsignaal saab olla kas „1“ või „0“, signaal on või ei ole.
2. Analoogväljundsignaaliga - Analoogandurid teisendavad keskkonnast mõõdetud suuruse signaaliks, üldjuhul kas 0…10V (või 2…10V) või 4…20mA (või 0…20mA).
   Pingeväljundiga andurid on häiretundlikumad ja neid ei ole soovitatav kasutada olulistes mõõtmissõlmedes. Eeliseks on kasutusmugavus, anduri mõõteväärtust on võimalik kontrollida voltmeetriga anduri ühendusklemmidelt.
   Voolutugevust mõõtvad andurid on häirekindlamad, kuid anduri väljundi kontrollimiseks on tarvis ampermeeter ühendada vooluringi jadamisi. Juba töös olevas süsteemis tähendaks see vooluringis katkestust, mida ei ole alati lihtne teostada. Lisaks on täpne milliampermeeter kulukas seade.
3. Andmesideväljundiga - vähesel määral on kasutusel ka andureid, mis annavad mõõtesignaali edasi kasutades andmesideprotokolli (näiteks Modbus RTU). Andmesideprotokolli kasutavad andurid on funktsionaalsemad, võimaldades tihtipeale edastada lisaks mõõteparameetrile ka muud informatsiooni näiteks alarmväärtusi. Puuduseks on keerulisem anduri töö kontroll ja integreerimine automaatjuhtimissüsteemiga.

Töös on kasutatud digitaalse- ja analoogväljundsignaaliga andureid.

Järgnevalt on lähemalt kirjeldatud tehnoloogilise protsessi juhtimiseks mõeldud kontrollerit.

2.1.2 Kontroller

Automaatikakontrolleri ülesanne tehnoloogilises süsteemis on protsessi jälgida, juhtida, ja edastada vajadusel alarme või muud informatsiooni kasutajale või teistele seadmetele.

Kontrollerisse tehtud programmi (koodi) ja loogika testimiseks on üldjuhul kontrolleris simulatsioonikeskkond (võimalused sõltuvad kasutatava kontrolleri funktsionaalsusest). Testimise käigus saab kontrollida programmi tööd ja algoritmi toimimist. Simulatsioonikeskkond ei suuda asendada reaalseid riistvaralisi seadmeid ja arvestada nende iseärasusi, eriti kui riistvara kasutab infoedastuseks andmesideühendust.

Automaatikakontrollerisse (PAC) programmeeritakse funktsionaalsus vastavalt tehnoloogia juhtimisvajadusele. Tuntuimad kaubamärgid, mida Eestis kasutatakse on näiteks Siemens, Schneider-Electric, Unitronics, Wago, Omron, Weidmüller, Allen-Bradley, Mitsubishi jm. Igal tootjal on mitu erinevat kontrolleri seeriat ja nende programmeerimiseks kasutatakse erinevaid arendustarkvarasid ning programmeerimiskeeli. Kasutatavate tarkvarade ja nende versioonide rohkus ning erinevad ühildusprobleemid (näiteks vanema arenduskeskkonna versiooniga tehtud programm ei pruugi sobituda uuema versiooniga) vajavad pidevalt kasutaja tähelepanu, kes peab kursis olema paljude erinevate tootjate ja nende toodete eripäraga.

Järgnevas peatükis on täpsemalt kirjeldatud kontrolleri valikut.

2.2. Kontrolleri valik

Kontrolleri valikul on automaatjuhtimissüsteemi katsestendi otstarvet silmas pidades tegemist vahetatava seadmega, mis sõltub programmeerija hetke vajadusest. Ettevõte kasutab enamasti Schneider-Electric tootja M241 seeria kontrollerit, mille programmeerimistarkvara põhineb CODESYS platvormil. Väiksemal määral on kasutatud sama tootja M221 seeria seadet, mille programmeerimiskeskkond on tootja enda välja arendatud. Sõltuvalt tellija poolt esitatud nõuetest on kasutatud ka Siemens S7-1200 seeria tooteid ning vähesel määral ka S7-300, S7-400 ja S7-1500 seeria mudelit. M241 ja S7 seeriate programmeerimistarkvarade litsentsid on tasulised. Kuna töö autor ei tegele igapäevaselt tööstuskontrollerite programmeerimisega, siis puudub võimalus neid tarkvarasid lõputöös kasutada.

Automaatjuhtimissüsteemi testlahenduse väljaarendamiseks, on tehtud otsus kasutada kontrollerit, millega ettevõtte töötajad pole seni kokku puutunud. Valitud on tootja Weidmüller kontroller UC20-WL2000-AC (spetsifikatsioon on esitatud lisas II). Seadme teeb omapäraseks sisseehitatud arenduskeskkond, mis tähendab, et programmeerijal ei pea olema arvutis eraldi installeeritud arendustarkvara. Seadmesse integreeritud arendustarkvara on tööstusautomaatikas uudne lähenemine ja sobib seatud lõputöö ühe eesmärgiga, milleks on automaatjuhtimissüsteemi katsetamine. Tähelepanuväärsemad andmed kontrolleri kohta on väljatoodud järgnevalt [16] [17]:

1. Platvormi ja seadmeneutraalne programmeerimine, seadistamine ja parameetrite määramine vastavalt IEC 61131-3 standardile. Standardis kirjeldatakse kontrolleri programmeerimiskeeli ja arhitektuuri, täpsemalt kirjeldatakse kahte tekstilist keelt:
    a) käsulist (*Instruction List*) ehk IL;

b) struktureeritud tekst (*Structured Text*) ehk ST;

ja kahte graafilist programmeerimiskeelt:

a) redelskeem ehk kontaktaseskeem (*Ladder Diagram*) ehk LD;
b) funktsiooniplokkskeem (*Function Block Diagram*) ehk FBD [18] [6].

Valitud kontroller võimaldab nimetatutest kasutada ainult FBD programmeerimiskeelt.

2. Kasutatakse standardiseeritud veebitehnoloogiaid nagu HTML5, CSS ja JavaScript. Programmeerimiskeskkond on kontrollerisse integreeritud ja ligipääsetav läbi HTML5 toetava veebibrauseri. Node-RED keskkonnas on kasutusel JavaScript.
3. Node-RED arenduskeskkond IoT lahendustele, OPC UA server ja sisseehitatud visualiseerimiskeskkond.
   Lisaks FBD programmeerimiskeelele on kontrollerisse integreeritud ka asjade interneti (IoT) jaoks suunatud Node-RED keskkond.
   OPC UA on avatud standard, mis kirjeldab infovahetust tööstuslikus andmesides [10]. OPC UA võimaldab masinalt masinale (*machine to machine* ehk M2M) suhtlust näiteks SCADA platvormi ja kontrolleri vahel [10].
4. Laiendatav tarkvaraliste lisamoodulitega, näiteks: OPC UA või Node-RED.

Kontrolleril on kaks Ethernet TCP/IP liidest ja USB mikro B liides, aga mitte ühtegi signaali sisendit ega väljundit. Erinevate andurite ühendamiseks on tarvis lisamooduleid digisisendite, -väljundite ja analoogsisendite jaoks. Erinevate moodulite hinnavahed on väikesed ja otsus on tehtud eelkõige kättesaadavust silmas pidades. Digisisendite jaoks kasutatakse moodulit UR20-16DI-P (16 digitaalset sisendit [19]), digiväljundite jaoks UR20-16DO-P (16 digitaalset väljundit [20]) ja analoogsisendite jaoks UR20-4AI-UI-12 (4 analoogsisendit [21]). Viimane moodulitest lubab iga sisendi seadistust muuta tarkvaras ja need võimalused on välja toodud järgnevas tabelis.

Tabel 2.1. Analoogmooduli sisendi võimalikud seadistused [21]

Voolutugevus, mA	Pinge, VDC
0…20	0…5
4…20	0…10
	1…5
	2…10
	±10

Valitud analoogmooduli kasutamisel on võimalik täita kaks süsteemile esitatud nõuet, milleks on pinge- ja voolusignaali mõõtmise võimekus.

Kontrolleri mõiste all on mõeldud järgnevat kooslust:

1. Põhimoodul, kus paikneb protsessor, mälu, andmesideliides ja mõnel juhul ka signaali sisendid ja väljundid;
2. Lisamoodulid, mida saab juurde valida vastavalt vajadusele.

Tegemist on modulaarse seadmega, kus moodulite tüüp ja arv valitakse vastavalt projekti vajadustele ja siinjuhul viidatakse kogu komplektile kui kontrollerile.

Lisaks kontrollerile, on toimiva süsteemi jaoks vaja ka teisi seadmeid, mida on kirjeldatud järgnevas peatükis.

2.3. Muud riistvaralised komponendid

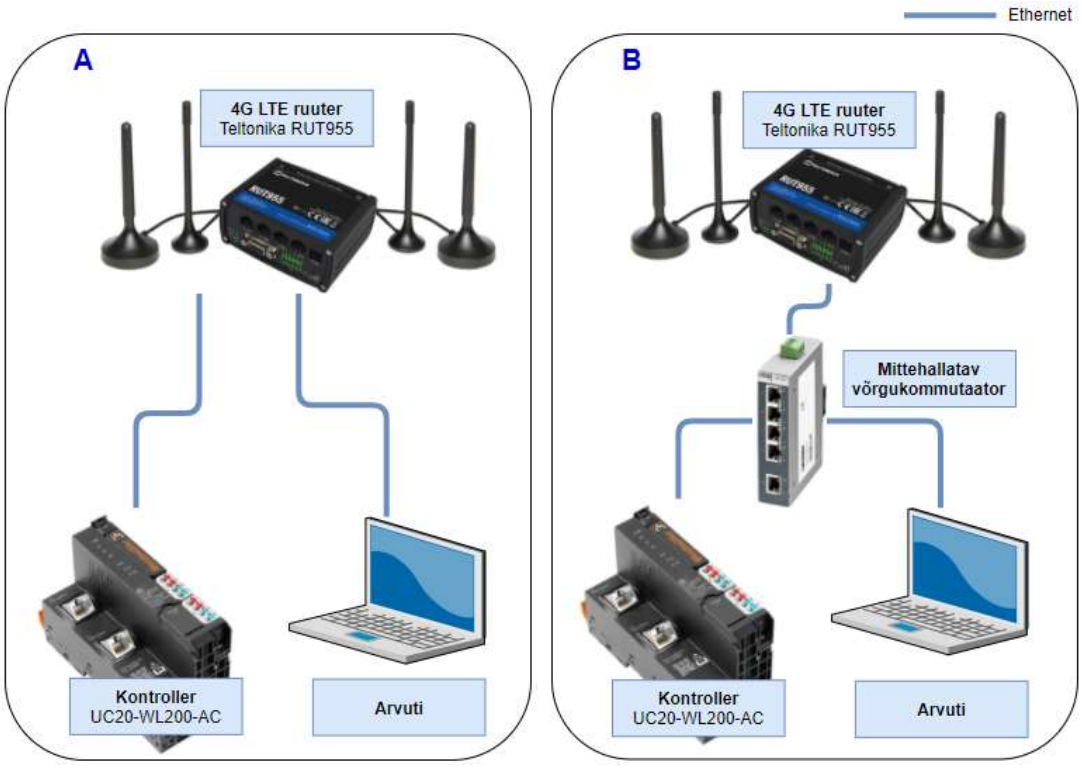
E-kirja teel häirete edastuse jaoks, peab kontrolleril olema internetiühendus. Internetiühenduse võib siinjuhul tekitada kahel viisil:

1. Ühendada kontroller internetiga kohtvõrku, kasutades selleks seadme ühte kahest Ethernet TCP/IP liidesest;
2. Kasutada stendis eraldi ruuterit.

Tööstuses kasutatakse võrguliidesega seadmete puhul tihtipeale fikseeritud IP aadressi, ja seda lähenemist kasutatakse ka valitud kontrolleri puhul. Stend on olemuselt mobiilne seade ja kui see ühendada kellegi kohtvõrku võib tekkida vajadus kontrolleri võrguseadete muutmiseks. Otstarbekas on kasutada stendis eraldi ruuterit, mis loob kontrollerile ja teistele stendi seadmetele oma alamvõrgu.

Andmeside ruuterina kasutatakse Teltonika RUT955 seadet, mis pakub 4G LTE andmesideühenduse võimalust (kuni 150Mbit/s) [22]. Seadmel on neli Ethernet TCP/IP võrguliidest, jadaliides, sisendväljundsignaalide liides, kaks mobiilside ja kaks Wi-Fi antenni.

Süsteemis on kommutaatori rolli võimeline täitma valitud Teltonika RUT955 ruuter, kuid seadme taaskäivitamise ajal katkeb võrgus andmesideühendus. Probleemi paremaks kirjeldamiseks on esitatud joonisel 2.1 kaks võrgu ülesehituse näidet.



Joonis 2.1. Andmesidevõrgu ülesehituse näited

Joonisel 2.1 näidatud variandi „A“ korral on arvuti ja kontrolleri andmevahetus sõltuv ruuterist. Ruuter võib vajada taaskäivitust (kas tarkvara sisest rebuutimist või toitekatkestust) näiteks võrguseadete muutmisel või interneti teenusepakkuja tegevuste tagajärjel. Taaskäivitus võib võrguseadmel võtta aega umbes üks minut, mille vältel ei toimu andmevahetust arvuti ja kontrolleri vahel ning kasutajal puudub ülevaade, mis tehnoloogilises protsessis toimub.

Variant „B“ korral on arvuti ja kontroller ühendatud mittehallatava võrgukommutaatoriga (switch), mis omakorda on ühendatud ruuteriga. Arvuti ja kontrolleri vaheline andmeside toimib ka ruuteri taaskäivitamisel või seadme rikke korral. Lahenduses kasutatud võrgukommutaatori Phonenix-Contacts 2891001 toitekatkestuse korral taastub andmeside mõne sekundi jooksul. Kommutaator tekitab süsteemile valmiduse lisada juurde seadmeid, mis kasutavad andmevahetuseks TCP/IP protokolli Ethernet võrgus.

Praktikas kasutatakse ka ruuteri automaatset taaskäivitust, mis võib olla:

1. Perioodiline – taaskäivitus tehakse kindla perioodi tagant (ruuteris saab seadistada nädalapäevad ja kellaaja);
2. Sündmusepõhine – ruuter pingib perioodiliselt võrgus asuvat sihtkohta. Kui vastust ei õnnestu saada teatud kordade ja perioodi jooksul, teeb ruuter taaskäivituse.

Magistritöös valmivas lahenduses on arvestatud variandis „B“ näidatud ülesehituse põhimõtetega, kus on kasutatud mittehallatavat võrgukommutaatorit. Täpsemalt on struktuuri kirjeldatud peatükis 3.

Analooganduri mõõtesuurust peab automaatjuhtimissüsteemi kasutajal olema võimalik mõjutada ja seeläbi näidu väärtust muuta. Lahenduses on kasutatud temperatuuriandurit PT100 koos signaalimuunduriga. Signaalimuundur teisendab mõõdetud takistuse alalispinge või voolu signaaliks, mis edastatakse kontrollerisse.

Töö käigus valminud automaatjuhtimissüsteemi testimise stend koos komponentidega on esitatud lisas III.

Ettevõtte vajadusi arvestav ja töös kasutatud võrgu arhitektuuri on kirjeldatud järgnevas peatükis.

3. Võrgu arhitektuur

Andmesidevõrk tööstuses peab võimaldama süsteemis olevatel seadmetel omavahel suhelda, kasutades tööstuses levinud andmevahetusprotokolle. Magistritöös valitud seadmed kasutavad andmevahetuseks TCP/IP protokolli Ethernet võrgus. Peatükis on kirjeldatud töös kasutatavat andmeside struktuuri ja selle erinevaid kihte.

3.1. Andmesidevõrk

Tööstuslike andmesidevõrkude puhul tuleb üha rohkem tähelepanu pöörata võrgu turvalisusele. Järjest enam jõuab uudiseid meediasse selle kohta, kuidas on rünnaku alla sattunud erinevad elutähtsaid infrastruktuuri osad nagu elektrijaamad, veevarustus ja katlamajad.

Turvakaalutlustel on oluline eraldada ärivõrk (*business Network* või *Enterprise Network*) ja tööstusvõrk (*industrial Network*). Ärivõrgu all mõeldakse infoinfrastruktuuri ühendavat võrku (müük, turundus, klienditugi, raamatupidamine jne) [23]. Tööstusvõrk on tehnoloogilisest protsessist otseselt osa võtvaid seadmeid ühendav andmesidevõrk [23]. Võrkude lahutamine aitab vähendada võimaliku rünnaku mõju ettevõtte süsteemidele.

Näiteks võib tuua joogiveetoomise teenust pakkuva ettevõtte, mille tegevuse võib jagada kaheks:

1. Kontori pool (administratiivtegevused) nagu näiteks klienditugi, raamatupidamine ja teised äriks olulised, kuid mitte joogivee tootmiseks olulised osad [23].
2. Tootmise pool kuhu kuuluvad protsessi juhtimiseks kontrollerid, sagedusmuundurid ja SCADA süsteem [23].

Kontori poolel olevaid seadmeid (arvuteid, nutiseadmeid) kasutavad inimesed ja tootmise poole peal olevad seadmed suhtlevad omavahel ilma inimese osaluseta (M2M). Küberturvalisuse seisukohast on üheks nõrgimaks lüliks inimene ja nende mõju esimeses loetelupunktis on tunduvalt suurem kui teises. Tootmisvõrgu turvalisust mõjutavad ennekõike seadmete turvauuenduste paigaldamine ja võrgu ülesehitus.

Kui need kaks (kontor ja tootmine) andmesidevõrku on teineteisest eraldatud, kasutades kas tulemüüri või füüsilist võrkude eraldamist, siis rünnaku alla sattunud võrk ei pruugi mõjutada teise võrgu toimimist. Näiteks on joogiveevarustuse tagamine kriitilise tähtsusega teenus ja on oluline, et sellise teenuse osutamisel ei toimuks ootamatuid katkestusi. Kui rünnaku alla satub kontori võrk, siis joogiveevarustuse tagamise teenuse häirimine on ebatõenäoline. Kui kontor ja tootmine oleksid samas andmevahetusvõrgu, on teenuse häirimine tunduvalt lihtsam.

Valitud kontrolleril on kaks võrgukaarti, mis võimaldab luua kaks eraldi võrku. Võrkude vaheline andmevahetus saab sel juhul käia ainult läbi kontrolleri. Protsessis osalevad seadmed koondatakse tihtipeale ühte alamvõrku, mis on turvalisuse kaalutlustel eraldatud ülejäänud võrgust. Kõige turvalisem on protsessi seadmete võrku mitte ühendada internetiga. Alarmide edastus e-kirja teel vajab internetiühendust ja see võrk peab olema eraldi protsessis osalevate seadmete võrgust. Töö käigus loodud süsteem ei võimalda selle katsetamist, küll aga on loodud kõik eeldused selle edaspidiseks rakendamiseks.

Töö arhitektuur ei näe ette ärivõrku, kõik on tööstusvõrgu tasemel. Tööstusvõrk omakorda jaguneb mitmeks alamvõrguks ehk kihiks. Alamvõrkudeks jaotamise eesmärk on vähendada protsessi juhtimiseks mitte vajaliku info liiklust tööstuslikus andmesidevõrgus. Töö raames võib võrgud jagada järgnevatesse kihtidesse [24] [25]:

1. Sisend/väljundseadmete (*input/output* ehk I/O) kiht;
2. Juhtimise kiht;
3. SCADA kiht.

Töös ei ole kõige madalamal tasemel (I/O kihil) kasutusel andmesideprotokolli kasutavaid seadmeid. SCADA süsteemi töös ei rakendata ja kontrolleri sisene visualiseerimine asub juhtimiskihil. Projektis oleva juhtimiskihi ühendamisel ärivõrguga või välisvõrguga tuleb kasutada tulemüüri. Elementaarse tulemüürina saab kasutada Teltonika RUT955 ruuterit.

Andmesidevõrgu erinevaid kihte on täpsemalt kirjeldatud järgnevates alampeatükkides.

3.1.1 Sisend- ja väljundseadmete kiht

I/O kihil on seadmed, mis paiknevad objektil (*field devices*) ja võtavad otseselt osa tehnoloogilise protsessis juhtimisest või jälgimisest. Seadmed on ühendatud järgmises kihis paikneva kontrolleriga. Ühenduse tüüp võib olla:

1. Elektriline näiteks digitaal- või analoogsisend/väljund signaalid;
2. Jadaside protokoll, näiteks Modbus RTU;
3. Ethernet võrgul põhinev TCP/IP protokolli kasutav andmesideühendus, näiteks Modbus TCP/IP.

Tööstusautomaatika lahenduste programmeerimise lähteülesandeks on tehnoloogiline projekt. Automaatjuhtimissüsteemi katsestendi puhul projekt puudub ja funktsionaalsus tuleb üles ehitada vastavalt esitatud nõuetele ja loodud tööülesandele. Süsteemi I/O kihi ülesehitus on kirjeldatud järgnevalt:

1. Analoogsisendsignaal 4…20mA – kontrolleri analoogsignaalide sisendmooduli sisend on seadistatud milliamper signaalile. Signaali teisendamiseks ja edastamiseks kasutatakse signaalimuundurit (*transmitter*), mille sisend mõõdab elektrilist takistust ja väljund on seadistatud milliamper signaalina.
2. Digitaalsed sisendsignaalid – kontrolleril on digitaalsete sisendsignaalide moodul, mille tööpinge on 24VDC. Signaalile omistatakse väärtus „1“ kui sisendis on pinge üle 11V ja väärtus „0“ kui pinge langeb alla 5V [19]. Kui sisendis on pinge väärtus vahemikus 5…11V säilitatakse eelmine olek [19].
3. Digitaalne väljundsignaal – kontrolleril on digitaalsete väljundsignaalide moodul. Väljund aktiveeritakse vastavalt programmiloogikale.

Levinud seadmed I/O kihil, mis kasutavad info edastuseks andmesidet on näiteks sagedusmuundurid ja kulumõõtjad (näiteks soojusenergia arvestid ja veekuluarvestid).

I/O kihilt liigub info edasi juhtimise kihile, mida on kirjeldatud järgmises peatükis.

3.1.2 Juhtimise kiht

I/O kihist kõrgemal paikneb juhtimise kiht, kus asuvad protsessi juhtimis- ja sideseadmed. Kontroller kogub kokku info madalama kihi seadmetelt ja juhib protsessi vastavalt algoritmile ja vajadusel edastab alarme e-kirja teel kasutades selleks ruuterit. Süsteemi juhtimiskihi ülesehitus on kirjeldatud järgnevalt:

1. Kontroller – seadmel on kaks eraldi võrguna seadistatavat Ethernet TCP/IP liidest. Programmeerimise vabadusastmete järgi saab kontrollereid jagada kaheks:
    a. Vabalt programmeeritav kontroller – kasutaja saab luua programmi vastavalt oma äranägemise järgi kasutades programmeerimiskeeli (näiteks vastavalt standardile IEC 61131-3).
    b. Programmeeritav kontroller – võimaldab kasutada eelprogrammeeritud programme ja funktsioone ning seadistada ettenähtud parameetreid, programmi võimalused on piiratud vastavalt tootja poolt ettenähtud funktsionaalsustega.

       Terminit PLC või PAC kasutatakse mõlemas ülaltoodud tähenduses, kuid vastavalt seadme funktsionaalsusele on oluline neid eristada. Töös valitud seadet UC20-WL2000-AC saab programmeerida IEC 61131-3 standardis loetletud FBD keelega, mis kategoriseerib selle seadme vabalt programmeeritavaks kontrolleriks.
2. 4G/LTE ruuter – häireedastusega seotud e-kirjade saatmiseks, ühtlasi tekitab kontrollerile internetiühenduse.
3. Mittehallatav võrgukommutaator – võimaldab süsteemi lisada võrguliidesega seadmeid.

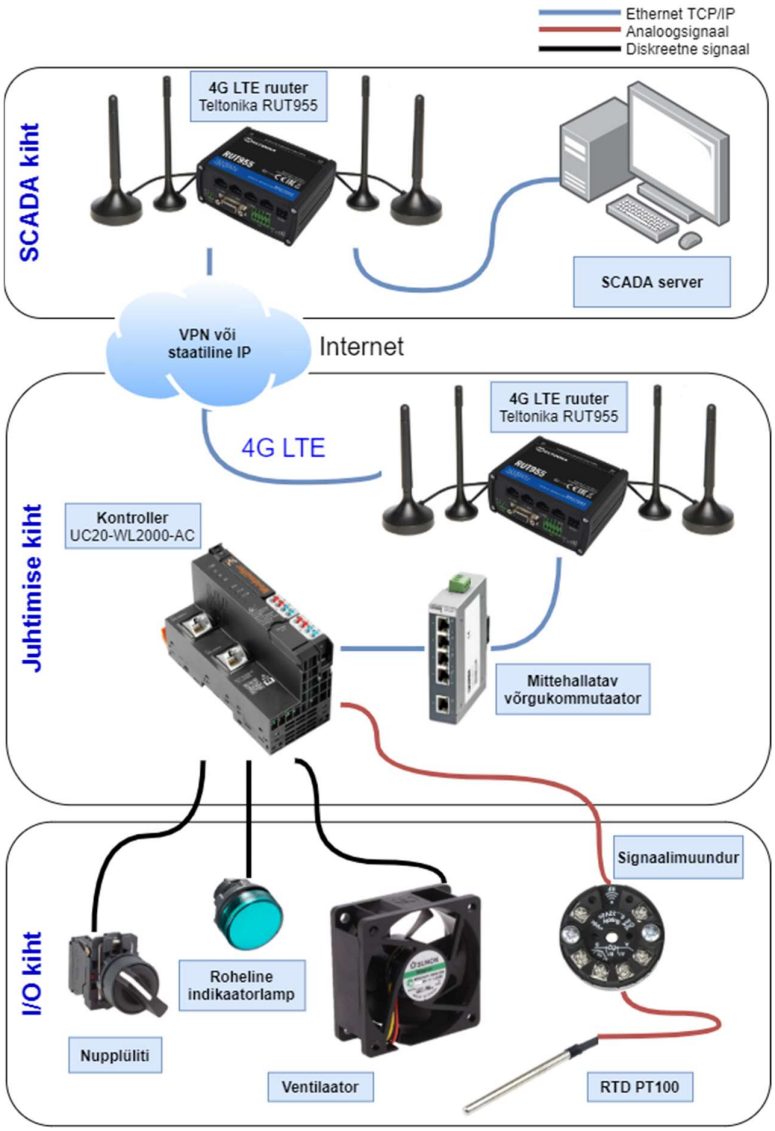
Juhtimise kihil võib paikneda veel teisi seadmeid, mis ülalnimetatud loetelust puuduvad. Näiteks operaatorpaneel, mis on mõeldud protsessi jälgimiseks ja juhtimiseks kohapeal. Juhtimise kihilt edastatakse infot tihtipeale ülemisele, kaugjälgimissüsteemi kihile, mida on kirjeldatud järgmises peatükis.

3.1.3 Kaugjälgimissüsteemi kiht

Kaugjälgimissüsteemi kihil paikneb SCADA server, mis visualiseerib I/O kihilt saadud info. Omab kasutajaliidest, mille kaudu on operaatoril võimalus tehnoloogilist protsessi jälgida, seadeparameetreid muuta ja protsessi juhtida. SCADA server kogub protsessi kohta andmeid, mille abil on võimalik tehnoloogilist protsessi analüüsida.

Väiksemates süsteemides võib SCADA ja kontrolleri kihi lugeda üheks juhtimise kihiks. Hajusa süsteemi puhul, kui SCADA asub kontrollerist eemal ja andmeside kahe seadme vahel toimub läbi ruuterite, võib olla oluline struktuuris eraldada SCADA ja juhtimise kiht. Visuaalne eraldamine joonisel võimaldab paremini rõhutada, et SCADA server ja kontroller ei ole võrgukaabliga otse ühendatud, vaid andmevahetus toimub läbi interneti, mistõttu tuleb lisa tähelepanu pöörata võrgu turvalisusele.

Katsestendis on kontroller ühendatud 4G ruuteriga, mis kasutab interneti ühenduse tekitamiseks staatilise IP'ga SIM kaarti. Joonisel 3.1 esitatud struktuur iseloomustab täpsemalt ettevõtte projektides kasutatavat struktuuri. Stendis kasutatud lahendus piirdub I/O ja juhtimiskihiga. Magistritöös SCADA süsteemi lähemalt ei käsitleta.



Joonis 3.1. Võrgu ja seadmete struktuur

Joonisel 3.1 on esitatud näide ettevõtte poolt kasutatavast kolmekihilisest võrgu struktuurist, kus SCADA paikneb eraldi kihil ja on ühendatud protsessi juhtiva riistvaraga, kasutades selleks 4G ruuterit. SCADA serveri ja kontrolleri vahelise andmeside loomiseks on kasutusel kaks võimalikku lahendust:

1. Andmevahetuseks SCADA ja objekti kontrolleri vahel kasutatakse objekti ruuteris staatilist IP aadressi. Kui SCADA arvutil on olemas juhtmega internetiühendus, siis kasutatakse teenusepakkuja ruuterit. SCADA server ei vaja siinjuhul staatilist IP aadressi.
2. Andmevahetuseks kasutatakse Open VPN tunnelit. Objektidel paiknevad VPN otspunktid ja SCADA süsteemi juures VPN server, mis vajab staatilist IP aadressi.

Andmesideturvalisuse seisukohast on eelistatum viimane variant.

Juhtimise kihil paiknevat kontrollerit on kirjeldatud järgmises peatükis.

4. Kontroller

Kontrolleri programmeerimiskeskkond on seadmesse integreeritud ja seda kuvatakse läbi HTML5 toetava veebilehitseja. Enne kontrolleri programmi kirjutamist on otstarbekas uuendada püsivara (*firmware*), mis lisab seadmele funktsionaalsust, parandab vigasid ja süsteemi stabiilsust ning lisab turvauuendusi.

Weidmülleri kontrolleril UC20-WL2000-AC on 512MB muutmälu (*random-access memory*) ja 4GB püsimälu (*flash memory*) [4].

Järgnev informatsioon on saadud toote kasutajatoelt e-kirja teel. Muutmälust on umbes 100MB reserveeritud süsteemi toimimiseks (*overhead*). Ülejäänud osas hoitakse reaalajaprogrammi, visualiseerimiskeskkonnas koostatud kasutajaliidest ja Node-RED programmi. Püsimälus hoitakse kontrolleri operatsioonisüsteemi, arendustarkvara ja püsivara. Reaalaja programmi tsüklite läbimiskiirus on seadmes fikseeritud 1ms.

Kontrolleri ekspluatatsiooni käigus töötati läbi seadme spikker dokument (*Help*) [26].

Programmeerimisele ja visualiseerimiskeskkonna kasutamisele eelnevad tegevused on kirjeldatud järgmistes peatükkides.

4.1. Eelseadistus

Seadmel on kaks Ethernet TCP/IP võrguliidest, tähistusega X1 ja X2, ning üks USB liides (USB mikro B). Mõlemale võrguliidesele saab määrata eraldi võrguseaded, mis on oluline süsteemides, kus on vajalik võrgud teineteisest eraldada, näiteks:

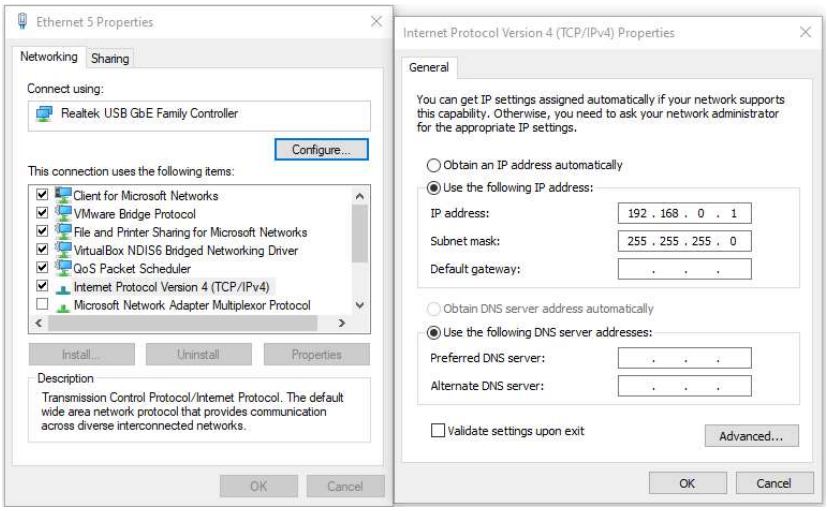
1. Sisevõrk – Tehnoloogilise protsessi juhtimise tasandi võrk, kus on kontroller ja muud protsessi juhtimiseks kasutatavad võrguliidesega seadmed;
2. Välisvõrk – Välisvõrk võib omada internetiühendust. Kaughaldussüsteemiga võrgus on näiteks SCADA, mis asub füüsiliselt objektist eemal ja millega ühendumiseks kasutatakse 4G andmeside ruuterit.

Võrkude lahusus aitab piirata ligipääsu välisvõrgust sisevõrku.

Kontrollerile esmaseks ligipääsuks on kaks võimalust, kasutada Ethernet võrguliidest X1 või USB liidest. Järgnevas peatükis on kirjeldatud ligipääsu loomist kontrolleri arenduskeskkonnale.

4.1.1 Ligipääs arenduskeskkonnale

Kasutades ligipääsuks liidest X1 on selle vaikimisi määratud IP aadress 192.168.0.101 ja USB kasutades on ligipääsuks aadress 192.168.10.202. Kontrolleri ja arvuti ühendamiseks kasutatakse LAN kaablit, mille mõlemas otsas on RJ45 pistik. Kontrollerile ligipääsemiseks peavad seadmed asuma samas alamvõrgus. Arvuti võrgukaardi IPv4 seadetes omistatakse IP aadress käsitsi. Kontrolleriga ühendatud arvuti võrgukaardi seadistus on näidatud joonisel 4.1.



Joonis 4.1. Arvuti võrgukaardi IPv4 seadistus

Arvuti ja kontrolleri IP aadressid peavad kuuluma samasse alamvõrku. Kontrolleri liidese X1 aadress on vaikimis juba määratud ja seda ilma ligipääsuta muuta ei saa (192.168.0.101).

Järelikult on vaja muuta arvuti IP aadress sobivaks. Lähtudes kontrolleri võrguaadressist on vaja arvuti IP aadress seadistada vahemikku 192.168.0.1 kuni 192.168.0.254 välja arvatud aadress 192.168.0.101, mis on kasutatud kontrolleri poolt. Alamvõrgumask (*subnet mask*) on juba vaikimisi määratud 255.255.255.0. Vaikelüüsi (*default gateway*) ja nimeserveri (DNS *server*) määramine ei ole hetkel oluline, kuna ühendus ruuteri või internetiga puudub.

Alternatiiv on ühendada arvuti ja kontroller kasutades USB kaablit, sel juhul ei ole arvuti võrgukaardi ümberseadistus vajalik.

Võrgustruktuur näeb ette ruuteri kasutamist ja vajalik on ligipääs internetist. Ruuteri IP aadress alamvõrgus on 192.168.1.1, mis määrab ära teiste seadmete võimalikud IP aadressid selles võrgus. Esimese tegevusena muudetakse kontrolleri võrguliidese X2 seaded sobivaks, et see ruuteriga ühenduks. Uus IP aadress X2 liidesel on 192.168.1.101 ning lisaks määratakse ka vaikelüüsi aadress, milleks on 192.168.1.1. Andmeside struktuur ei näe ette teise võrgukaardi kasutamist ja liides X1 jääb vaikesätetega.

Kui välisvõrgust ligipääsuks kasutatakse joonisel 3.1 näidatud võrgustruktuuri on tarvis ruuteris suunata port 443 (HTTPS protokoll) seadmega ühendatud kontrolleri IP aadressile.

Kontrollerisse on eelseadistatud kaks erinevate õigustega kasutajat – *admin* ja *service* [26]. Töös seisukohast olulised kasutajate õigused on esitatud tabelis 4.1.

Tabel 4.1. Kontrolleri kasutajate õigused [26]

Funktsionaalsus	***admin***	***service***
Reaalaja programmi (FBD) rakenduse redigeerimine	X	
Reaalaja programmi käivitamine ja seiskamine	X	X
Ligipääs Node-RED rakendusele	X	
Ligipääs visualiseerimiskeskkonnaga loodu kasutajaliidesele[1)]	X	X
Kontrolleri seadete muutmine	X	
Kontrolleri tehaseseadete taastamine	X	

1) *Service* kasutaja saab ligipääsu kasutajaliidesele, lisades IP aadressi järele „/visu“. Täpsemalt peatükis 4.1.2.

Mõlema konto kasutajanimed ja salasõnad on esitatud seadme abidokumentatsioonis [26], seadme kasutusjuhendis on mainitud ainult *admin* kasutaja andmed [27]. Kasutajate lisamine, kustutamine ja õiguste muutmine ei ole kontrolleris võimalik.

Järgmises peatükis on kirjeldatud ligipääsuvõimalusi seadme visualiseerimiskeskkonna abil loodud kasutajaliidesele.

4.1.2 Ligipääs kasutajaliidesele

Kontrollerisse integreeritud visualiseerimiskeskkonna abil loodud kasutajaliidest kirjeldatakse peatükis 7. Kasutajaliidesele ligipääsemiseks on kaks võimalust:

1. Siseneda läbi arenduskeskkonna ja valides „*Visualisation*“ mooduli, mis on näha joonisel 4.2 sektsioonis „4“. Võimalik ainult *admin* kasutajale;
2. Siseneda otse kasutajaliidese esilehele kasutades selleks kontrolleri IP aadressi, mille lõppu sisestada „/visu“. Võimalik *admin* ja *service* kasutajale.

Välisvõrgust ligipääsemiseks peab kontrolleriga ühendatud ruuteris olema suunatud port 443 kontrolleri IP aadressile.

Enne uue kontrolleri põhjalikumat kasutamist on oluline teostada püsivara uuendus, mida on kirjeldatud järgmises peatükis

4.1.3 Püsivara uuendus

Kui kontrollerile arenduskeskkonnale on ligipääs tekitatud, siis järgmine samm on seadme püsivara uuendamine. Selle jaoks valitakse seadetest püsivara uuendus (*Firmware update*), mis järel tehakse kontrollerile taaskäivitus ja kasutaja suunatakse püsivara uuenduse lehele (*controller software update mode*), kus on eraldi aken püsivara manuse lisamiseks. Uue versiooni paigaldamiseks tuleb alla laadida tootja kodulehelt uusim püsivara, milleks on töö kirjutamise hetkel versioon 1.11. Uuendamise käigus esineb veateade, et manus on liiga suur (*File is too big*). Lähemalt analüüsides püsivara versioonide dokumendist (*release notes*), selgub, et kasutusel olev püsivara (versioon 1.8) on liiga aegunud ja selle pealt uusimale üleminekuks tuleb installida vahepealne püsivara versioon 1.10 [28].

Uuenduse järgselt on kontrollerile ligipääsuks kasutusel HTTPS protokoll (algselt kasutusel HTTP protokoll). Kontrollerisse sisse logimiseks on vaja kasutajal teada kasutajanime ja salasõna. HTTP protokolli puhul toimub andmeedastus veebilehitseja ja kontrolleri vahel teksti kujul ja krüpteerimata, mis potentsiaalselt võimaldaks sama alamvõrku kasutaval kõrvalisel isikul ligipääsu kasutajaandmetele (näiteks *man in the middle* ehk MITM rünnaku puhul). Uue püsivara kasutuselevõtuga rakendatakse HTTPS protokolli, mille korral toimub andmevahetus krüpteeritult. Nii Ethernet TCP/IP kui USB liidesega ligipääsu tekitamisel on kasutusel HTTPS protokoll.

Kui eelneva püsivara kasutamisel oli tehtud salasõna muudatus, siis viimasele versioonile (versioon 1.11) üleminekul taastati süsteemi vaikimisi kasutusel olnud kasutajanimi ja salasõna [28].

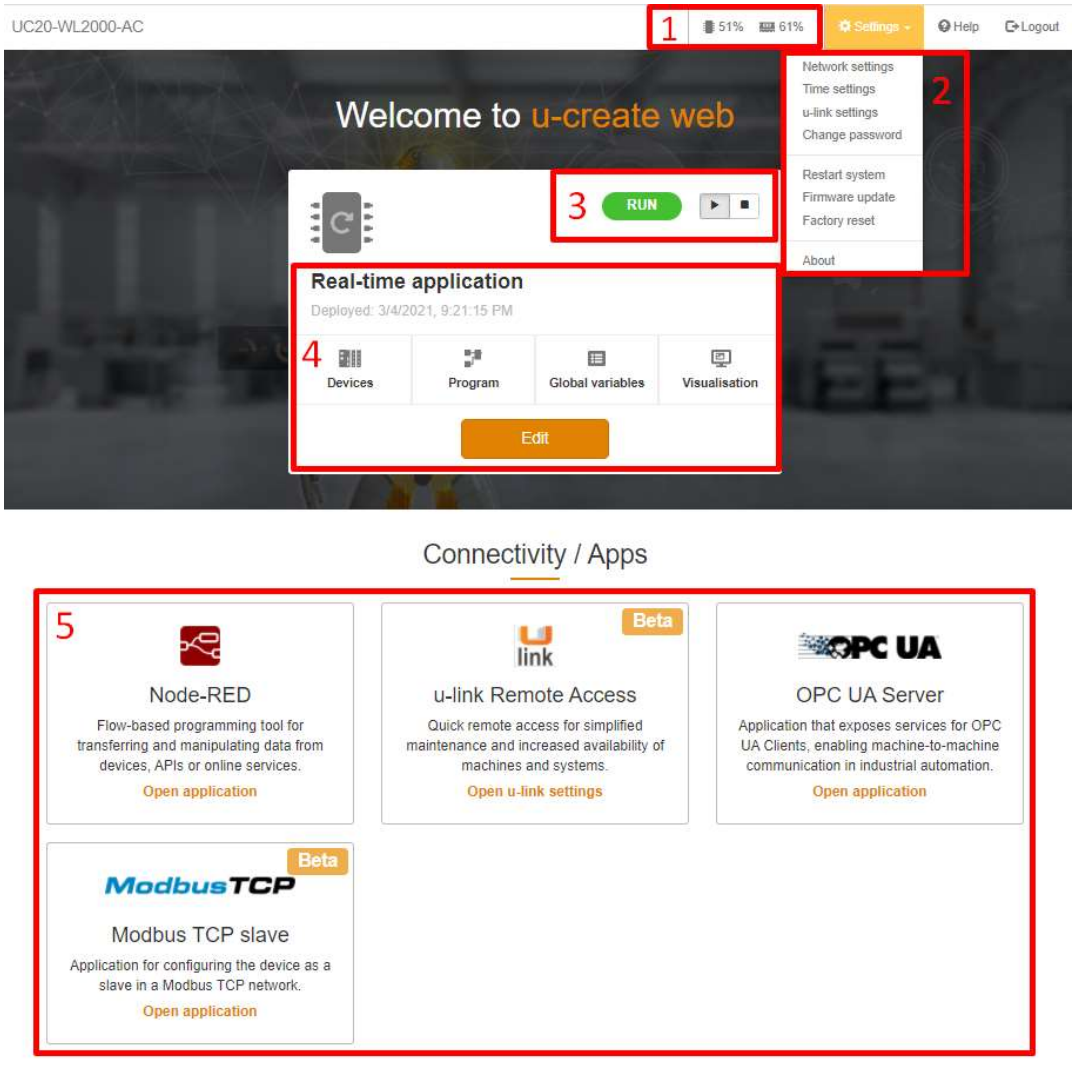
Arenduskeskkonna avalehte ja selle võimalusi on kirjeldatud järgnevas peatükis.

4.2. Kontrolleri arenduskeskkonna kirjeldus

Arenduskeskkond on kontrollerisse integreeritud. Kõik vajalikud seadistus ja programmeerimistegevused tehakse läbi HTML5 toega veebilehitseja. Arenduskeskkonnale ligipääsemiseks on vaja brauseri aadressireale sisestades kontrolleri IP aadress (vastavalt ligipääsuks kasutatud liidesele, vt peatükk 4.1.1). Seejärel avaneb aken, kus kasutaja peab ennast tuvastama, kasutades selleks ühte kahest eelseadistatud kasutajakontost (vt peatükk 4.1.1).

Kui konto salasõna ei ole teada, siis on ainus võimalus tehaseseadete taastamine. Viimast saab käivitada kasutaja tuvastamise vaatest, järgides juhendit mis avaneb lingilt *Forgot password*. Kasutajatuvastuse aken ja tehaseseadete taastamise juhend on esitatud lisas IV.

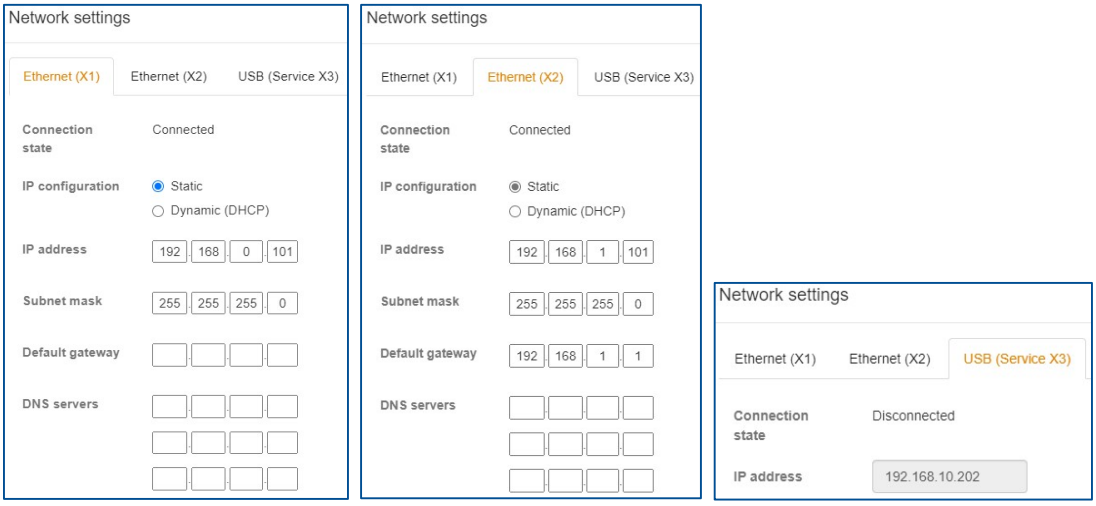
Sisse logides avaneb arenduskeskkonna põhileht. Keskkonna elemendid on näidatud joonisel 4.2.



Joonis 4.2. Kontrolleri arenduskeskkond

Töö käigus kasutatud keskkondade töölehed on näidatud lisas V. Järgnevalt on kirjeldatud joonisel 4.2 näidatud avalehe olulisemaid elemente:

1. Kuvatakse kontrolleri protsessori ja mälu kasutusprotsenti;
2. Seadete (*Settings*) menüüs on esitatud järgnevad alampunktid:
    a. Võrgu seaded (*Network Settings)* avab akna, mis on esitatud, koos vahelehtedega, joonisel 4.3.



Joonis 4.3. Kontrolleri võrguseaded

Jooniselt 4.3 on näha, et valikus on kaks Ethernet TCP/IP võrguliidest ja üks USB jadaliides. Mõlema võrguliidese seadistusvõimalused on identsed. „*Connection state*: *Connected*“ näitab, et võrguliides on ühendatud mingi välise seadmega (siinjuhul arvutiga). IP seadistus (*configuration*) on määratud vaikimisi staatiline (*Static*).

USB alamlehel toodud IP aadressi ei saa muuta, mis tagab kontrollerile ligipääsu ka siis, kui Ethernet võrguseaded ei ole mingil põhjusel teada.

    b. Aja seadistus (*Time settings*) menüü võimaldab kell ja kuupäeva käsitsi või automaatset seadistust. Automaatseks aja seadistamiseks kasutatakse NTP teenust, mis kasutab võrguaja protokolli ja see eeldab internetiühenduse olemasolu. Projektis on märgitud NTP serveriks pool.ntp.org.
    c. Püsivara uuendus (*Firmware Update*) võimaldab laadida uue püsivara versiooni (kirjeldatud lähemalt peatükis 4.1.3).
3. Kuvatakse kontrolleri FBD programmi olek, kas töös või peatatud (*run, stop)*, ning nupud oleku muutmiseks. Kui kontrolleri töö on peatatud, siis säilib kontrollerile ligipääs ja redigeerimisvõimalus, kuid kontrolleri programmi ei täideta.
   Redigeerimiskeskkonnas on võimalus programmi täitmine pausile panna, mis säilitab kõik hetke olekud ja väärtused. Pausi kasutamine annab võimaluse programmi täitmist teostada ühe tsükli kaupa, mis aitab rakenduse tööd täpsemalt jälgida ja vajadusel teostada silumist (*debugging*).
4. FBD programmeerimiskeskkonnas navigeerimise nupud. Arenduskeskkond eristab rakenduse kahte olekut:

a. Reaalaja rakendus (*Real-time application*) võimaldab tööolekus süsteemi jälgida ja kontrollida koodi ja algoritmi tööd. Kasutatud kontrolleri arenduskeskkond ei luba tarkvaraliste muutujate manipuleerimist reaalajas, mis võimaldaks erinevate olukordade simuleerimist.
b. Redigeerimine (*Edit*) võimaldab kontrollerit konfigureerida ja programmi koodi ning algoritmi luua, kuid selles olekus ei saa jälgida kontrolleri tööd. Muudatuste rakendamiseks tuleb programm uuesti peale laadida, mille käigus teostatakse ka koodi kompileerimine. Kui kompileerimise käigus tuvastatakse viga, siis pealelaadimine katkestatakse ja kontroller kasutab eelnevalt kompileeritud koodi.

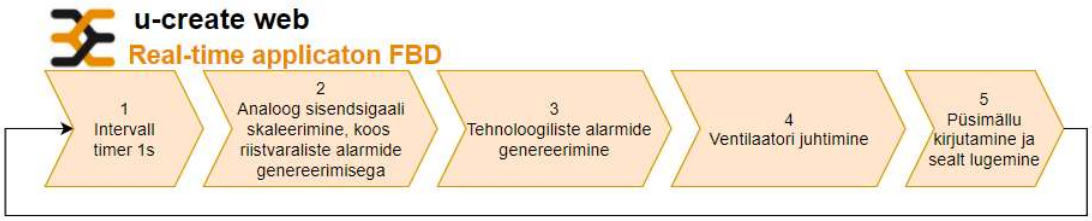
Reaalaja- ja redigeerimisrakenduse vaated on paralleelselt kasutatavad. Reaalajarakenduse vaated avatakse kõik brauseris eraldi alamlehel. Redigeerimislehti avatakse üks, mille sisu muutub vastavalt valitud teemale.

Arenduskeskkonda saab kasutada ainult reaalajas, mis tähendab, et arvuti peab olema kontrolleriga ühendatud kogu arendustegevuse vältel.

Järgnevas peatükis kirjeldatakse täpsemalt kontrolleri põhiülesande täitmiseks kasutatud FBD rakendust.

5. Kontrolleri funktsiooniplokkskeem rakendus

FBD rakendus on reaalaja programm (*real-time program*), mida kontroller täidab tsükliliselt ja pidevalt. Kontrolleri reaalaja programmi kirjutamiseks on arenduskeskkonnas ette nähtud funktsiooniplokkskeem (*Function Block Diagram* ehk FBD) programmeerimiskeel. Keel vastab standardile IEC 61131-3 ja on seal loetletutest üks kahest graafilisest programmeerimiskeelest [18]. Funktsiooniplokil on alati üks või rohkem sisendit ja üldjuhul ka vähemalt üks väljund [29]. Kontroller täidab programmi ülevalt alla ja vasakult paremale. Sellest lähtudes on oluline silmas pidada funktsiooniplokkide asetust lehel. Arenduskeskkonna programmeerimisala on fikseeritud suurusega. Sellise piirangu tõttu on oluline mahutada võimalikult suur osa koodist erinevate funktsiooniplokkide sisse, vastasel juhul ei pruugi programm lehele ära mahtuda. FBD programmi olulisemad ülesanded ja nende täitmise järjekord on esitatud joonisel 5.1.



Joonis 5.1. FBD programmi ülesanded ja nende täitmise järjekord

Joonisel 5.1 on toodud kokkuvõtlikult põhilisemad programmi sammud ja nende täitmise järjekord. Nende ülesannete jada läbimine on üks programmi tsükkel. Täielik FBD programm on esitatud lisas VI. Järgnevas peatükis on lähemalt kirjeldatud programmi muutujate tähistust ja selle põhimõtteid.

5.1. Muutujate kirjeldus ja tähistus FBD programmis

Kontrolleri sisenditele ja väljunditele, mis suhtlevad väliskeskkonnaga, on vaja omistada muutujad, et neid saaks kasutada programmeerimiskeskkonnas. Valitud kontrolleri arenduskeskkond ei võimalda sisendsignaali kasutada otse programmis. Selle jaoks on vaja defineerida vahemuutuja, mis lisab programmeerimise alguses tööd juurde, kuid lihtsustab hilisematest muudatustest tingitud ümbertegemisi. Kui kontrolleri riistvaralisel ühendamisel on näiteks kaks signaali omavahel vahetuses (seadme töö ja rike), siis füüsilise aadressi muutusest tingituna, tuleb terve programm üle vaadata ja teha parandus kohas, kus neid muutujaid on kasutatud. Kui kontrolleri signaali sisend on defineeritud eraldi sisemise muutujaga, siis piisab selle definitsiooni muutmisest ja puudub vajadus terve programmi läbivaatamiseks.

Muutujate ja teiste programmielementide tähistamisel ühtset standardit või head tava automaatika kontrollerite programmeerimiseks ei ole. Töös kirjutatud programmi parema struktureerimise huvides on lähtutud ühe tuntud tootja juhendmaterjalist, „*Programming style guide for SIMATIC S7-1200/ S7-1500*“, mis on leidnud laialdast kasutust tööstusautomaa-

tika valdkonnas, ka väljaspool nimetatud tootja seadmeid. Valitud kontrolleri funktsionaalsus erineb oluliselt juhendmaterjali kontrolleriseeria funktsionaalsusest. Järgnevalt on esitatud olulisem info juhendmaterjalist [30], mida saab rakendada magistritöös.

1. Juhendmaterjal nõuab inglise keele kasutamist muutujate ja teiste identifikaatorite määramisel.
2. Koodi muutujate tähistamisel kasutatakse *camelCase* reeglistikku. Muutujad kirjutatakse väikese algustähega. Kui muutuja koosneb mitmest sõnast, siis iga uus sõna algab suure tähega ja sõnad kirjutatakse kokku. Kirjavahemärkide või muude sümbolite kasutamine sõnade visuaalseks eraldamiseks ei ole lubatud.
3. Programmis kasutatud konstantide nimed kirjutatakse läbiva suure tähega. Visuaalse eraldamise eesmärgil on lubatud kasutada alakriipsu sõnade vahel. Siinjuhul rakendati seda meetodit funktsiooniploki sisestele muutujatele, mille väärtus on alati konstant. Kui muutujat kasutatakse funktsiooniploki enda sisendi või väljundina, siis ei arvestatud seda kui konstanti.
4. Funktsiooniplokkide nimetamisel kasutatakse *PascalCasing* reeglistikku. Mitmest sõnast koosneva nimetuse iga uus sõna on suure algustähega. Parema struktureerimise eesmärgil võib kasutada sõnade vahel alakriipsu, kuid mitte üle kolme). Funktsioonid ja plokid algavad tegusõnaga (näiteks „*Get*“, „*Set*“, „*Put*“, „*Find*“, „*Calc*“).

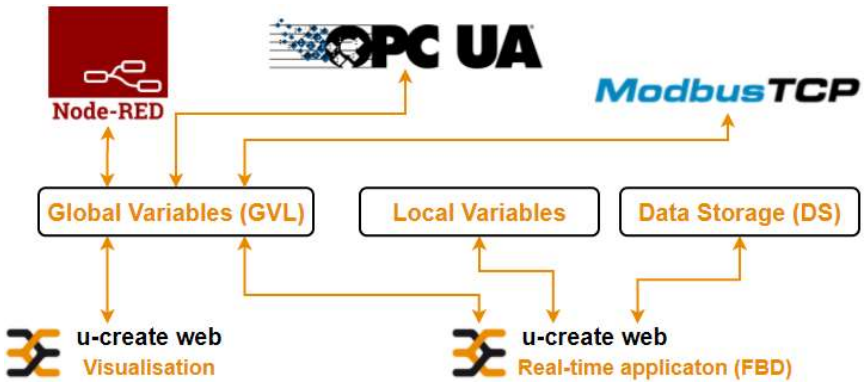
Töös kasutatud programmeerimiskeskkonnas on muutujad jaotatud kolme kategooriasse:

1. Lokaalsed muutujad – on kasutatavad ainult FBD programmeerimiskeskkonna piirides. Saab kasutada põhitöölehel ja funktsiooniplokkide sees. Kui lokaalset muutujat on kasutatud funktsiooniploki siseselt, siis ei ole see ligipääsetav väljastpoolt plokki (põhitöölehel olevast programmist või teisest funktsiooniplokist). Toitekatkestuse või programmilaadimise korral määratakse muutujatele vaikimisi seatud algväärtused.
2. Globaalsed muutujad – on kasutatavad FBD programmeerimiskeskkonna ulatuses ja ka väljastpoolt seda (näiteks Node-RED ja visualiseerimiskeskkonnast). Toitekatkestuse või programmilaadimise korral määratakse muutujale vaikimisi seatud algväärtus.
3. Püsimälu muutujad – on kasutatavad ainult FBD programmeerimiskeskkonna piirides. Ei ole kättesaadavad visualiseerimiskeskkonnast ega Node-RED keskkonnast. Püsimälu muutujatele on ligipääs nii funktsiooniploki seest kui põhitöölehe programmist. Muutuja väärtus säilitatakse püsimälus ka peale toitekatkestust ja programmilaadimist.

Programmeerimiskeskkond on muutujate tähistusele seadnud järgmised piirangud.

1. Lokaalsete muutujate nimed ei tohi ühtida globaalsete- ega püsimälumuutujatega. Küll aga võivad lokaalsete muutujate nimed erinevates funktsiooniplokkides korduda.
2. Globaalsete muutujate nimed võivad ühtida püsimälumuutujate nimedega, eristatavad vastavate eesliidete järgi „GVL“ ja „DS“.

Globaalseid ja lokaalseid muutujaid hoitakse muutmälus, püsimälumuutujaid EEPROM püsimälus. FBD programmeerimiskeskkonna ja teiste tarkvaramoodulite vahel saab infot vahetada kasutades globaalseid muutujaid. Selleks, et paremini kirjeldada kontrolleri erinevate keskkondade ja rakenduste ligipääsu muutujatele, on koostatud joonis 5.2.



Joonis 5.2. Kontrolleri muutujate ja rakenduste omavaheline ühenduvus

Jooniselt 5.2 on näha, et kõik keskkonnad pääsevad ligi globaalselt defineeritud muutujatele. Lokaalsetele ja püsimälumuutujatele pääseb ligi ainult FBD rakendus. Kui püsimälus olevatele muutujatele on vajadus ligi saada mingist muust keskkonnast, tuleb need ümber defineerida või siduda lisaks globaalse muutujaga.

Programmi koodi kirjutamise lihtsustamiseks on otstarbekas teha sisend-/väljundsignaalide tabel.

Tabel 5.1. Sisend- / väljundsignaalide tabel

Signaali kirjeldus	Riistvaraline aadress	Tarkvaraline aadress	Signaali tüüp
Digisisendid (*Digital inputs*)			
Süsteem automaatjuhtimisel *System in „Auto“ mode*	DI1.0	1.ch_0	24V
Süsteem käsijuhtimisel *System in „Manual“ mode*	DI1.1	1.ch_1	24V
Ventilaatori olek töös *Fan status: running*	DI1.2	1.ch_2	24V
Digiväljundid (*Digital outputs*)			
Ventilaatori käivitamise käsk *Fan start command*	DQ2.0	2.ch_0	24V
Analoogsisendid (*Analog inputs*)			
Süsteemi temperatuur *Temperature*	AI3.0	3.ch_0	4…20mA = 0…50°C

Tabel 5.1 annab kogu vajaliku info, mida on tarvis programmis muutujate defineerimiseks. Signaali täpne kirjeldus ja tüüp koos riist- ja tarkvaralise aadressiga. Riistvaraliste aadresside nimede struktuur on koostatud kaheosalisena. Tähtedega tähistatud osa näitab sisendi tüüpi:

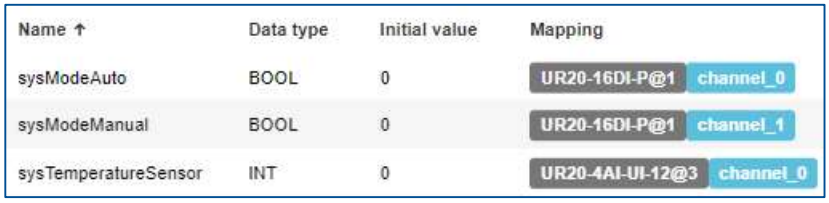
1. DI – digisisend;
2. DQ – digiväljund;
3. AI – analoogsisend.

Riistvaralise aadressi vasakult esimene number näitab signaali mooduli asetust kontrolleri põhimooduli suhtes. Kontrolleri põhimooduli tähis on „0“, kuna põhimoodulil endal puuduvad sisendid ja väljundid, siis esimene sisend saab asuda moodulis „1“ ja signaalid selles moodulis on tähistatud järjest, alustades nullist. Analoogset loogikat on kasutatud kogu süsteemi ulatuses.

Tarkvaraline aadress näitab, kuidas on signaali sisend tähistatud kontrolleri arenduskeskkonnas, mis võimaldab sisendi ära siduda kontrolleri sisese muutujaga.

Signaali tüüp näitab, mis olekus peab sisend olema, et kirjeldus vastaks tõele. Siinjuhul tähistab 24V, et sisend peab olema aktiivne. Sellisel juhul loetakse vastav signaali olek tõeseks. Rikkesignaali sisendid on tehtud selliselt, et 0V vastab signaali kirjeldusele, mis aitab tuvastada lisaks nimetatud rikkele ka toitepinge kadu vastavas ahelas. Töös digitaalse signaaliga rikkeid ei ole kasutusel. Analoogsignaali puhul on näidatud sisendsignaali tüüp ja mõõtevahemik.

Järgnevalt on defineeritud sisenditele ja väljunditele programmi sisemised muutujad vastavalt sisend-/väljundsignaalide tabelile 5.1. Selle jaoks on valitud kontrolleri arenduskeskkonna avalehelt programmeerimise alamleht ja sealt redigeerimise vaade. Kõik kontrolleri sisendid ja väljundid on programmis kasutamiseks seotud globaalsete muutujatega. Näide muutujate nimetamisest on esitatud joonisel 5.3.



Joonis 5.3. Muutujate tähistamine programmis

Jooniselt 5.3 on näha muutuja nimetamisel kasutatud *camelCasing* meetodit.

Initial value veerus on näidatud muutuja vaikeväärtus, mis omistatakse muutujale kontrolleri käivitamisel või programmilaadimisel.

Mapping veerus on näidatud signaali aadress arenduskeskkonnas. Hallis osas on kuvatud kontrolleri mooduli info ja pärast „@“ märki on näidatud mooduli järjekorranumber. Sinisega on tähistatud mooduli kanali number.

Data type veerus on näidatud muutuja andmetüüp. Kontrolleri poolt toetatud muutujate andmetüübid ja nende kirjeldus on esitatud tabelis 5.2.

Tabel 5.2. Muutujate andmetüübid [18] [26]

Andmetüüp	Võimalikud väärtused
BOOL	Kahendarv TRUE või FALSE, mis vastavad väärtustele 1 või 0
BYTE	8 bitine number 0…255
DINT	32 bitine number -2147483648…2147483647
DWORD	32 bitine number 0…4294967295
INT	16 bitine number -32768… 32767
REAL	32 bitine ujukomaarv -3.402823e38…3.402823e38
SINT	8 bitine number -128…127

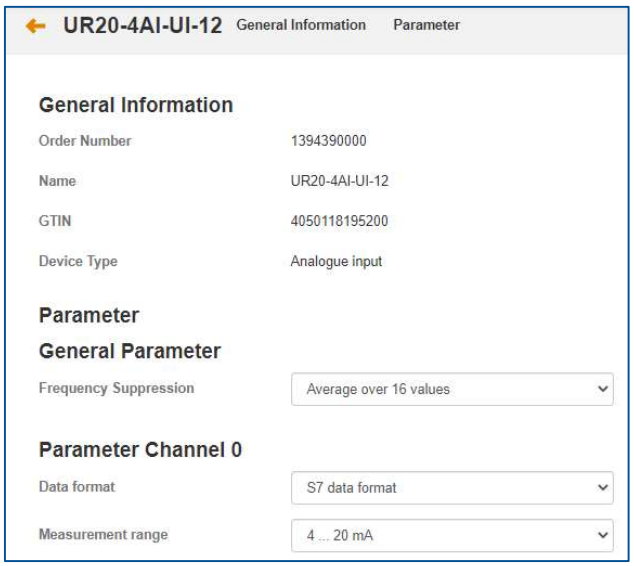
TIME	32 bitine 0ms…49d17h,2m47s29ms (d – päeva, h – tundi, m – minutit, s – sekundit, ms – millisekundit)
UDINT	32 bitine number 0…4294967295
UINT	16 bitine number 0…65535
USINT	8 bitine number 0…255
WORD	16 bitine number 0…65535 (soovitus kasutada selle asemel UINT)

Digisisendite ja -väljundite puhul on kasutusel „BOOL“ tüüpi muutuja, mille väärtus saab olla kas „1“ või „0“. Analoogsisend on defineeritud kontrolleri tarkvaras kui 16 bitine „WORD“. Seda tüüpi muutujat ei ole võimalik kasutada programmis matemaatiliste tehete tegemiseks. Lähtudes järgmises peatükis kirjeldatust, võib etteruttavalt määrata analoogsisendi muutuja andmetüübiks „INT“. Töös on kasutatud muutujaid andmetüübiga „BOOL“, „INT“, „REAL“ ja “TIME“.

Järgnevas peatükis on kirjeldatud kontrolleri analoogsignaali kasutamise jaoks olulisi seadeid ja arvutusi.

5.2. Analoogsignaali töötlus

Kontroller teisendab 4…20mA analoog sisendsignaali väärtust kasutades analoog-digitaalmuundurit. Seade kasutab analoogsignaali skaleerimiseks teise tootja poolt väljatöötatud andmevormingu loogikat, millele viidatakse kui „S7 *data format*“. Seda ei ole mainitud ega kirjeldatud kontrolleri juhendis, spikkermenüüs (*Help menu*) ega ka mooduli enda spetsifikatsioonis. Kui analoogmoodul ühendada kontrolleriga ning avada mooduli konfigureerimise leht on valida kahe andmevormingu (*Data format*) vahel, milleks on „S7“ või „S5“. Kontrolleri puuduliku dokumentatsiooni tõttu ei ole ilma eelneva kogemuseta tööstusautomaatika valdkonnas võimalik tuvastada, mis selle seadistuse valiku all on mõeldud. Siemensi kontrollerite seeria tähistus on S7, siis järeldati, et siinjuhul viidatakse Siemens S7 andmevormingule analoogsignaali töötluses (S5 on vanem generatsioon). Analoogmooduli seadistus on esitatud joonisel 5.4.



Joonis 5.4. Analoogmooduli seadistus

Mõõtepiirkonna parameetriks (*Measurment range*) määrati temperatuurianduri sisendiks kontrolleris 4…20mA. Mooduli kõikidele sisenditele saab eraldi seadistada signaali andmevormingu ja mõõtepiirkonna.

Analoogsignaalide sisendmoodulil on üks üldine muudetav parameeter, sageduse summutamine (*frequency suppression*), mis on mõeldud signaali müra vähendamiseks. Valikus on järgnevad häiringute vähendamise meetoid:

1. Desaktiveeritud – häiringute filtreerimine on väljalülitatud;
2. 50Hz ja 60Hz – vahelduvpingest (üldjuhul elektrivõrgust) tekitatud häiringute vähendamine;
3. *Average over 16 values* – mõõdetud 16 väärtuse keskmistamine. Vähendab müra mõju töödeldud signaalile, kuid vähendab ka diskreetimissagedust (*sampling rate*) [31].
4. 2Hz *low pass* – kasutatud meetodi kohta kirjeldus puudub. Katsetamisel tuvastati sarnane käitumine nagu eelmise meetodi puhul.

Automaatjuhtimissüsteemi katsestend on kompakte seade, kus andur asub kontrolleri vahetusläheduses ja häiringute mõju süsteemile on hinnatud väikeseks. Kuna häirete vähendamise meetodite rakendamine on praktikas levinud, on otsustatud seda rakendada ka magistritöös. Kirjeldatud lahenduses on müra vähendamiseks kasutatud 16 mõõteväärtuse keskmistamist. Selline müra filtreerimise meetod on suhteliselt universaalne ja sobib kasutamiseks paljude erinevate andurite ja protsesside puhul. Temperatuuri näidu uuendamise sagedus (näiteks 0,1s või 1s) ei oma suurt rolli kirjeldatud süsteemi juhtimise seisukohast, kuna temperatuuri muutus on oma iseloomult aeglane protsess.

Siemens S7 seeria kontrolleri programmeerimise juhendi info analoogsignaali andmevormingu kohta on esitatud tabelis 5.3 [32].

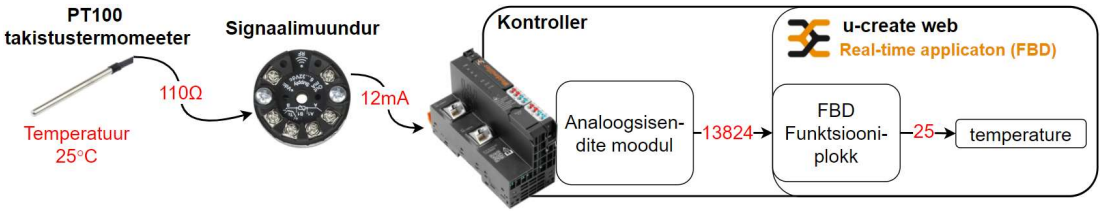
Tabel 5.3. Analoogsignaali töötlemine vastavalt Siemens S7 loogikale [32]

Elektrilise signaali väärtus	Kontrolleri signaali väärtus	Järeldus
0mA ≤ sig[1] < 1,185mA	-32768…-4865	Sisendsignaali väärtus lubatust madalam, alatäidetud (*underflow*)
1,185mA ≤ sig[1] < 4mA	-4864…-1	Signaal alla mõõtepiirkonna, alla normi (*underange*)
4mA ≤ sig[1] ≤ 20mA	0…27648	Normaaltöö vahemik
20mA < sig[1] ≤ 22,81mA	27649…32511	Signaal üle mõõtepiirkonna, üle normi (*overange*)
22,81mA < sig[1] ≤ 22.96mA	32512…32767	Sisendsignaali väärtus lubatust kõrgem, ületäidetud (*overflow*)

1) signaal

Vastavalt tabelile 5.3 on näha, et analoogsignaali väärtuse normaaltöö vahemik on 0…27648. Sellest tulenevalt on analoogsisendi muutuja andmetüübiks määratud juba eelnevalt „INT“.

Analoogsignaali genereerimise paremaks kirjeldamiseks on esitatud joonis 5.5.



Joonis 5.5. Analoogsignaali genereerimine

Joonisel 5.5 on näha PT100 andur, mille väljund 25°C juures on umbes 110Ω [33]. Signaalimuunduri sisend on seadistatud PT100 andurile, mõõteskaala on vahemikus 0…50°C ja väljundi tüübiks on 4…20mA. Sisendi 110Ω korral on signaalimuunduri väljundiks 12mA. Signaalimuundur on ühendatud kontrolleri analoogsisendite mooduliga, mille seadistused on näidatud joonisel 5.4. Järgnevalt tuleb luua funktsiooniplokk, mis teisendaks analoogsignaali temperauuri väärtuseks ja omistaks selle globaalsele muutujale.

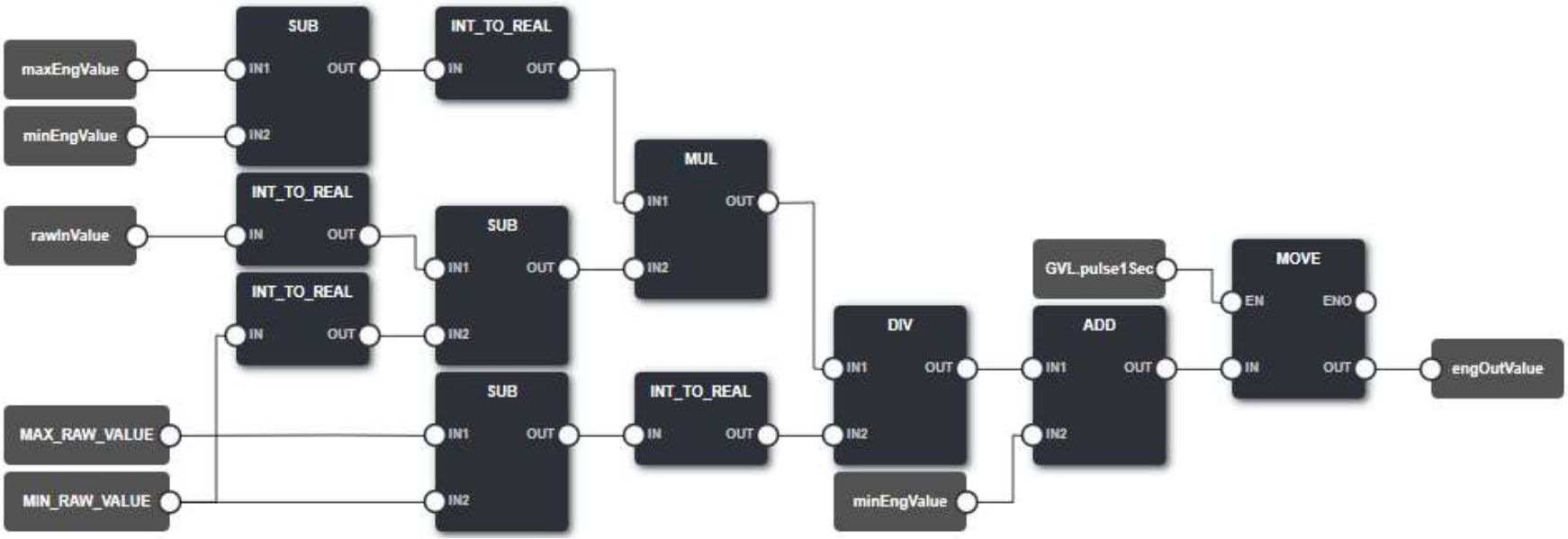
Funktsiooniplokk, mis teisendab analoogsisendist saadud anduri info programmis lihtsamini kasutatavaks temperatuuri väärtuseks, vajab järgnevaid sisendeid ja väljundeid:

1. Temperatuurianduri signaal, mis seotakse ära kontrolleri analoogsisendi vastava globaalse muutujaga:
    a. Funktsiooniploki muutujale „rawInValue“ omistatakse süsteemi temperatuurianduri väärtus analoogsignaalide moodulist „sysTemperatureSensor“.
2. Töötlemata signaali miinimum ja maksimum väärtused, mis on defineeritud lokaalsete konstant muutujatena:
    a. Funktsiooniploki muutujale „MIN_RAW_VALUE“ omistatakse konstant 0;
    b. Funktsiooniploki muutujale „MAX_RAW_VALUE“ omistatakse konstant 27648.
3. Mõõteskaala miinimum ja maksimum väärtused, mis on seadistatud temperatuurianduri signaalimuunduris, on toodud ka funktsiooniploki sisse. Programmis defineeritud muutujad on järgnevad:
    a. Funktsiooniploki muutujale „minEngValue“ omistatakse muutuja „minEngValueTemperature“ väärtus, milleks on 0°C;
    b. Funktsiooniploki muutujale „maxEngValue“ omistatakse muutuja „maxEngValueTemperature“ väärtus, milleks on 50°C.
4. Väljundsignaal °C ühikutes, mis seotakse ära globaalse muutujaga:
    a. Globaalsele muutujale „temperature“ omistatakse funktsiooniploki muutuja „engOutValue“.

FBD programmi ülesehitus graafilises keskkonnas näeb ette struktuuri, kus sisendid on vasakul ja väljund paremal. Programmi algoritmi aluseks on valem 5.1 [34]:

$$engOutValue = \frac{(maxEngValue - minEngValue) \cdot (rawInValue - \quad\_RAW\_VALUE)}{MAX\_RAW\_VALUE - MIN\_RAW\_VALU} + minEngValue \qquad (5.1)$$

Koostatud programmilõik on esitatud joonisel 5.6, kus on näha valemis 5.1 esitatud algoritm FBD programmeerimiskeeles.



Joonis 5.6. Analoogsignaali töötluse programmilõik kontrolleris

FBD programmi graafiline arenduskeskkond lubab kasutada erinevaid valmisolevaid elemente teegist (*library*). Järgnevalt on kirjeldatud joonisel 5.6 esitatud programmis kasutatud aritmeetilisi tehteid ja nende vastavaid elemente teegist:

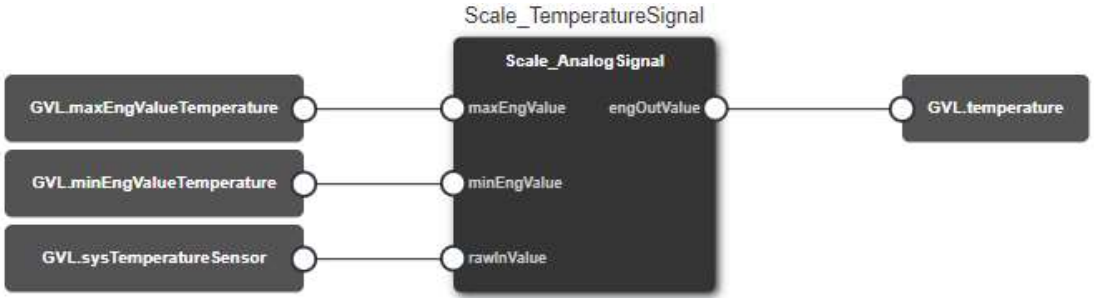
1. SUB – (*subtraction*) lahutamistehe;
2. MUL – (*multiplication*) korrutustehe;
3. DIV – (*division*) jagamistehe;
4. ADD – (*addition)* liitmistehe.

Tehteid saab teha ainult ühesuguste andmetüüpidega. Joonisel 5.6 esitatud programmilõigus kasutatud muutujad ja nende andmetüübid on esitatud tabelis 5.4.

Tabel 5.4. Analoogsisendi teisendamiseks kasutatud muutujad

Nimi	**Funktsioon**	**Tüüp**
MAX_RAW_VALUE	VAR	Täisarv (*Integer*)
MIN_RAW_VALUE	VAR	Täisarv (*Integer*)
maxEngValue	VAR_INPUT	Täisarv (*Integer*)
minEngValue	VAR_INPUT	Täisarv (*Integer*)
rawInValue	VAR_INPUT	Täisarv (*Integer*)
engOutValue	VAR_OUTPUT	Ujukomaarv (*Real*)

Tabelist 5.4 on näha, et kasutusel on nii täisarvud kui ujukomaarv. Erinevat tüüpi muutujatega tehete tegemiseks on vaja need teisendada ühele kujule. Täisarvud tuleb teisendada ujukoma arvuks. Selleks kasutatakse programmis teisendusfunktsiooni „INT_TO_REAL“. Tabelis 5.4 on veerus „Funktsioon“ näidatud muutuja otstarve. „VAR“ tähistab lokaalset muutujat, mis on kasutusel ainult funktsiooniploki sees, väljastpoolt ei ole võimalik seda lugeda ega sinna kirjutada. „VAR_INPUT“ ja „VAR_OUTPUT“ defineerivad vastava muutuja funktsiooniploki sisendi või väljundina. Joonisel 5.7 on esitatud analoogsignaali teisenduse funktsiooniplokk „Scale_AnalogSignal“ esialgne versioon, mille sees on joonisel 5.6 näidatud programmilõik.

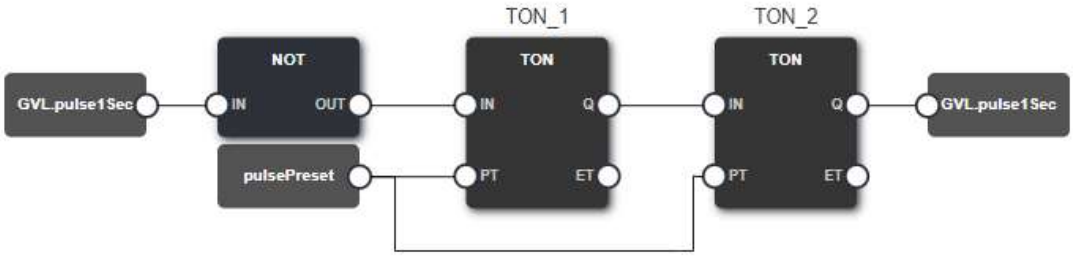


Joonis 5.7. Analoogsignaali teisenduse funktsiooniploki esialgne versioon

Joonisel 5.7 esitatud funktsiooniploki sisendid ja väljundid on määratud vastavalt tabelis 5.4 näidatud funktsioonile. Tegemist on globaalsete muutujatega, millele tarkvara lisab ise „GVL.“ eesliite. Funktsiooniploki sisenditeks on temperatuurianduri signaal kontrolleri analoogmoodulist ja selle mõõtevahemik ning väljundiks on temperatuuri väärtus, mille ühikuks on °C.

Funktsiooniplokina esitatud programmilõik on universaalne ja seda saab kopeerida ning kasutada teiste analoogsignaalide töötlemiseks. Ploki sisendite ja väljunditena on välja toodud kõik olulised seadeparameetrid, mida saab siduda muutujatega ja see lihtsustab ning kiirendab uute analoogsignaalide kasutuselevõttu programmis. Edaspidi on vajadus temperatuuri

kasutada Node-RED ja visualiseerimiskeskkonnas ning seetõttu on see defineeritud globaalselt.
Temperatuurianduri signaali väärtus muutub vastavalt analoogmooduli töötlemiskiirusele ja kontrolleri programmitsüklile. Programmi tsükli läbimise aeg on 1ms ja analoogsignaali mooduli andmelehes on märgitud maksimaalseks signaali töötlemise ajaks 50ms [21]. Temperatuuri näit võib seega uueneda programmis umbes kakskümmend (või rohkem) korda sekundis. Kasutatud temperatuuri signaalimuunduril on signaali töötlemise ajaks märgitud 600ms [35]. Seega reaalne temperatuuri näit jõuab andurist kontrollerisse umbes iga 650ms järel, sagedasemal näidu uuenemisel ei ole vajadust. Kontrolleri programmeerimiskeskkonna vaates uuendatakse näitu umbes neli korda sekundis. Vahepealne näidu väärtuse muutus võib olla tingitud analoog-digitaalkonverteri mürast või muude häiringute mõjust mõõteahelale [31]. Arvestades protsessi iseloomu ja tööülesannet, ei ole nii sagedane näidu uuendamine vajalik ning raskendatud on eelkõige näidu lugemine inimese poolt.
Järgnevalt on programmeeritud lahendus näidu uuendamise sageduse vähendamiseks. Kasutajal ei ole ligipääsu kontrolleri süsteemi kellale (arenduskeskkonna spikkermenüü seda ei kirjelda), mida oleks võimalik kasutada perioodilise näidu uuendamise eesmärgil. Selleks loodi taimer (*timer*), mis edastab impulsi umbes ühe sekundilise intervalliga. Tegelik taimeri intervall sõltub kontrolleri tsüklite läbimise kiirusest ja seega ei saa väita, et see on täpselt üks sekund. Kasutusotsarvet hinnates ei ole taimeri täpsus siinjuhul eriti oluline. Eesmärk on vähendada näidu uuendamise sagedust, et kuvatud väärtus oleks paremini loetav inimsilmale. Loodud taimeri programmilõik on esitatud joonisel 5.8.



Joonis 5.8. Intervalltaimer

Intervalltaimeri jaoks on kasutatud kahte viitega sisse lülituvat taimeri plokki jadamisi (TON_1 ja TON_2). Plokk koosneb järgmistest sisenditest ja väljunditest, nende andmetüüp on esitatud sulgudes:

1. IN – sisendsignaal mis käivitab taimeri (BOOL);
2. PT – aegväärtus milleni loendatakse (TIME);
3. ET – taimeri hetkeväärtus (TIME);
4. Q – taimeri väljund (BOOL).

Joonisel 5.8 näidatud plokil on kasutatud ühte globaalset muutujat („GVL.pulse1Sec“, BOOL), mis on ühtlasi ka ploki sisendsignaaliks. Muutuja vaikeväärtuseks kontrolleri käivitushetkel on „0“, seega on tarvis sisend inverteerida. Teine võimalus on määrata muutuja vaikimisi väärtus „1“ sel juhul pole inverteerimine vajalik. Funktsiooniploki lokaalne muutuja „pulsePreset“ on andmetüübiga TIME ja määrab ära aja, milleni loendatakse.
Sisendsignaali korral alustab taimer loendamist ettemääratud aegväärtuseni, mille saavutamisel aktiveerub taimeri „TON_1“ väljund. Järgmine plokk täidab sama ülesannet ja

TON_2 taimeri väljund omistab muutujale „pulse1Sec“ väärtuse „1“. See omakorda muudab selle sama intervalltaimeri sisendi väärtuseks „1“, mis inverteeritakse ja TON_1 sisendilt kaob signaal, taimer läheb tagasi algolekusse ja TON_1 väljundsignaal kaob. Sama toimub TON_2 taimeriga, mille väljund omistab „pulse1Sec“ muutujale väärtuse „0“ ja protsess kordub. Taimerid TON_1 ja TON_2 vajavad eelseadistusena aegväärtust, milleni loendatakse. Kasutatud on kahte taimerit jadamisi, seega ühe sekundilise perioodi saamiseks on vajalik mõlemale taimerile määrata 0,5 sekundiline aegväärtus. Selline programmilõik annab pulseeriva olekuga muutuja, mille olek („0“ või „1“) vaheldub perioodiliselt umbes ühe sekundilise intervalliga. Funktsiooniplokk ise ei vaja väljundit, kuna muutuja „pulse1Sec“ on defineeritud globaalselt.

Temperatuurinäidu uuendamise sageduse vähendamiseks on kasutatud FBD arenduskeskkonna teegist elementi „MOVE“ (vt joonis 5.6.), millel on järgnevad sisendid ja väljundid:

1. EN (*enable*) – kui tõene, siis edastab sisendi väärtuse väljundisse (BOOL);
2. ENO (*enable output*) – imiteerib sisendi EN väärtust (BOOL);
3. IN – ploki sisendväärtus (sobivad kõik andmetüübid);
4. OUT – ploki väljundväärtus (andmetüüp sama mis sisendis „IN“).

MOVE ploki töö käsk (EN) tuleb eelnevalt loodud intervalltaimeri muutujast „pulse1Sec“, mis käivitab plokki umbes ühesekundilise intervalliga. Ploki töö tulemusena on vähendatud temperatuurianduri näidu uuendamise sagedust kontrolleris.

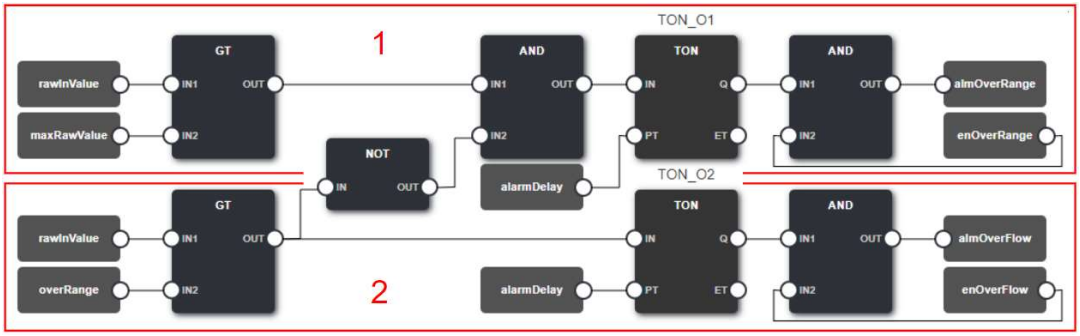
Järgnevas peatükis on kirjeldatud analoogsignaaliga seotud riistavaralised alarmide loomine.

5.3. Analoogsignaali riistvaralised alarmid

Eelnevalt kirjeldatud peatükis käsitletud analoogsignaali töötlemisloogika tabelit 5.3 analüüsides, saab igale signaalile programmeerida põhilised riistvaralised alarmid:

1. Sisend ületäitunud;
2. Sisend alatäitunud;
3. Sisend üle mõõtepiirkonna;
4. Sisend alla mõõtepiirkonna.

Nimetatud alarmid lisatakse olemasolevale funktsiooniplokile „Scale_AnalogSignal“. Normaaltöö olukorrast kõrgemate riistvaraliste alarmide genereerimise programmilõik on esitatud joonisel 5.9.



Joonis 5.9. Normaaltöö olukorrast kõrgemate riistvaraliste alarmide genereerimine

Joonisel 5.9 näidatud programmilõigus on näidatud kahe alarmi genereerimine:

1. Sisendsignaal üle mõõtepiirkonna (*overrange*);
2. Sisendsignaal ületäitunud (*overflow*).

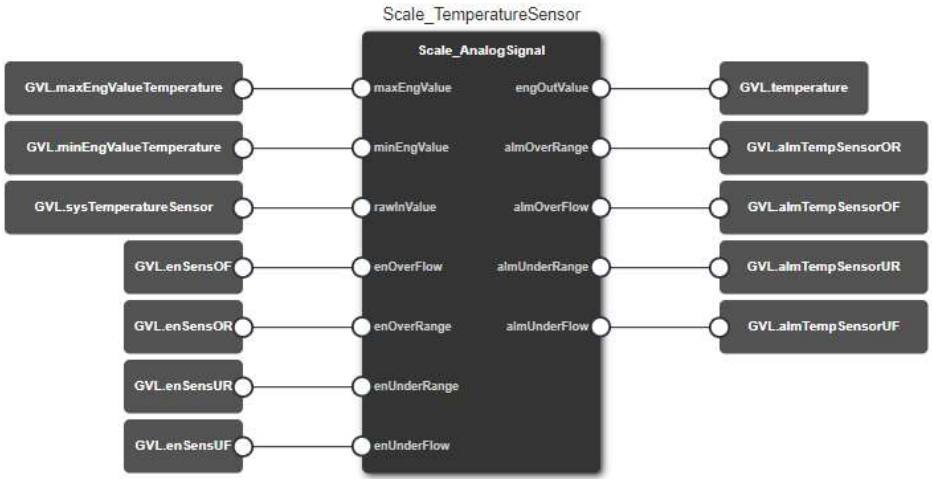
Programmi täitmisel võrreldakse plokis „GT“ (*greater than*), kas sisendväärtus („rawInValue“) on suurem kui normaaltöö maksimaalväärtus („maxRawValue“). Järgnevalt kontrollitakse tingimust, kas sisendsignaal on madalam kui mõõtepiirkonna maksimum („overRange“). Kui nimetatud tingimused on tõesed, on alarmi, sisendsignaal üle mõõtepiirkonna, tingimused täidetud.

Testimisel selgus, et tehnoloogilise protsessi ja seadmete eripärast võis esineda lühiajalist sisendsignaali kõikumist suurtes piirides. Sellised olukorrad edastavad asjatuid rikketeateid ning need olukorrad tuleb välistada, näiteks filtreerimise teel. Töös on kasutatud anomaaliate filtreerimiseks ajalist viidet. Programmi on lisatud viite jaoks taimer, mis edastab alarmi juhul, kui tingimused on tõesed olnud katkematult vähemalt viis sekundit. Sellise ajani jõuti testimise käigus, kui anduri ahel ajutiselt katkestada, püsisid anomaaliad mõõtesisendile umbes kolm sekundit. Katkestuse ja taas ühendamise korral muutus signaal mõlema äärmuse piirides. Programmiliselt on välistatud mõlema alarmi samaaegne olek. Kui on tõene ületäitumise signaal, siis ei genereerita alarmi „sisendsignaal üle mõõtepiirkonna“.

Joonisel 5.9 kujutatud koodi teises reas on esitatud sisendi ületäitumise alarmi tingimused. Kontrollitakse, kas sisendsignaali väärtus on kõrgem kui ületäitumise alarmi seadeväärtus, kui see tingimus on täidetud, rakendub viiesekundiline taimer ja seejärel edastatakse alarm.

Alarmide aktiveerimist ja desaktiveerimist saab juhtida läbi kasutajaliidese, selleks on ettenähtud muutujad „enOverRange“ ja „enOverFlow“. Alarmi vaigistamise muutujatel on seadistatud vaikimisi algväärtuseks „1“ ja iga programmi laadimise ning toitekatkestusega omistatakse neile vaikeväärtused. Alarmi desaktiveerimine on tehnoloogilise protsessi seisukohast ajutine tegevus (näiteks anduri rike) ja muutuja väärtuse püsimällu kirjutamist siinjuhul ei ole rakendatud.

Analoogselt on lahendatud ka alarmid „sisend alla mõõtepiirkonna“ ja „sisend alatäitunud“. Lõplik funktsiooniplokk koos sisendite väljunditega on esitatud joonisel 5.10.



Joonis 5.10. Funktsiooniplokk „Scale_AnalogSignal“

Muutujate nimetuste lisamisel on võetud kasutusele lühendid, et kirjapilt oleks kompaktsem (OR – *overrange;* OF – *overflow*; UR – *underrange;* UF – *underflow*). Eesliide „alm“ (*alarm*) aitab paremini muutujaid grupeerida otstarbe järgi, mis lihtsustab nende leidmist muutujate loetelust.

Kõik funktsiooniplokiga seotud muutujad on defineeritud globaalselt, mis võimaldab nendele ligi pääseda ka teistest funktsiooniplokkidest, Node-RED rakendusest ja visualiseerimiskeskkonnast. Loodud funktsiooniplokk on universaalne kõikidele analoogsisenditele. Lisas VII on esitatud „Scale_AnalogSignal“ täielik programm ja muutujad.

Järgmises peatükis on kirjeldatud protsessi juhtimiseks vajalike tehnoloogiliste alarmide programmilõikude loomist.

5.4. Analoogsignaali tehnoloogilised alarmid

Eelnevalt on kirjeldatud analoogsignaali alarme, mis iseloomustavad riistvaralise komponendi või elektriahela riketest tingitud probleeme. Tehnoloogilise protsessi juhtimiseks ja ebanormaalsetele olukordadele reageerimiseks on tarvis seadistada analoogsignaalile alarmväärtused. Praktikas on levinud kaheastmeline alarmi süsteem, mis tähendab, et enne kriitilise väärtuse saavutamist on ka hoiatava sisuga teavitus. Töös on alarmi tähistes kasutatud lõppliiteid ja nende lühendeid (esitatud allpool sulgudes), mis aitavad kasutajal paremini programmi lugeda:

1. *Low* (L) – madal hoiatus;
2. *LowLow* (LL) – madal kriitiline;
3. *High* (H) – kõrge hoiatus;
4. *HighHigh* (HH) – kõrge kriitiline.

Sõltuvalt tehnoloogilisest protsessist võib olla vajadus kasutada ka teistsuguste astmetega süsteeme.

Analoogsignaali põhjal tekitatud alarm saab aktiivsest olekust tagastuda kas automaatselt (isetagastuv), või kasutaja sekkumisega (käsitsi tagastus). Automaatset tagastust kasutatakse olukordades, kus alarmi rakendumine ja seejärel normaalolukorra taastumine ei vaja inimese sekkumist. Kui tehnoloogilises protsessis esinenud häire olukord vajab inimese sekkumist, siis on vaja alarmi tagastus teha käsitsi. Töös kirjeldatud tehnoloogilise protsessi jaoks sobib alarmide automaatne tagastus. Kui sisendsignaali väärtus muutub alarmi seadeväärtuse ümber sagedasti, saadakse iga kord alarmväärtuse ületamisel veateade. Isetagastuv süsteem vajab alarmi tagastumiseks hüstereesi. Hüstereesiga tekitatava tundetuspiirkonna laiust võib määrata kui:

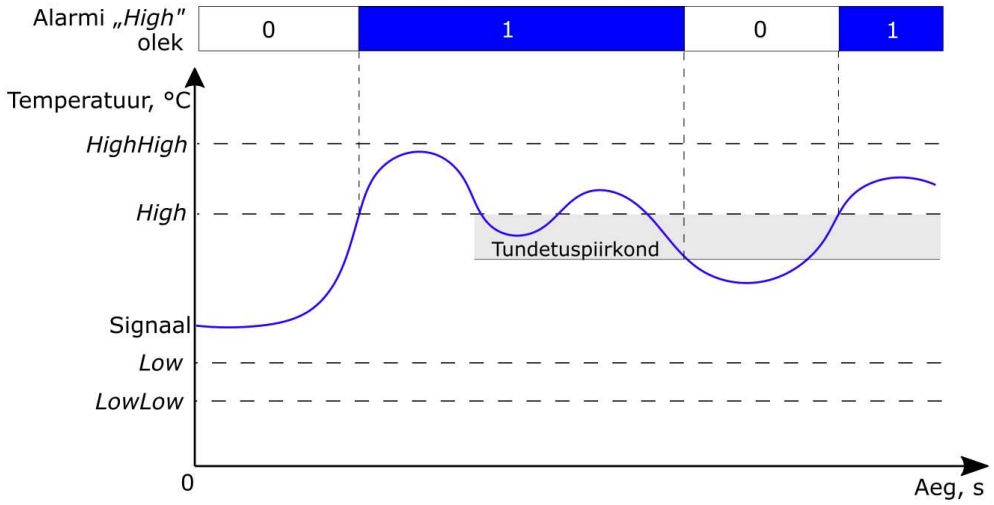
1. Protsenti seadeväärtusest;
2. Fikseeritud väärtusena.

Seadeväärtuse protsendi kaudu määratud alarmi hüstereesi kasutamine on programmiliselt mugav rakendada (hüstereesi suurus määratakse automaatselt vastavalt ettenähtud protsendile), kuid ei ole väga paindlik. Kui kasutajal on vaja täpselt fikseerida tundetuspiirkonna

laiust (näiteks 0,1bar, kui anduri mõõtevahemik on 0…6bar), siis on seda protsendi kaudu tülikas arvutada.

Fikseeritud suurusega seadeväärtus ei saa olla igale alarmile ühesugune. Näiteks on rõhuanduri mõõtevahemikuks 0…6bar, aga veetaseme nivooanduril 0…1000mm. Kahe nii erineva mõõteskaalaga anduri puhul ei ole võimalik rakendada ühesugust fikseeritud laiusega tundetuspiirkonda. Igale analoogalarmile on vaja eraldi seadistatavat hüstereesi, mille kasutaja peab ise käsitsi määrama. Selline lahendus on universaalsem ja on kasutatud ka lõputöös.

Alarmi tasemete ja hüstereesi kirjeldamiseks on koostatud joonis 5.11.

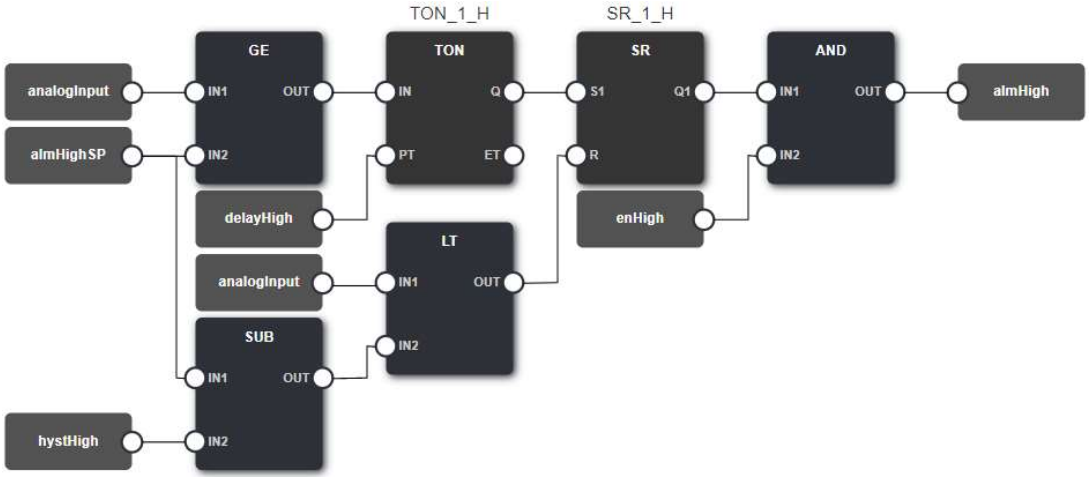


Joonis 5.11. Alarmi tasemed ja hüsterees

Joonisel 5.11 on näidatud alarmi oleku sõltuvus sisendsignaali muutumisest. Võimalikke alarmi ajalisi viiteid ei ole joonisel kujutatud. Kui signaal tõuseb üle alarmi seadeväärtuse (*High*) genereeritakse programmi poolt alarm. Kui signaal langeb alla seadeväärtuse, siis alarm jääb aktiivsesse olekusse, kuni see püsib tundetuspiirkonnas, mis on seadeväärtusest hüstereesi võrra väiksem. Sellist meetodit on rakendatud kõikide analoogalarmide tagastusel.

Reaalses tehnoloogilises süsteemis tuleb arvestada ka signaalihäiringute ja anomaaliatega, mis võivad mõõteahelas tekkida. Selliseid probleeme aitab vähendada alarmi viite tekitamine taimeriga.

Eelnevast infost lähtuvalt on koostatud funktsiooniplokk kaheastmelisele analoogalarmile. Joonisel 5.12 on esitatud hoiatava kõrge alarmi genereerimise programmilõik.



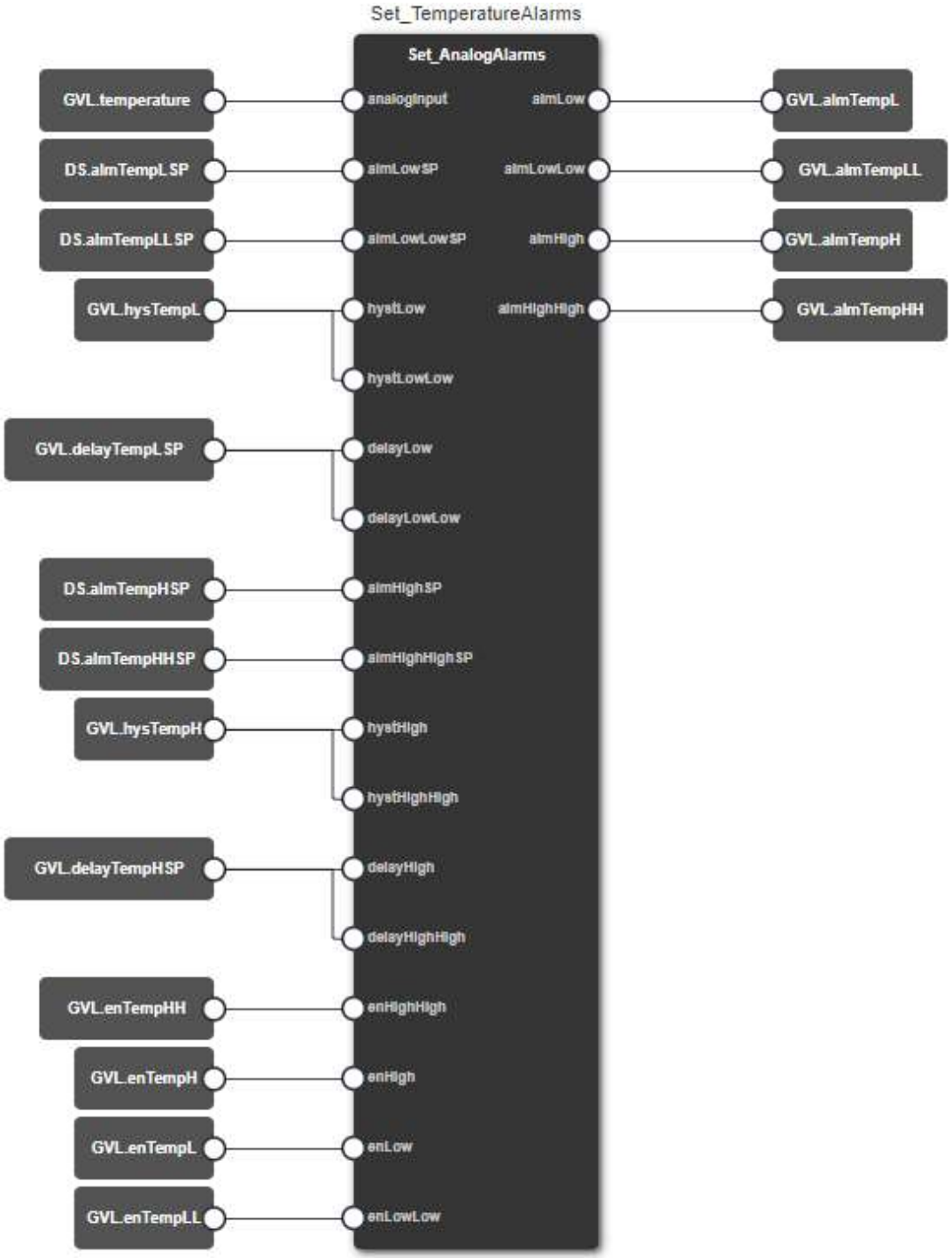
Joonis 5.12. Hoiatava kõrge alarmi genereerimise programmilõik

Joonisel 5.12 on näha järgmised sisendid ja väljundid koos andmetüübiga:

1. analogInput – sisend, eelnevalt töödeldud temperatuuri analoogsignaali väärtus (REAL);
2. almHighSP – sisend, hoiatava kõrge alarmi rakenduse seadeväärtus (REAL);
3. hystHigh – sisend, hüstereesi väärtus hoiatava kõrge alarmi tagastusele (REAL);
4. delayHigh – sisend, alarmi rakendumise viide (TIME);
5. enHigh – sisend, võimaldab alarmi desaktiveerida (BOOL);
6. almHigh – väljund, genereeritud alarmi oleku kahendväärtus (BOOL).

Esmalt kontrollitakse plokis „GE“ (*greater or equal*), kas sisendsignaali „analogInput“ väärtus ületab alarmi seadeväärtust „almHighSP“. Järgnevalt rakendatakse alarmi genereerimiseks ajaline viide plokis „TON_1_H“, mis on seadistatud kolme sekundi peale. Signaal peab katkematult olema üle alarmi seadeväärtuse vähemalt kolm sekundit, seejärel genereeritakse alarm „almHigh“. Alarm tagastub, kui sisendsignaal langeb allapoole alarmi seadeväärtust hüstereesi „hystHigh“ võrra. Võrdluseks on plokk „LT“ (*less than*). Alarmi lülitamise jaoks on kasutatud funktsiooniplokki „RS“ (*Reset Set*). Plokk seab väljundi „Q1“ väärtuseks „1“, kui sisend „S1“ on aktiivne (piisab sisendi hetkelisest aktiveerimisest) ning ploki väljundi vaigistamiseks on vaja aktiveerida sisend „R“ (piisab sisendi hetkelisest aktiveerimisest). Kui mõlemad sisendid „S1“ ja „R“ on aktiivsed, siis on prioriteetne „R“ ja väljund desaktiveeritakse. Alarmi aktiveerimist ja desaktiveerimist saab juhtida läbi kasutajaliidese. Teised alarmi tasemed on analoogselt ülesehitatud, täielik funktsiooniploki sisene programmilõik on esitatud lisas VIII.

Funktsiooniplokk „Set_AnalogAlarms“ koos kasutatud sisendite ja väljunditega on esitatud joonisel 5.13.



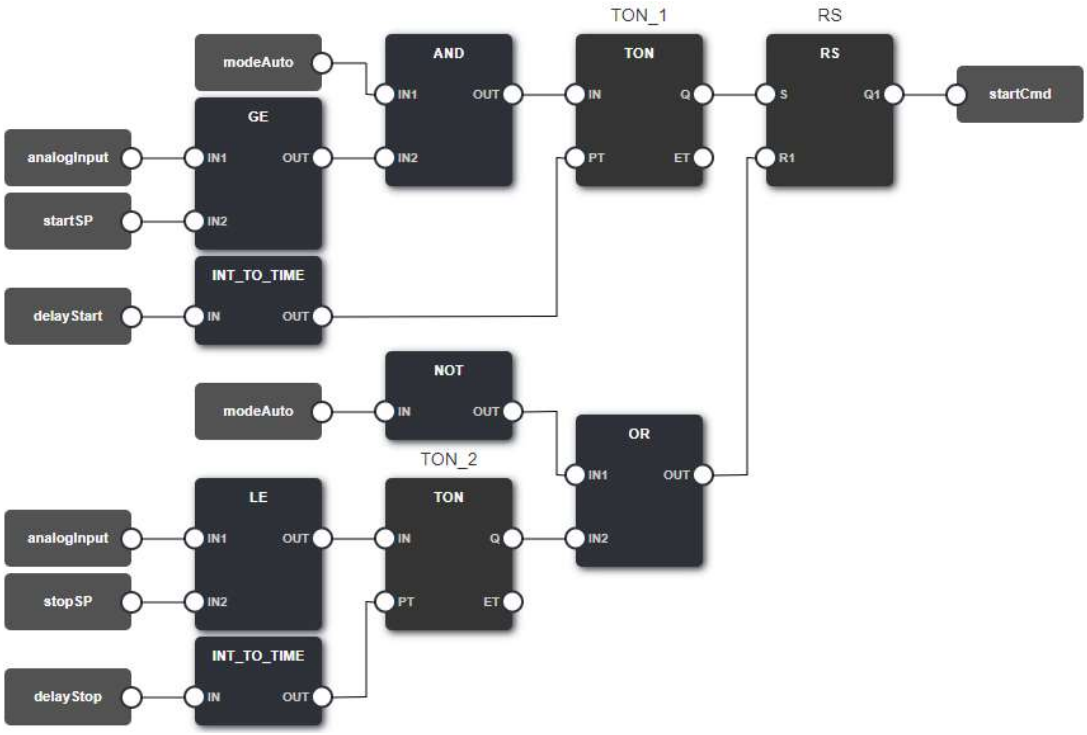
Joonis 5.13. Analoogalarmide genereerimise funktsiooniplokk

Funktsiooniplokk joonisel 5.13 on loodud universaalne kaheastmelise alarmiedastuse jaoks. Töös ei ole ploki funktsionaalsust vaja täielikult rakendada ja kasutatud on ühte hüstereesi väärtust kõrgete alarmide ja teist madalate jaoks. Analoogselt on toimitud ka alarmide viidetega.

Järgmises peatükis on kirjeldatud ventilaatori juhtimise põhimõtteid ja selle jaoks loodud programmi.

5.5. Ventilaatori juhtimine

Vastavalt koostatud lähteülesandele, peab automaatjuhtimissüsteemi katsestendis olev ventilaator tööle hakkama, kui temperatuur ületab ettenähtud seadeväärtust. Väljalülitamiseks võib kasutada teist seadeväärtust, protsenti käivitamise seadeväärtusest, ajalist viidet või hüstereesi. Töös on kasutatud seiskamiseks eraldi seadeväärtust, mis on kasutajale intuitiivsem. Lisaks on tekitatud võimalus kasutada käivituseks ja seiskamiseks ajalist viidet. Ventilaatori juhtimise programmilõik on esitatud joonisel 5.14.



Joonis 5.14. Ventilaatori juhtimise programmilõik

Kui analoogsisend („analogInput“), siinjuhul temperatuur, on võrdne või kõrgem kui käivitamise seadeväärtus („startSP“) ja süsteem on automaatjuhtimisel („modeAuto“), rakendub ajaline viide plokis „TON_1“ ning ventilaator käivitatakse läbi ploki „RS“ aktiveerimise. Kui sisendi väärtus („analogInput“) langeb alla seiskamise seadeväärtuse („stopSP“), siis ploki „TON_2“ ajalise viite möödudes desaktiveeritakse ploki „RS“ väljund ja ventilaator seiskub. Kui süsteem ei ole automaatjuhtimisele („modeAuto“ väärtus on „0“), siis ventilaator seiskub. Programmi väljund „startCmd“ on seotud kontrolleri riistvaralise väljundiga, mis juhib ventilaatori tööd. Funktsiooniplokk „start_motor“ on esitatud lisas VI. Funktsiooniploki lokaalsed muutujad on esitatud lisas IX.

Eelnevalt on käsitletud programmiosad võimaldavad tehnoloogilise protsessi põhiülesande täitmist. Järgnevas peatükis on kirjeldatud Node-RED keskkonna poolt täidetavat kõrvalülesannet, milleks on alarmi edastus e-kirja teel.

6. Programmeerimine Node-RED keskkonnas

Tööülesandes esitatud kirjelduse kohaselt on vaja edastada süsteemi häired e-kirja teel. Selle lahendamiseks on töös kasutatud kontrollerisse integreeritud Node-RED arenduskeskkonda.

Node-RED pakub ettevõttele võimalusi liikuda lähemale Industry 4.0 põhimõtete integreerimisele oma lahendustes. Industry 4.0 näeb ette seadmete suuremat omavahelist ühenduvust ja andmevahetust, mis võimaldab tehnoloogilist protsessi paremini analüüsida ja probleeme ennetada [36]. Üks võimalus IoT seadmetest andmete kättesaamiseks on ühendada need Ethernet TCP/IP andmesidevõrku. Automaatikakontrollerite poolt toetatavad andmesideprotokollid võivad erineda tihtipeale IoT seadmete pakutavast ja see tekitab nende vahel ühilduvuse probleemi. Kontrolleris olevat Node-RED keskkonda saab kasutada lüüsina (*gateway*), mis võimaldab andmevahetust erinevate süsteemide vahel.

Järgnevas peatükis on antud ülevaade kasutatud Node-RED keskkonnast.

6.1. Ülevaade Node-RED keskkonnast

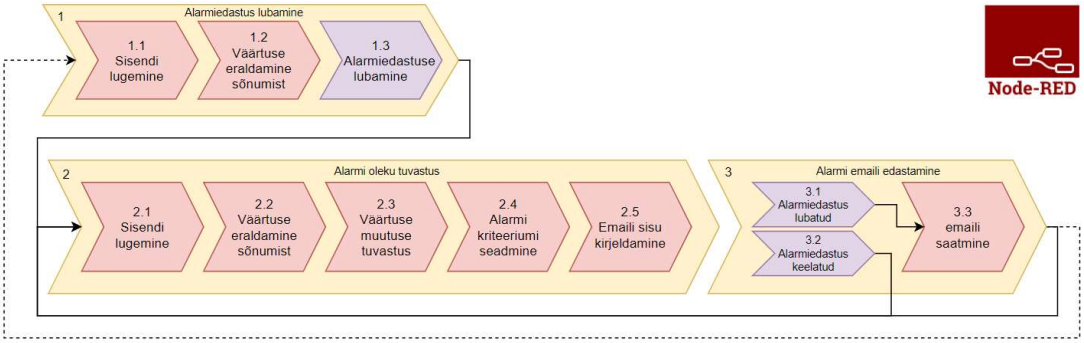
Node-RED on voopõhine sündmustest juhitud programmeerimistööriist, mis on arendatud Node.js peal [37]. Node.js on JavaScript käitusajaobjekt (*runtime*), mis on arendatud Chrome'i V8 JavaScript mootorile [9].

Node-RED keskkonnas kasutatud visuaalne elementide seoste kujutamine on sarnane graafilistele programmeerimiskeeltele, mis vastavad standardile IEC 61131-3. Sarnasus lihtsustab süsteemiintegraatorile, kes on kursis tööstusautomaatika kontrollerites kasutatavate graafiliste programmeerimiskeeltega, kuid võõrad JavaScripti ja Node.js keelega, ülemineku ühest keskkonnast teise. Node-RED võimaldab ühendada ääreseadmeid, mis kasutavad erinevaid andmesideprotokolle. Tööstuses kasutatavad levinuimad automaatikakontrollerid toetavad üldiselt mõnda üksikut andmesideprotokolli ühe seadme kohta. Vastavalt vajadusele on võimalik juurde paigaldada erinevaid riistvaralisi lüüsiseadmeid, kuid lahendus on kulukas ja võib vähendada süsteemi töökindlust.

FBD programm jälgib töötamisel signaali pidevust, ning signaalil puuduvad meta-andmed. Node-RED keskkonnas täidetakse programmi sõnumi pakettide kaupa, millel üldiselt on küljes meta-andmed. FBD programmist loetakse signaal Node-RED keskkonda ja selle edasine töötlemine toimub läbi sõnumi pakettide. Kui signaali väärtus või olek on „1", siis kahe sõnumipaketi vahel signaalil väärtus Node-RED keskkonnas puudub. Sellised põhimõttelised erinevused teevad keeruliseks ühe programmeerimiskeele pealt teisele ülemineku. Vastavalt seadme spikrimenüüs (*Help menu*) kirjeldatule, saab Node-RED keskkonna ja FBD programmi vahel infot vahetada ainult globaalseid muutujaid kasutades.

Programmeerimiskeskkonnas nimetatakse ühendatud sõlmede süsteemi vooks (*flow*) [37].

Node-RED rakenduse poolt täidetavad tööülesanded automaatjuhtimissüsteemi katsestendis on lihtsustatud kujul näidatud joonisel 6.1.



Joonis 6.1. Node-RED programmi ülesanded ja nende täitmise järjekord

Joonisel 6.1 on kirjeldatud Node-RED keskkonnas teostatud sammud häirete saatmiseks e-kirja teel. Esmalt kontrollitakse sammus 1, kas alarmide edastus on kasutaja poolt lubatud. Järgnevas punktis 2.1 loetakse alarmi sisend programmi ja kontrollitakse alarmi olekut. Sammu 3 täitmise teekond sõltub sellest, kas alarmiedastus kasutaja poolt on lubatud (punkti 1.3 olek mõjutab punkte 3.1 ja 3.2). Kui alarmiedastus on lubatud, saadetakse vastavasisuline e-kiri kasutajale ja liigutakse sammu 2 algusesse. Kui alarmiedastus on sammus 3 keelatud liigutakse punktist 3.2 tagasi sammu 2 algusesse ning protsess kordub seni, kuni kõik alarmid on läbi käidud. Seejärel liigutakse sammu „1“ algusesse ja kogu programmi täitmist alustatakse uuesti.

Kirjeldatud protsessi ülesannete täitmise järjekord ei pruugi Node-RED rakenduses olla selline nagu kirjeldatud, kuna programmi täitmine on sündmuste põhine ja ei jälgi FBD programmi täitmise loogikat (ülevalt alla ja vasakult paremale). Kirjeldatud on programmi ülesannete loogilist järjestust nagu programmi algoritmis on mõeldud.

Töös koostatud täielik Node-RED programmi voog on esitatud lisas X. Keskkonna kasutamise abimaterjal on leitav tootja kodulehelt, alamlehelt „Documentation“ ja peatükist „User Guide“ [37]. Töö järgnevates peatükkides on lähemalt kirjeldatud loodud programmivoogu ja kasutatud sõlmi.

6.2. Alarmi edastuse lubamine ja keelamine

Kõikide e-kirjade edastamise lubamiseks ja keelamiseks on tekitatud vastav globaalne muutuja „enEmail“, mis on lülitatav läbi kasutajaliidese. E-kirjade saatmise lubamiseks ja keelamiseks koostatud koodi osa Node-RED keskkonnas on esitatud joonisel 6.2.

Joonis 6.2. E-kirja saatmise lubamine ja keelamine

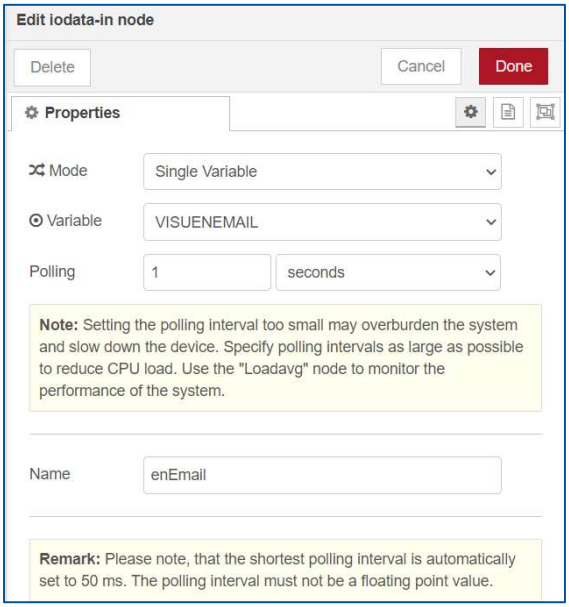
Programmi sõlmede väljundite kontrollimiseks on kasutatud silumise (*debug*) sõlme, mis on joonisel automaatselt genereeritud nimega „msg.payload“. Sõlme saab aktiveerida, kasutades selle lõpus olevat nuppu. Sisselülitatud sõlm kuvab vastava ahela sõnumi atribuudi,

payload, silumise (*debug*) infoaknas. Lisades iga voo elemendi juurde silumise sõlme, võimaldab see reaalajas jälgida ja kontrollida programmi eri etappide tööd.

Järgnevates peatükkides on täpsemalt kirjeldatud joonisel 6.2 esitatud programmivoos kasutatud sõlmed.

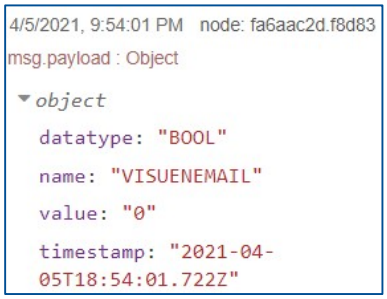
6.2.1 Globaalsete muutujate lugemine Node-RED keskkonda

Muutuja lugemiseks Node-RED keskkonda on kasutatud vastavat sõlme „iodata in“. Sõlme seadistused on esitatud joonisel 6.3.



Joonis 6.3. Sisendsõlme seaded

Režiimi (*mode*) väli võimaldab valida kõik (*All Variables*) või üksiku muutuja (*Single Variable*). Programmivoo eesmärk on ühe muutuja abil keelata kõikide e-kirjade saatmine, seega sobib siia valikusse üksikmuutuja. Muutuja (*Variable*) väljale on valitud vastav globaalne muutuja „visuEnEmail“, mille infot soovitakse keskkonda lugeda. Päringu sageduseks (*Polling*) on valitud üks sekund. Alarmide saatmine on teavitava eesmärgiga ning selle sagedus ei oma olulist rolli kirjeldatud automaatjuhtimissüsteemi töös. Sõlmele on antud nimi vastava globaalsele muutuja järgi, et lihtsustada programmi lugemist. Sõlme väljundiks on JavaScript objekt, mille sisu näide on esitatud, kasutades silumise sõlme joonisel 6.4.



Joonis 6.4. Node-RED sõlme „iodata in“ väljund sõnum

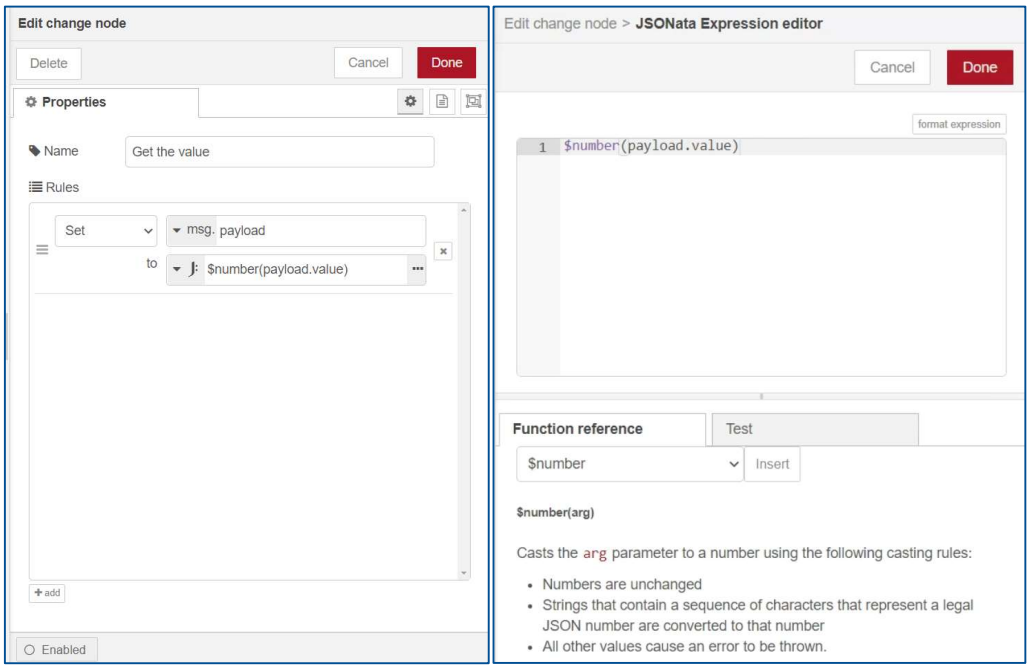
Jooniselt 6.4 on näha, et sõnumis on kirjeldatud andmetüüp (*data type:*), muutuja nimi (*name*), väärtus (*value*) ning ajatempel (*timestamp*). Edaspidises programmis on vaja kasutada ainult muutuja väärtust ja see on vaja ülejäänud sõnumist eraldada. Kahendmuutuja väärtus „1“ iseloomustab aktiivset alarmi ja selle sõnumist eraldamist on kirjeldatud järgmises peatükis.

6.2.2 Muutuja väärtuse eraldamine JavaScript objektist

Sõnumi töötlemiseks kasutatakse muudatuse (*change*) sõlme, mis võimaldab muuta sõnumi atribuute. Igasse sõlme saab kirjutada mitu tegevust ja neid täidetakse järjekorras. Sõlmes tehtavad toimingud on järgnevad:

1. Määra atribuut (*set*) – võimaldab valida atribuuti ja määrata selle väärtust;
2. Muuda (*change*) – vaheta või muuda mingit osa sõnumi atribuudist;
3. Liiguta (*move*) – liiguta või nimeta atribuut ringi;
4. Kustuta (*delete*) – kustuta atribuut.

Väärtuse eraldamiseks sõnumist on tarvis määrata uus väärtus atribuudile „msg.payload“. Selle jaoks on kasutatud käsku „määra atribuut“ (*set*). Väärtuse eraldamiseks kasutatakse JSONata avaldist, mis lihtsustab JSON andmete töötlust. JSONata võimaldab defineerida atribuuti „msg.payload“ numbrilise väärtusena. Sõlme seaded on esitatud joonisel 6.5.



Joonis 6.5. *Change* sõlme kasutamine

Sõlme redigeerimisaknas avanenud JSONata koodi väljale kirjutatud avaldis võtab sisendiks sõnumi *payload* atribuudi sisu ja teeb sellest numbrilise väärtuse, eemaldades üleliigsed jutumärgid. Sellise koodi tulemusel on *change* sõlme väljundiks ainult numbriline väärtus ilma ühegi muu sõnumi atribuudita.

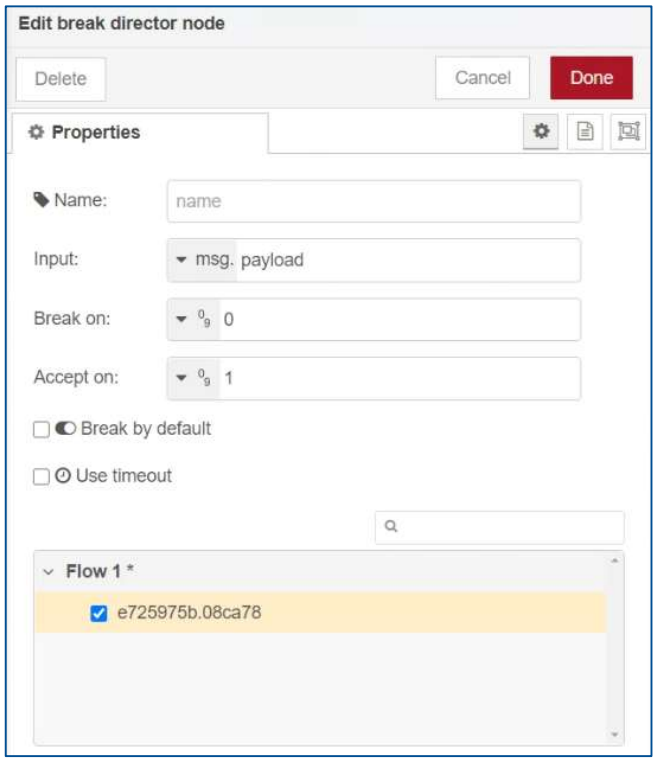
Kui eelnevalt kirjeldatud sõlme väljundiks on „0“, siis on vaja sõnumiahel katkestada, mida on kirjeldatud järgmises peatükis.

6.2.3 Sõnumiahela tingimuslik katkestamine

Programmivoo eesmärk on e-kirjade saatmise lubamine või keelamine vastavalt kasutaja valikule. Selleks on tarvis rakendada tingimust, mis lubab e-kirja edastust, kui muutuja „visuEnEmail“ väärtus on „1“ ja keelab saatmise väärtusel „0“. Siinjuhul on kasutatud võimalust rakendada Node-RED kogukonna poolt loodud sõlme. Node-RED keskkond lubab importida ja rakendada teiste kasutajate poolt loodud ja Node-RED kogukonda üles laetud sõlmi. Sõlmede info on säilitatud JSON andmevormingus ja see võimaldab tehtud programme ja sõlmi teiste kasutajatega jagada [37]. Tingimusliku katkestamise funktsiooni rakendamiseks leiti kogukonnast sõlm nimega „node-red-contrib-switch-break“ [38]. Järgnev osa põhineb sõlme kohta esitatud dokumentatsioonil, mis on leitav sõlme veebilehelt [38]. Sõlm koosneb kahest osast:

1. *Break director* (katkestuskäsu andja) – Asetseb programmivoos, kus toimub katkestuse tingimuse kontroll. Kontrollib siseneva sõnumi sisu, kui katkestuse tingimus on täidetud, edastatakse sõnumite blokeerimise käsk selle sõlmega seotud teistele elemendile *brake actor*;
2. *Break actor* (katkestuse täideviija) – Asetseb programmivoos, mille sõnumiedastus on vaja katkestada, katkestust täideviiv sõlm. Kui tuleb käsk *break director* sõlmest, peatatakse sõnumite edastuse läbi *brake actor* sõlme.

Selline lahendus võimaldab ühe tingimuse kontrolliga katkestada mitme programmivoo sõnumite edastus. Peatükis on vaatluse all katkestuse tingimuse kontroll ja selles programmivoos on kasutatud *break director* sõlme. Töös kasutatud sõlme seaded on esitatud joonisel 6.6 (*break actor* sõlm on kasutusel vaikesätetega).



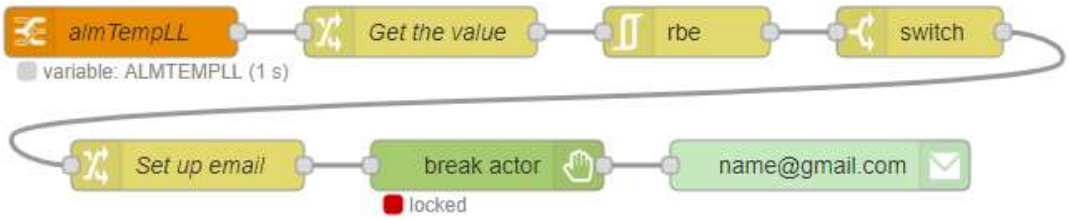
Joonis 6.6. *Brake director* sõlme seadistus

Sisendiks (*input*) on määratud „msg.payload“, mille sisu on eelnevalt juba töödeldud ja saab olla kas „0“ või „1“. Katkestuse tingimuseks on sõnumi numbriline väärtus „0“ ja sõnumi edastus jätkub kui väärtus on „1“. Järgnevas voo (*Flow 1*) väljas saab katkestuskäsu andja siduda ära täideviijatega, siinjuhul on täideviijaid üks, mis on näha joonisel 6.7. Sõlme paiknemine programmi voos on näidatud lisas X.

Järgnevas peatükis on kirjeldatud alarmi tekitamist Node-RED keskkonnas ja selle programmivoo erinevaid sõlmi.

6.3. Alarmi teavituse tekitamine Node-RED keskkonnas

Iga alarmi aktiveerimiseks on erinev globaalne muutuja ja selle kohane kirjeldus. Mistõttu on iga alarmi kohta vaja luua eraldi programmi voog. Need kõik lõppevad ühes „*Brake Actor*“ sõlmes, mis lubab või keelab alarmide e-kirjaga edastust. Kriitiliselt madala alarmi häireedastuse programmilõik Node-RED keskkonnas on esitatud joonisel 6.7.

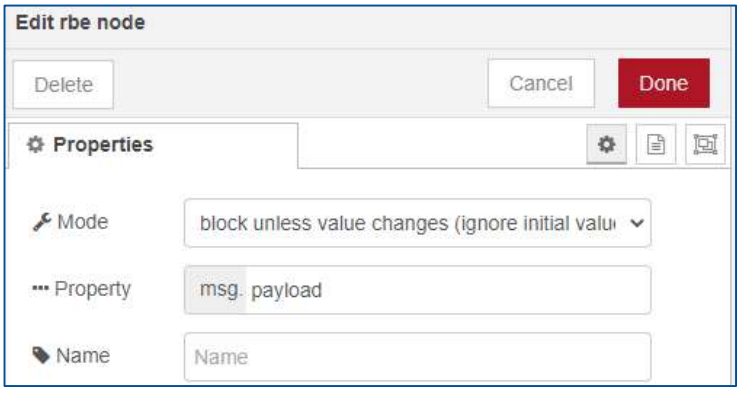


Joonis 6.7. Kriitiliselt madala alarmi e-kirja saatmine

Esimese kahe sõlme sisu on analoogne peatükis 6.2 kirjeldatule. Teiste sõlmede sisu on kirjeldatud järgnevates peatükkides.

6.3.1 Sisu muutusest tingitud sõnumi edastus

Voo kolmas sõlm „rbe“ (*Report by Exception*) edastab sõnumi juhul, kui selle sisu erineb eelmisest sõnumipaketist. Keskkonna olemusest tingituna edastatakse sõnumeid pakettidena, sellest lähtuvalt on oluline teada kui sõnumi sisus olev väärtus on muutunud, vastasel juhul edastatakse iga saabunud paketi korral vastava alarmiga e-kiri. Sõlme seaded on esitatud joonisel 6.8.



Joonis 6.8. Sõlme „rbe“ seadistus

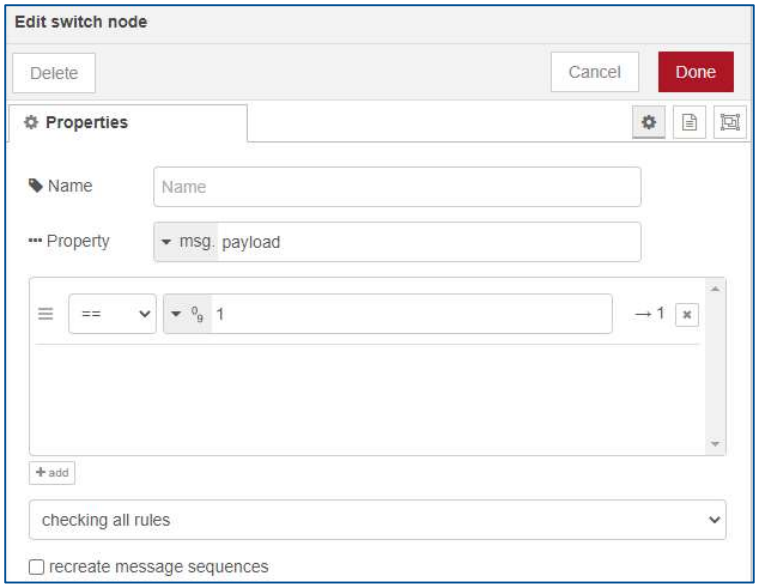
Režiimi väli võimaldab määrata sõnumi blokeerimise tingimuse. Valitud tingimus blokeerib sõnumi, kui see ei erine eelmisest. Esimese programmitsükli jooksul ei ole võimalik sisse tulnud sõnumit millegagi võrrelda ning seetõttu on valitud tingimus, mis ignoreerib esimese sõnumi sisu väärtust, vastasel juhul edastatakse programmi käivitumisel esimese tsükliga alarmi teavitus.

Omaduse väljale määratakse atribuut, mida sõnumites võrreldakse, siinjuhul on selleks sõnumi sisu ehk *payload*. Eraldi nime pole sõlmele määratud.

Järgnevas peatükis on kirjeldatud sõlme, mis tuvastab, kas alarm on aktiivne.

6.3.2 Alarmi oleku tuvastamine sõnumist

Eelmine sõlm „rbe“ edastab sõnumi, kui alarmi olek (sõnumi sisu) on muutunud, see võib olla kas „1“ või „0“. Järgnevalt on vaja lahendust, mis väärtuse „1“ korral edastaks sõnumi ja mingi muu väärtuse korral blokeeriks. Selle jaoks on kasutatud kommutaator sõlme (*switch*), mis vastavalt tingimusele suunab sõnumi sobivasse väljundisse. Sõlmel võib olla mitu väljundit, antudjuhul piisab ühest. Sõlme seaded on esitatud joonisel 6.9.



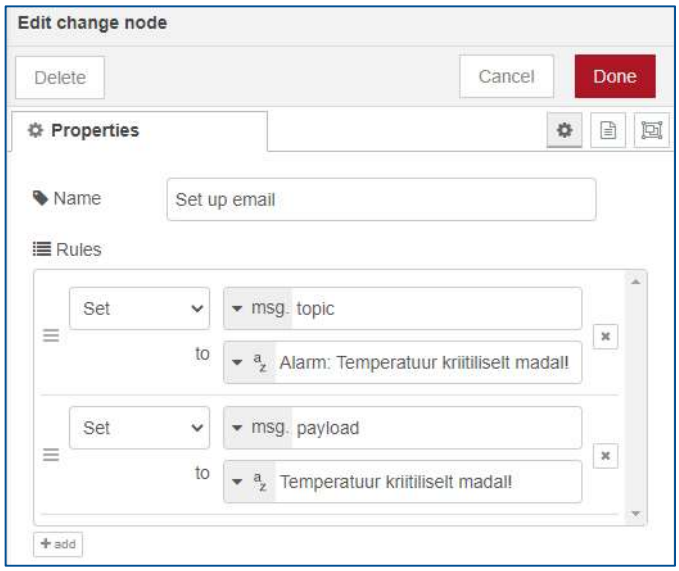
Joonis 6.9. Kommutaatorsõlme seaded

Omaduse lahtris on võrdluse atribuudiks sõnumi sisu ja kui selle väärtus on „1“, siis sõnum edastatakse. Lahtri alt on võimalik valida, kas kontrollitakse kõiki tingimused (*checking all rules*) või esimese tõese vasteni. Eraldi nime pole sõlmele määratud.

Kui alarmi olek on tuvastatud, on tarvis luua e-kirjale teemarida ja sisu, mida kirjeldatakse järgmises peatükis.

6.3.3 E-kirja sisu

Eelnevalt kirjeldatud sõlmede läbimisel on kõik alarmi genereerimiseks vajalikud tingimused täidetud. Järgnevalt on määratud e-kirjale pealkiri ja sisu. Selle jaoks kasutatakse *change* sõlme. Sõlme seaded on esitatud joonisel 6.10.



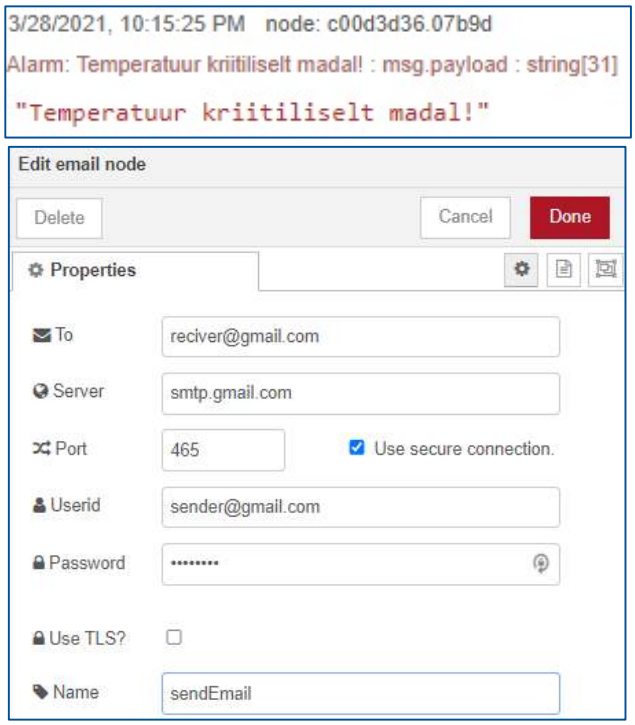
Joonis 6.10. *Change* sõlme seaded

Reeglite väljal on esimeses jaoks määratud sõnumile atribuut nimega „msg.topic“, mis määrab ära e-kirja teema, ja selle sisu on kirjeldatud järgmisel real. Järgnevas jaos on määratud sõnumi sisu „msg.payload“, kuhu kirjutatakse e-kirja minev alarmi teavitustekst. Muudatuse sõlmesid on kasutatud mitmes erinevas programmi osas erinevatel eesmärkidel, seetõttu on mõistlik anda tema funktsiooni täpsemalt kirjeldav nimi „Set up email“.

Programmivoo järgmine element on katkestuskäsu täideviija ehk *brake actor*, mida on kirjeldatud peatükis 6.2.3. Loodud sõnum sisaldab e-kirja teemarida ja sisu ning järgnevas peatükis on kirjeldatud e-posti saatmiseks vajaliku sõlme seadistusi.

6.3.4 E-posti seadistus

Eelnevas punktis täidetud e-kiri edastatakse sõnumina Node-RED keskkonnas olevasse e-kirja saatmise sõlme, mis on leitav rakenduse teegist.Töös on kasutatud ettevõtte Gmail kontot ja kirja saatmiseks on kasutatud Google e-posti serverit. Google piirab teenuse kasutamist mitteturvalistele seadmetele ja rakendustele, mistõttu sõlme looja on viidanud vajadusele muuta e-posti serveri kontol vastav seadistus *Less secure app access* tõeseks [39]. Sisenev näidissõnum ja e-posti sõlme seaded on kujutatud joonisel 6.11.



Joonis 6.11. Siseneva sõnumi näidis ja e-kirja edastamise sõlme seaded

Joonisel 6.11 on näidatud sõlme seadistusparameetrid on kirjeldatud järgnevalt:

1. *To* – E-kirja adressaat, kellele alarmid saadetakse. Kui e-kirjale on vaja mitut adressaati, tuleb need eraldada teineteisest komaga;
2. *Server* – E-kirja saatva e-posti serveri aadress. Gmaili poolt kasutatav SMTP serveri aadress on smtp.gmail.com;
3. *Port* – kasutatakse porti 465 ja turvalise ühenduse kasutamine on aktiveeritud. Sõlm kasutab „nodemailer“ paketihaldus moodulit, mis TLS valikute all kirjeldab port 465 kasutamisel aktiveerida *Use secure connection* [39][40].
4. *Userid* – E-kirja saatva konto kasutajanimi;
5. *Password* – E-kirja saatva konto salasõna;
6. *Use TLS* – Võimaldab aktiveerida turvalisemaks andmevahetuseks TLS protokolli;
7. *Name* – Sõlme nimi.

Node-RED keskkonna abil on teostatud üks tööülesandes esitatud kõrvalülesanne – alarmide edastus e-kirja teel. Teine kõrvalülesanne, kasutajaliides, on kirjeldatud järgmises peatükis.

7. Kasutajaliidese loomine

Tehnoloogilist protsessi juhtival ja kontrollival isikul on vaja süsteemiga suhtlemiseks kasutajaliidest, mis võimaldab jälgida protsessi erinevaid parameetreid, muuta seadeväärtusi ja annab ülevaate süsteemi üldisest olekust. Töös kasutatakse selle loomiseks kontrollerisse sisseehitatud visualiseerimiskeskkonda. Seadmesse integreeritud visualiseerimislahendused sobivad väga lihtsate süsteemide protsesside kuvamiseks, sest keskkonna pakutud vahendid on piiratud võimalustega. Protsessi visualiseerimine ei ole kontrolleri põhiülesanne. Visualiseerimiskeskkonnaga saab siduda programmis olevaid globaalseid muutujaid. Järgnevalt on kirjeldatud visualiseerimiskeskkonnas kasutajaliidese loomist.

7.1. Visualiseerimiskeskkonna ülevaade

Visualiseerimiskeskkonnale ligipääsemise tingimused ja võimalused on kirjeldatud peatükis 4.1.2. Keskkonna pakutavate elementide valik ja kirjeldus on esitatud tabelis 7.1.

Tabel 7.1. Visualiseerimiskeskkonnas olevad elemendid

Element	Kirjeldus
A Label	Tekstiväli.
A Variable output	Globaalse muutuja väärtuse lugemiseks. Muutuja tüüp võib olla täisarv või ujukomaarv.
B Button	Võimaldab lülitada kahendmuutuja väärtust olekusse üks või null. Seni kuni hoitakse nuppu all edastatakse väärtus üks, vastasel juhul tagastub väärtusele null.
B Link	Võimaldab navigeerida visualiseerimiskeskkonna eri lehtede vahel.
↔ Slider	Liugur, mida liigutades saab sujuvalt muutuja väärtust suurendada või vähendada etteantud skaala ulatuses.
Rectangle	Ristkülik, mida saab kasutada kujunduselemendina. Võimaldab ka üles laadida ja kuvada pilti.
Trace	Joongraafik. Võimaldab kuni kahe muutuja kujutamist ühel graafikul. Sobib muutuja ajaloo kuvamiseks, kuid selle jaoks peab kasutajaliides olema seadmes avatud.
Bar	Tulpdiagramm, mis kuvab ühe muutuja väärtust vertikaalsihis.
LED	Visuaalne indikaator kahe olekuga – sees või väljas, kahendmuutuja oleku kuvamiseks.
Switch	Ümberlüliti kahe olekuga, mis on visuaalselt eristatavad. Kahendmuutuja väärtuse muutmiseks ja oleku kuvamiseks.
Variable input	Võimaldab andmete kirjutamist globaalsesse muutujasse. Muutuja tüüp võib olla täisarv või ujukomaarv.
XY	X ja Y teljega graafik kuhu saab kuvada kuni kaheksa punktiga joongraafiku. X ja Y jaoks eraldi muutujad (kokku kuni 16 muutujat).

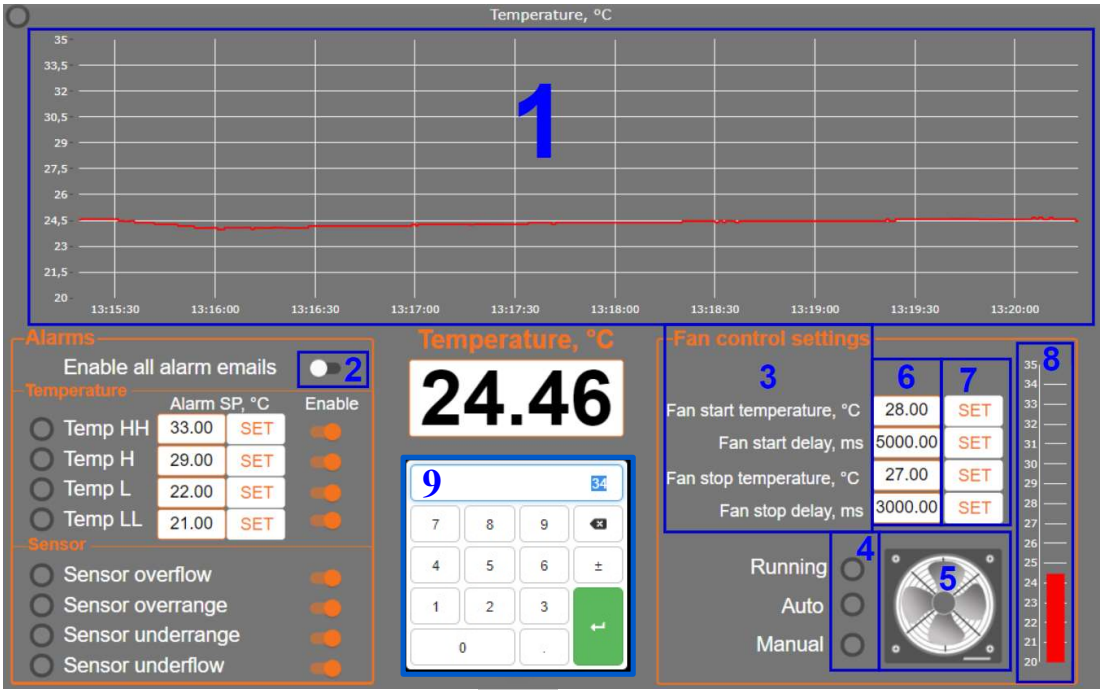
Kasutajaliidese loomiseks saab rakendada tabelis 7.1 nimetatud elemente. Kõikide elementide nähtavust (*visibility*) on võimalik siduda eraldi muutujaga, mis võimaldab kontrollida nende kuvamist näiteks nupuvajutuse või koodi sisestamisega. Visualiseerimiskeskkonna redigeerimisvaade on näha lisas V, joonis V.5.

Kasutajate loomine ja õiguste muutmise võimalus kontrolleris puudub ja seega sobib selline lahendus kasutamiseks eriala spetsialistile, kes on loodud süsteemiga tuttav. Kui kasutajal

puudub teadmine nii tehnoloogilisest protsessist kui kasutatud programmeerimiskeskkondadest on oht automaatjuhtimissüsteemi tahtmatult rikkuda.

7.2. Kasutajaliidese esileht

Kasutajaliidese esilehele on välja toodud kõik põhilisemad süsteemi olekud ja parameetrid, mis annavad ülevaate protsessist. Oluline on vahet teha muutujatel, mille väärtust on vaja ainult lugeda (R – *read*) ja nendel, mida on vaja lugeda ja kirjutada (R/W – *read / write*). Keskkond ei võimalda ühe elemendiga mõlemat funktsionaalsust lahendada ja selline tegevus vajab kahte erinevat kasutajaliidese elementi. Andmeid, mida esilehel kuvatakse koos muutuja tüübi ja funktsionaalsusega, on toodud lisas XI. Ülevaade loodud kasutajaliidesest on esitatud joonisel 7.1.



Joonis 7.1. Kasutajaliides

Joonisel 7.1 esitatud kasutajaliidese lehe mõõtmed on vabalt määratavad, töös on selleks 1280 x 800 pikslit. Lehe ülaosas on temperatuurigraafik ja selle all keskel suurelt hetke mõõdetud temperatuuri väärtus. Sinisega on välja toodud näited erinevatest visualiseerimiskeskkonna elementidest, mida on töös kasutatud.

1. Joongraafik (*Trace*);
2. Lüliti (*Switch*);
3. Tekstiväli (*Label*);
4. Indikaator (*LED*);
5. Pildikast (*Rectangle*);
6. Loetav numbriväli (*Variable output*);

7. Kirjutatav numbriväli (*Variable input*), avab klaviatuuri hüpikakna, mis on tähistatud numbriga 9;
8. Tulpdiagramm (*Bar*).

Kokku on töös kasutatud kaheksat elementi kaheteistkümnest võimalikust. Leht on grupeeritud erineva funktsionaalsusega aladeks, mis on piiritletud oranži kastiga.

Alarms sektsioonis on vasakul pool esitatud värviliste indikaatoritega alarmide olekud koos lühikese nimetusega ja nende kõrval vastava alarmi genereerimise seadeväärtused. Sarnast loogikat on rakendatud terve lehe ulatuses. Kriitiliselt kõrged, madalad ja riistvaralised alarmid on punase ning hoiatused kollase indikatsiooniga.

Alarmide üldist e-kirja edastust saab aktiveerida ja desaktiveerida *Alarms* sektsioonis nupust *Enable all alarm emails*. Üksiku alarmi aktiveerimiseks ja desaktiveerimiseks on rea lõpus *Enable* veerus oma lüliti. Kontrolleri taaskäivitusel lähtestatakse üksikute alarmide lüliti vaikeolekusse – aktiivne. Numbrilises aknas kuvatakse seadeväärtus ja *SET* nupust avaneb numbrite sisestuseks klaviatuur hüpikaknas seadeväärtuse muutmiseks.

Fan Control settings parameetrite osas on lisaks olekutele ja seadeväärtustele välja toodud visuaalne ventilaatori pilt ja vertikaalne termomeetri kujutis, mis iseloomustab süsteemi tehnoloogilist lahendust. Ventilaatori eesmärk on jahutada keskkonda, kus paikneb temperatuuriandur. Ventilaatori juhtimisega seotud indikatsioonid on aktiivolekus rohelised. Valminud stend on näha lisas III.

7.3. Püsimällu kirjutamine ja sealt lugemine läbi kasutajaliidese

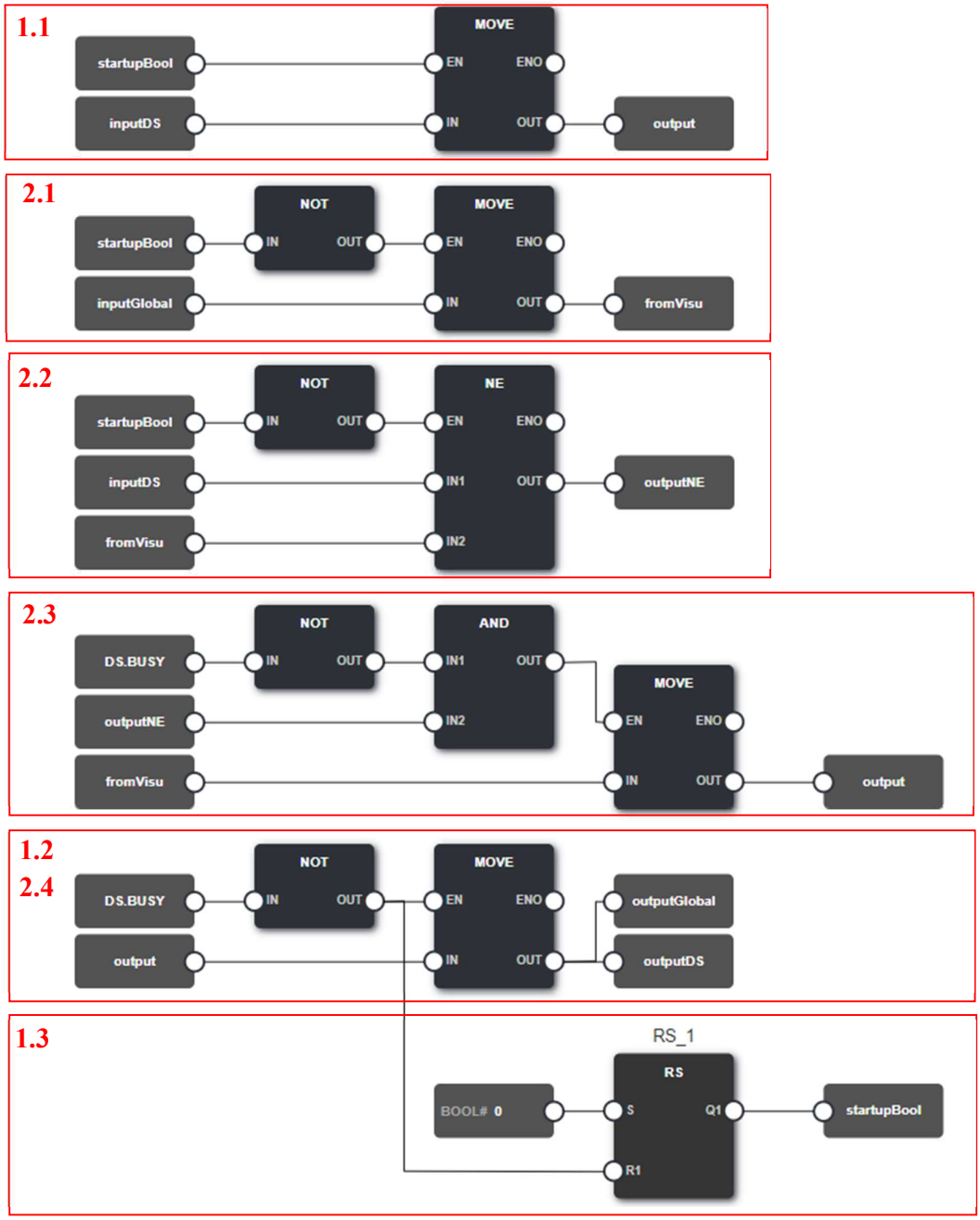
Visualiseerimiskeskkonnas esitatud seadeparameetrid peavad olema kasutajale muudetavad. Need muutujad peavad säilitama oma väärtuse ka peale kontrolleri toitekatkestust või programmiuuenduse laadimist. Muutujate olekute säilitamise jaoks on tarvis defineerida need kui püsimälu muutujad, mis kirjutatakse püsimälu mälualasse. Kasutatud kontrolleril on selliseks otstarbeks säilitatud andmete sektsioon (*Data Storage* ehk DS). DS muutujate käsitlemisel tuleb silmas pidada kirjutuskordasid, sest sage ülekirjutamine viib mäluosa kahjustumiseni. Töös kirjutatakse DS väärtusi üle ainult kasutaja käsul, mistõttu on oht püsimälu kahjustumiseks väike. Kõik muutujad, mille väärtust on tarvis säilitada, tuleb luua uuesti DS sektsioonis (vt lisa VI joonis VI.2 püsimälu muutujad).

Kasutajaliidese loomisel tuvastati, et säilitatud andmete sektsiooni muutujad on ligipääsetavad kõikidele funktsiooniplokkidele FBD programmis, kuid ei ole kättesaadavad visualiseerimiskeskkonnast. Vastavasisuline päring sai esitatud kontrolleri kasutajatoele, kes seda ka kinnitas.

Probleemi lahendamiseks on tekitatud kõikidele püsimälu muutujatele vaste globaalse muutuja näol. DS ja globaalset muutujat ei saa omavahel otse siduda. Kui püsimälu muutuja defineerib vastava globaalse muutuja, saadakse alati viimane programmi salvestatud väärtus, kuid seda väärtust ei saa kasutajaliidese abil muuta. Kui defineerida teistpidi, siis programmi käivitamisel on globaalse muutuja väärtus „0“ (või muu eeldefineeritud väärtus) ja

püsimällu salvestatud muutuja kirjutatakse üle. Kui neid kahte koodilõiku ümber kombineerida või koos kasutada, siis programmi täitmise olemus (programmi täidetakse ülevalt alla ja vasakult paremale) ei luba seda probleemi sobivalt lahendada.

Kasutajaliidese kaudu DS muutujate väärtuste lugemiseks ja kirjutamiseks on koostatud järgnev programmilõik, mis on esitatud joonisel 7.2.



Joonis 7.2. Globaalse ja püsimälu muutuja lugemine ja kirjutamine

Joonisel 7.2 näidatud programmil on kaks võimalikku läbimisteekonda, sõltuvalt sellest, kas tegemist on kontrolleri käivitusprotsessiga (esimene programmi täitmise tsükkel peale toite taastamist või programmilaadimist), joonisel tähistatud sammud 1.1…1.3, või normaaltöö

olekuga, joonisel tähistatud sammud 2.1…2.4. Käivitusprotsessiga seotud koodi rakendatakse ühe korra seadme käivitamisel või pärast programmilaadimist.

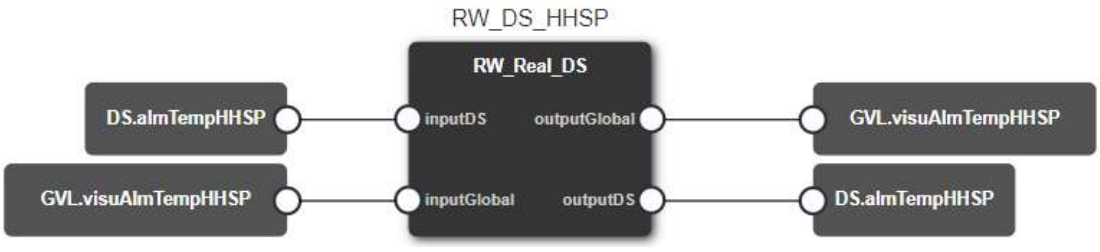
Joonisel 7.2 kirjeldatud programmilõigu eesmärk on omistada funktsiooniploki väljundile „outputGlobal“,mis on esitatud kasutajaliideses, püsimälust vastava muutuja „inputDS“ väärtus. Programmi sammud on lahti kirjeldatud järgnevalt:

1.1. Esimesel programmitsüklil on määratud muutujale „startupBool“ vaikimisi algväärtus „1“. Sisendiks võetakse püsimälust muutuja „inputDS“ ning läbi „MOVE“ funktsiooni omistatakse väärtus ploki sisesele lokaalsele muutujale „output“.
1.2. Globaalsele muutujale „outputGlobal“ omistatakse sisendi „output“ väärtus juhul, kui süsteemimuutuja „BUSY“ ei ole rakendunud. Nimetatud muutuja on aktiivne kui parasjagu toimub püsimällu kirjutamine, mistõttu on sel hetkel välistatud uue info salvestamine püsimällu.
1.3. Käivitusprotsessis kirjutatakse programmi esimese läbimise lõpus muutuja „startupBool“ väärtuseks „0“, seda tingimusel, et eelmisel sammul ei olnud kontroller püsimällu kirjutamisega hõivatud. Ploki „RS“ sisend „S“ (*set*) on seotud konstant väärtusega „0“, mis välistab kahendmuutuja „startupBool“ aktiivse oleku mingil muul ajal, kui ainult esimese programmitsükli jooksul, mil on jõus muutuja määratud algolek „1“.

Teiseks märgitud koodi sammude eesmärk on võtta kasutajaliidese kaudu sisestatud väärtus ja kirjutada see vastavasse püsimälu muutujasse.

2.1. Muutujale „fromVisu“ kirjutatakse ploki sisendi „inputGlobal“ väärtus. Viimasele omistatakse väärtus läbi kasutajaliidese.
2.2. Järgnevalt võrreldakse püsimälu muutuja väärtust kasutajaliidese omaga. Kui need on erinevad, siis saab järeldada, et kasutaja on sisestanud uue väärtuse ja võrdlusplokk „NE“ (*not equal*) väljund rakendub.
2.3. Kui kontroller ei ole hõivatud püsimällu kirjutamisega ja punktis 2.2 kirjeldatud väljund on rakendunud, siis muutujale „output“ omistatakse uus väärtus muutujast „fromVisu“.
2.4. Kirjeldus vastavalt sammus 1.2 toodule. Uus muutuja väärtus kirjutatakse ploki mõlemasse väljundisse.

Joonisel 7.2 esitatud programmilõigust on tehtud funktsiooniplokk. Näitena on joonisel 7.3 esitatud kriitiliselt kõrge temperatuuri seadeväärtuse püsimälust lugemise ja sinna kirjutamise funktsiooniplokk.



Joonis 7.3. Püsimälu ja globaalse muutuja väärtuste omistamise funktsiooniplokk

Joonisel 7.3 esitatud funktsiooniploki esimesel real olevad muutujad „DS.almTempHHSP“ ja „GVL.visuAlmTempHHSP“ kirjeldavad, püsimälu muutujast globaalsesse muutujasse kirjutamist. Alumised kaks muutujad on seotud kasutajaliidesest püsimällu kirjutamisega. Kõik alarmid, mille seadeparameetrit saab kasutajaliidese abil muuta, on lahendatu analoogselt. Ploki muutujad on esitatud lisas XII.

Ventilaatori käivitamise ja seiskamise jaoks on vajalik muuta püsimällu salvestatud viidete parameetreid läbi kasutajaliidese. Viited on täisarvulised muutujad, mis on esitatud millisekundites ja nende lugemiseks ja kirjutamiseks on tehtud eraldi funktsiooniplokk analoogselt joonisel 7.2 ja näidatud programmile. Muudetud on ujukomaarvudega seotud elemendid ja muutujad täisarvuliste vastu. Vastav funktsiooniplokk (RW_Int_DS), programmilõik ja muutujad on esitatud lisas XIII.

E-kirjade edastamise lubamiseks ja keelamiseks on kasutatud kahendmuutujat, mille väärtus peab samuti säilima püsimälus peale toitekatkestust või programmilaadimist. Joonisel 7.2 esitatud programmilõigus on muudetud vastavad elemendid ja muutujad selliselt, et analoogne lahendus toimiks ka kahendarvudega. Funktsiooniplokk (RW_Bool_DS), programmilõik ja muutujad on esitatud lisas XIV.

8. Hinnang lahendusele

Töö koostamisel rakendati erinevaid funktsionaalsusi, mida kontrolleri arenduskeskkond pakkus. Põhiliselt leidis rakendust FBD programmeerimiskeskkond ja visualiseerimiskeskkond. E-kirjaga häireedastuse jaoks kasutati Node-RED rakendust, mis on seadme üks lisafunktsionaalsustest. Arenduskeskkond ei vaja eraldi litsentsi soetamist ja sinna on lisaks FBD programmeerimiskeskkonnale integreeritud palju lisavõimalusi (Node-RED, OPC UA, visualiseerimiskeskkond), mis teevad sellest kontrollerist väga mitmekülgse seadme. Seadmel on kaks võrgukaarti ja see võimaldab kasutada kontrollerit mitmekihilises võrgustruktuuris, mis on oluline turvalisuse seisukohast.

Töös esitatud lahenduse põhiülesannet (protsessi juhtimine) täidab FBD rakendus. Kõrval ülesanneteks võib lugeda kasutajaliidese ja alarmiedastuse. Hinnangu andmisel on suurema kaaluga põhiülesandega seotud märkused.

Järgnevalt on välja toodud tähelepanekud ja hinnangud, mis tekkisid tööprotsessi käigus.

8.1. Funktsiooniplokkskeem programmeerimiskeskkond

Kontrollerisse integreeritud arenduskeskkond on mugav lahendus programmeerijale, kes ei pea oma arvutisse installeerima arendustarkvara ja selle erinevaid versioone. Lahendus ei sea erilisi eeltingimusi programmeerija kasutuses olevale riistvarale. Seadmepõhise arenduskeskkonna puhul saab arendustegevust teha ainult konkreetse seadmega ühenduses olles. Kontrollerile ligipääs peab olema arendustegevuse ajal tagatud, mis seab programmeerimisele suured piirangud olukorras, kus kontroller on objektil paigas ja sellele puudub ligipääs üle interneti.

Kasutatud FBD programmeerimiskeel võimaldab suuremaid programmilõike esitada kompaktselt funktsiooniplokina, mis annab parema ülevaate programmi tööst ja lihtsustab silumist. Kasutaja poolt loodud plokke on võimalik sarnaste ülesannete lahendamiseks kopeerida, mis kiirendab programmi kirjutamist ning võimaldab kasutajal lihtsalt luua teeke erinevate ülesannete lahendamiseks. Automaatikakontrollerile on tulevikus ette nähtud ka ST (*Structured Text*) programmeerimiskeele tugi. Mõned teised tootjad (näiteks Schneider-Electric ja Unitronics) kasutavad tasuta litsentsiga arendustarkvara puhul kontrolleri programmeerimiskeelena LD (*Ladder*), mis on suuremahuliste programmide puhul raskesti loetav ja silutav.

Püsivara uuenduse järgselt mindi üle HTTPS protokollile ning ühtlasi paranes ka arenduskeskkonna reageerimiskiirus ning koos sellega kasutusmugavus. Aegajalt esines süsteemi hangumist ja ebaõnnestunud programmilaadimisi. Seadme kahetuumaline ARM protsessor ja mälu peavad toime tulema samaaegselt FBD programmi, Node-RED rakenduse, kasutajaliidese ja ka arenduskeskkonna käitamisega, mis tähendab, et protsessori ja mälu ressurss on nende vahel jagatud. Siin projektis ei olnud probleem tuntav, kuid suuremamahulise programmi korral võib tekkida ressursi kasutamisel kitsaskohad. Püsivara uuenduse järgselt oli kontrolleril mälu kasutatud 61% ulatuses ning töö lõpuks on mälukasutus umbes 85%.

Puudub täpsem ülevaade, kuidas on mälukasutus jaotunud erinevate rakenduste vahel. Loodud programmid on pigem väikesemahulised ja hõlmavad põhiliselt andurite ja alarmide eelseadistust. Tehnoloogilise protsessi juhtimise algoritm on väike osa kogulahendusest.

FBD arenduskeskkonna kasutajaliides on arusaadav, kuid programmi visuaalne esitus võtab funktsiooniploki kujunduse tõttu liialt palju ruumi töölehel. FBD programmi töölaud on fikseeritud mõõtmetega ja see seab piiri võimalikule programmi suurusele. Töölehe ruumilist kasutust koos valminud programmiga on näha lisas XV. Lisas olevast joonisest saab järeldada, et valminud programm võtab umbes 25% kogu töölehest. Iga funktsiooniploki sees on omakorda sama suur fikseeritud mõõtemetega tööleht.

Töö väljundiks on automaatjuhtimissüsteemi testimiseks mõeldud lahendus. Arendusplatvormil puudub simulatsioonikeskkond, mistõttu ei saa kirjutatud programmi lihtsasti testida. Lisaks sellele ei saa töötavas FBD programmis muutujatele väärtusi omistada. Testimise käigus on vaja muudatusi teha redigeerimisvaates, programm kompileerida ja kontrollerile peale laadida, mis teeb programmi testimise ajamahukaks. Valminud lahenduses on võimalik kõik programmi katsetused läbi viia reaalse riistvara peal, kuid mõne teise projekti puhul ei ole füüsilise riistvara peal testimine otstarbekas, ohutu või üldse võimalik.

Kontrolleri kasutamise abistamiseks mõeldud materjal spikkermenüü ja kontrolleri dokumentatsiooni näol on puudulikud. Küsimuste ja probleemide korral võib ilmneda vajadus kontakteeruda seadme kasutajatoega.

Järgnevalt on esitatud FBD programmeerimiskeskkonna eelised ja puudused, mis tuvastati tööprotsessi käigus.

Eelised:

1. Seadmepõhine arenduskeskkond;
2. Funktsiooniplokkskeem programmeerimine;
3. Arenduskeskkond kasutab HTTPS protokolli.

Puudused:

1. Programmeerimine võimalik ainult kontrolleriga ühenduses olles;
2. Ressurss on jagatud erinevate rakenduste vahel;
3. Fikseeritud suurusega programmeerimise töölaud;
4. Puudub simulatsioonikeskkond;
5. Puudulik abimaterjal.

Nendest FBD rakenduse programmeerimiskeskkonna tähelepanekutest võib järeldada, et valitud kontroller ei ole sobiv suuremahulistes projektides kasutamiseks ja ei sobi ettevõtte põhilahendustesse. Seda eelkõige simulatsioonikeskkonna puudumise ja fikseeritud töölaua suuruse tõttu. Kasutamisele seab piiranguid vajadus olla kontrolleriga ühenduses terve programmeerimise aja, mis ei ole alati teostatav olukorras, kus kontroller on objektile juba paigaldatud. Programmi suurusele võib piirangu seada ka mäluressurss, mis on loodud programmi näol juba 85% ulatuses kasutatud.

Järgnevas peatükis on kirjeldatud tähelepanekuid, mis tekkisid visualiseerimiskeskkonda kasutades.

8.2. Visualiseerimiskeskkond

Kontrollerisse integreeritud visualiseerimiskeskkond on loogilise ülesehitusega ja kasutab vektorgraafikat elementide kuvamiseks. Väikese funktsionaalsuse tõttu on võimalused piiratud. Keskkonnas on olemas kõik elementaarne, et anda ülevaade protsessist. Püsimälu muutujatele ligipääsu puudumine lisas oluliselt keerukust FBD programmile.

Visualiseerimiskeskkond võimaldab kahendarvude (Bool), täisarvude (Int) ja ujukomaarvude (Float) lugemist ja kirjutamist. Töös on häireedastuseks kasutatud e-kirja, mille adressaadi ja sisu muutmist ei ole võimalik kasutajaliidesega lahendada. Kontrolleril puudub täielikult tekstilise muutuja kasutamise võimalus programmis.

Ligipääs kasutajaliidesele läbi HTML5 toega veebibrauseri toimis nii Windows kui ka Android operatsioonisüsteemi puhul. Kui visualiseerimiskeskkonnale läheneda läbi arenduskeskkonna, on vajalik kasutaja tuvastus. Kasutajaliidese avamine õnnestub sealt kaudu ainult Windows operatsioonisüsteemiga. Android seadme kasutamisel esineb veateade. Kui kasutada ligipääsuks IP aadressi, millele on lisatud „/visu“, ühendatakse koheselt kasutajaliidesega. Seadistusest on võimalik kasutaja tuvastamine kasutajaliidese jaoks desaktiveerida. Sel viisil on kasutajaliides ligipääsetav kõigile, kes on kontrolleriga samas võrgus. Puudub võimalus tekitada erinevate ligipääsuõigustega alasid ja kasutajaid.

Kontrolleri põhiülesanne loodud lahenduses on FBD programmi töö ja sellega seotud tehnoloogilise protsessi juhtimine. Teiste ülesannete seadmine kontrollerile võib pärssida selle võimekust tegeleda oma põhiülesandega. Visualiseerimiskeskkonda tuleks rakendada minimaalselt ja ainult kõige vajalikum osa lisada kasutajaliidesesse. Võimalusel kasutada kontrollerist sõltumatut riistvara või tarkvara kasutajaliidese loomiseks.

Eelised:

1. Loogiline ülesehitus;
2. Sobib hästi elementaarse info kuvamiseks;
3. Vektorgraafika;
4. Kasutajaliidesele ligipääs läbi HTML5 toetava brauseri.

Puudused:

1. Piiratud funktsionaalsus;
2. Puudub ligipääs püsimälumuutujatele;
3. Puudub tekstilise muutuja kasutamise võimalus;
4. Puudub võimalus kasutajaid ja õigusi lisada ning muuta.

Visualiseerimiskeskkond võimaldab luua vähese funktsionaalsusega kasutajaliidese, mille kaudu saab kasutaja süsteemi jälgida, juhtida ja seadeparameetreid muuta. Keskkond salvestab graafiku ajalugu ainult avatud kasutajaliidese korral ja eelnevalt defineeritud perioodi ulatuses. Alarmide ajaloo kuvamiseks võimalused puuduvad. Visualiseerimiskeskkond ei sobi kasutamiseks lahendustes, kus on oluline jälgida tehnoloogilise protsessi ajalooga seotud andmeid, näiteks alarmide ajalugu (millal alarm tekkis ja millal tagastus) või mingi protsessi ajalugu, näiteks temperatuuri muutust.

Järgnevas peatükis on kirjeldatud Node-RED rakendusega seotud tähelepanekuid.

8.3. Node-RED programmeerimiskeskkond

Häirete edastuseks on kasutatud e-kirja, mis on teostatud Node-RED arenduskeskkonda kasutades. FBD ja Node-RED on sisult väga erinevad programmeerimiskeeled ja kohanemine ühelt keelelt teisele võib vähese kogemuse puhul olla raskendatud. Kahe keele koos kasutamine ühes seadmes lisab süsteemile funktsionaalsust ja paindlikkust, kuid selle täielikuks rakendamiseks vajab kasutaja teadmisi kahes programmeerimiskeeles, mis võib pikendada õpikõverat.

Siinjuhul kehtib eelmistes peatükis mainitud probleem kontrolleri ressursi kasutamisega, juhul kui Node-RED ei ole põhiülesannet täitev rakendus. Loodud programmi ülesanne on saata häireteateid, mis ei ole kõige ajakriitilisem tegevus. Ressursi kokkuhoiu eesmärgil uuendatakse muutujaid rakenduses ühesekundilise intervalliga.

Node-RED keskkonna teegis olev sõlm „*Memory*“ kuvab kontrolleri mälukasutust, kuid selle järgi on kasutusel 97% mälust. Vastavalt kasutajatoelt saadud tagasisidele on tegemist süsteemi veaga ja teemat uuritakse. Nad tuvastasid, et süsteem on eraldanud selles projektis Node-RED rakenduse jaoks 3MB mälu.

Eelised:

1. Võimaldab ühendada omavahel riistvara erinevate teenusetega;
2. Silumise sõlm võimaldab sõnumite sisu täpselt jälgida, mis lihtsustab programmi testimist ja silumist.
3. Suur sõlmede valik teegist;
4. Aktiivne kogukond;
5. Võimalus importida ja eksportida programmilõike.

Puudused:

1. Programmeerimiskeel ei vasta automaatikakontrolleritele suunatud IEC 61131-3 standardile. Vajab programmeerija teadmisi teistes programmeerimiskeeltes.

Node-RED platvormi võimekus importida ja eksportida sõlmi ja terveid programmilõike ning toimiv aktiivne kogukond on suureks abiks alustavale kasutajale, aidates vähendada õpikõverat. Keskkonna suur funktsionaalsus teevad kontrollerist väga mitmekülgse seadme.

Kui ettevõttel peaks tekkima vajadus Node-RED pakutava funktsionaalsuse järele on see kontroller hea üleminekuseade. Kontrolleri ühendab endas nii traditsioonilisema tööstuslike automaatikakontrollerite programmeerimiskeele kui ka IoT suunalise Node-RED funktsionaalsuse. Nende kombineerimine pakub rohkelt uusi võimalusi, kuid FBD arenduskeskkonna piirangute tõttu peaks seadme kasutamise põhirõhk olema suunatud Node-RED rakendusele.

Järgnevas peatükis on hinnatud valminud lahenduse vastavust seotud osapoolte esitatud nõuetele.

8.4. Nõuete valideerimine

Järgnevalt on hinnatud, millised seotud osapoolte esitatud nõuded said täidetud. Nõuete vastavuse hinnangu paremaks esitamiseks ja eristamiseks on kasutatud kolme värvi: 1) roheline – vastab; 2) oranž – osaliselt vastab; 3) punane – ei vasta. Nõuded on grupeeritud vastavalt prioriteedile.

Peab olema (*must have*)

1. Ohutu inimestele ja seadmetele – Süsteemi komponendid ja seadmed on valitud ja paigaldatud selliselt, et need oleksid inimesele ohutud. Juhtmete ja seadmete tähistus aitab vältida valeühendusi.
2. Abistama uue töötaja integreerimist ettevõttesse – Eeldused selleks on loodud, et lihtsustada üldist automaatjuhtimissüsteemiga tutvumist. Küll aga ei ole see kontroller kasutusel ettevõtte tüüpilisemates projektlahendustes.
3. Kontroller peab omama Ethernet TCP/IP võrguliidest.
4. Kontrolleri programmeerimiskeel peab vastama IEC 61131-3 standardile – FBD programmeerimiskeel vastab standardile. Node-RED rakenduses kasutatav keel ei vasta.
5. Analoog signaali mõõtmise võimekus (4…20mA ja 0…10V).
6. Digisisendite mõõtmise võimekus.

Peaks olema (*should have*)

7. Lihtne seadmete vahetamine stendis – Nõude kirjelduses kasutatud termin „lihtne“ on suhteline. Automaatjuhtimissüsteemi katsestend on kompaktne seade, mille valmistamisel arvestati ka reservruumi olemasoluga. Erinevate ühenduste loomine vajab tööriistu ja seda võib teha ainult süsteemi tundev isik.
8. Stend annab ülevaate ettevõttes kasutatavatest tehnoloogiatest – Kõikide ettevõttes levinud komponentide kasutamine katsestendis ei ole otstarbekas ja rakendatav (vee rõhuandurid, vooluhulgamõõtja jms). Valitud seadmed teevad lahendusest ühe funktsioneeriva terviku. Kõik kasutatud andurid ja muud komponendid täidavad mingit eesmärki vastavalt esitatud tööülesandele.
9. Alarmide edastamine e-kirja teel.
10. Alarmide kuvamine kasutajaliidese avalehel.
11. E-kirja adressaadi muutmine läbi kasutajaliidese – Kontrolleri kasutajaliides ei võimalda teksti sisestust ja selle omistamist muutujale.
12. Modbus TCP/IP andmeside kasutamine kontrolleri ja mõne teise seadme vahel – Ettevõtte kasutatavatest tehnoloogiatest ei leitud sobivat seadet, mida oleks olnud mõistlik rakendada ja mis oleks täitnud tööülesandes mingisugust eesmärki. Eeldused selleks on loodud. Katsestendile on paigaldatud mittehallatav võrgukommutaator, mis võimaldab erinevate võrguseadmete ühendamist kontrolleri andmesidevõrguga. Modbus TCP/IP andmevahetus teiste seadmetega peab käima läbi Node/RED rakenduse.

Võiks olla (*could have*)

13. Stend peab olema lihtsasti teisaldatav, kompaktne, alusele kinnituv ja seinale kinnituv – Stend on käsitletav ühe inimese poolt, lauapealse alusena on kasutatud molbertit ja võre laadne montaažplaat võimaldab riputada seadet seinale.
14. Seadmed ja juhtmed peavad olema tähistatud.
15. Alarmi seadeväärtuste muutmine läbi kasutajaliidese.
16. Tarkvara peab võimaldama erinevate kasutajate ja rollide loomist – Kontrolleril on sisseehitatud kaks eelseadistatud õigustega kasutajat „admin“ ja „service“. Kasutajaid ja rolle ei saa juurde luua, ega nende õigusi muuta.
17. Kõikide või üksikute alarmide e-kirja teel edastamise väljalülitamine – Programm võimaldab e-kirja edastust desaktiveerida, kuid alarmide genereerimine jääb aktiivseks. Alarme saab ühekaupa vaigistada, mis peatab selle genereerimise ja e-kirja teel edastuse.

Ei ole (*won't have*)

18. Keskkond peab võimaldama raportite koostamist – Kontrolleris on võimalik piiratud kujul raporteid luua, kuid sellise funktsionaalsuse loomine ei ole töö mahus. Raporti koostamine vajab muutuja kohta ajaloo või statistika tegemist. Statistiliste andmete kogumine ja selle põhjal raportite koostamine ettemääratud perioodi ja intervalliga on selle kontrolleriga teoreetiliselt teostatav. Muutuja ajalooga seotud informatsiooni tekitamise ja käitlemise jaoks puudub kontrolleril funktsionaalsus. Tehnoloogilise protsessi ajalooga seotud andmete jälgimiseks ja analüüsiks on otstarbekas kasutada näiteks SCADA süsteemi.

Nõuete valideerimisest saab järeldada, et 10 nõuet (~56%) said täielikult rahuldatud. Neli nõuet osaliselt (~22%), ja neli nõuet (~22%) jäi rahuldamata. Kokku 14 nõuet sai kas täielikult või vähemalt osaliselt täidetud. Kolm nõuet jäid kontrolleri ebapiisava funktsionaalsuse tõttu rahuldamata. Ühe nõude rahuldamine ei mahtunud töö mahtu. Osaliselt täidetud nõuded on enamuses mittefunktsionaalsed nõuded. Nende täitmine on tingitud eelkõige nõude puudulikust kirjeldusest, kus ei ole defineeritud mõõdetavat suurust vaid põhineb subjektiivsel hinnangul. Nõuete vastavust hinnates on jäädud pigem kriitiliseks.

8.5. Hinnang kontrolleri sobivusele ettevõtte lahendustes

Kontrolleri sobivuse hindamisel on lähtutud automaatjuhtimissüsteemile esitatud nõudetest ja keskkondadele antud hinnangutest. Suurem mõju hinnangule on põhiülesannet täitval rakendusel.

UC20-WL2000-AC kontroller sobib lihtsamates projektlahendustes kasutamiseks, kus on oluline e-kirjaga teate edastus või muu Node-RED poolt pakutava funktsionaalsuse rakendamine. Ainult FBD programmi (või mõne muu standardis IEC 61131-3 nimetatud keele) kasutamiseks on olemas teisi kontrollereid (näiteks Schneider Electric M241 või Siemens S7-1200 seeria kontrollerid), mille arenduskeskkonnad pakuvad paremaid võimalusi, näiteks simulatsioonikeskkonda ja muutujate lihtsamat püsimällu defineerimist. Lisakitsendusi

seab fikseeritud suurusega FBD programmeerimise töölaud, mille piiridesse programm peab ära mahtuma.

Töös ei ole analüüsitud Modbus TCP/IP ja Modbus RTU andmeside rakendamist valitud kontrolleris. Seadme tehnilise toega suhtlemisel selgus, et tööstuslik andmeside teiste seadmetega peab käima läbi Node-RED keskkonna, mis teeb reaalajaprogrammi töö sõltuvaks kahest erinevast programmist ja programmeerimiskeelest. Modbus andmeside kasutamise vajadus esineb paljudes ettevõtte poolt pakutavates lahendustes, mida tuleks selle kontrolleri kasutamisel teisiti teostada kui hetkel kasutusel olevas kontrolleriseerias.

8.6. Lahenduse edasised kasutus- ja arendusvõimalused

Magistritöös on põhjalikult dokumenteeritud automaatjuhtimissüsteemis kasutatud kontrolleri seadistatus- ja programmeerimisprotsessi ning töö käigus valmis toimiv lahendus. Automaatjuhtimissüsteemi katsestendi saab kasutada nii turundusüritustel kui ka uuele töötajale kasutatavate tehnoloogiate esmatutvustamiseks. Töös kasutatud kontroller on suure funktsionaalsusega, eriti IoT suunal, tänu Node-RED keskkonnale. Node-RED võimaldab süsteemiga liita andureid seadmeid ja täiturmehhanisme, mis kasutavad suhtlemiseks erinevaid TCP/IP võrguprotokolle. Valitud kontrolleri arenduskeskkondade põhjalik analüüs võimaldab IoT lahenduste vajaduste tekkimisel kiirendada seadme kasutuselevõttu ettevõtte protsessides. Lähemat analüüsi vajab kontrolleri ja teiste seadmete vaheline Modbus andmeside lahenduse kasutamine. Kuidas kontrolleriga ühendada näiteks operaatorpaneeli või sagedusmuundurit?

Ettevõtte pakutavates lahendustes on olulise tähtsusega ka SCADA süsteem, mida töös käsitleti vähesel määral. Loodud lahenduse edasiarenduseks on võimalik kontroller siduda SCADA süsteemiga, et testida nende omavahelist sobivust, funktsionaalsust ja töökindlust.

Katsestendis rakendatud kontroller ei pruugi leida kasutust ettevõtte tööstussuunalistes põhilahendustes ja soovitav on see igapäevategevustes asendada sobivamaga (näiteks Schneider Electric M241 seeria kontrolleriga). Seejärel saab uue töötaja koolitamise eesmärgil kasutada töös esitatud tööülesannet ja lasta teha analoogne lahendus teist tüüpi kontrolleril, vastavalt seadme võimalustele. Valminud magistritööd saab kasutada testitud kontrolleri juhendmaterjalina ja võtta aluseks ka teistsuguste kontrollerite ja muu riistvara testimiseks ja nende sobivuse hindamiseks.

Kokkuvõte

Magistritöös kirjeldatud automaatjuhtimissüsteemi testimise käigus keskenduti kontrolleri programmeerimisele ja selle erinevate funktsionaalsuste rakendamisele. Esmalt kaardistati seotud osapoolte nõuded süsteemile. Seejärel koostati neid silmas pidades tööülesanne, mille lahendamisele ülejäänud töös keskenduti.

Tööülesandeks oli ventilaatori juhtimine vastavalt keskkonna temperatuurile, kasutajaliidese loomine ja alarmide edastamine e-kirja teel. Tööülesande lahendamiseks on kasutatud funktsiooniplokkskeem programmeerimiskeelt (FBD), seadmesse integreeritud visualiseerimiskeskkonda ja Node-RED rakendust. Kirjeldatud on automaatjuhtimissüsteemi erinevaid komponente ja nende valiku tingimusi ning ettevõtte poolt enim kasutust leidvat andmesidestruktuuri ja selle erinevaid kihte. Töö käigus on analüüsitud kontrolleri sobivust ettevõtte vajadustega ja valminud lahenduse edasisi kasutusvõimalusi.

Automaatikakontrolleri UC20-WL2000-AC tarkvara arenduskeskkond on seadmesse integreeritud. Arenduskeskkonnale ligipääsuks ja programmeerimiseks on vajalik HML5 toega veebilehitseja. Põhiülesande täitmiseks rakendati FBD programmeerimiskeelt, mis vastab standardile IEC 61131-3. Lisafunktsionaalsustest kasutati kontrolleri visualiseerimiskeskkonda ja Node-RED rakendust.

Kontrolleri põhiülesannet täitis FBD rakendus, kuhu programmeeriti ventilaatori juhtimisloogika, alarmide genereerimine ja teisi süsteemi toimimiseks vajalikke funktsionaalsusi. Seadme eripära tõttu tuli globaalsete ja püsimälu muutujate omavaheliseks andmevahetuseks koostada eraldi programmilõigud.

Kõrvalülesanneteks oli kasutajaliidese loomine ja e-kirjaga alarmide edastamine, mille teostamiseks kasutati Node-RED rakendust. Visualiseerimiskeskkonna võimalused kasutajaliidese loomiseks olid piiratud, kuid piisavad, et luua elementaarse funktsionaalsuse ja protsessist ülevaadet andev kasutajaliides. Node-RED lisab kontrollerile funktsionaalsust, kuid ühtlasi ka keerukust teise programmeerimiskeele näol, mis ei ole kirjeldatud standardis IEC 61131-3.

Töös hinnati valminud lahenduse sobivust esitatud nõuetele ning anti soovitusi edasiseks kasutus- ja arendustegevusteks. Töö alguses esitatud nõuetest 78% täideti vähemalt osaliselt. Täitmata jäi 22% nõuetest, mis oli üldjuhul tingitud seadme vastava funktsionaalsuse puudumisest.

Analüüsi käigus jõuti järeldusele, et kontroller ei sobi ettevõtte levinud projektlahendustesse. Otsust mõjutasid kõige rohkem:

1. Piiratud suurusega FBD programmeerimisala;
2. Simulatsioonikeskkonna puudumine;
3. Vajadus terve programmeerimise protsessi kestel olla kontrolleriga ühenduses;
4. Piiratud mäluressurss.

Seade võib leida potentsiaalset kasutust projektides, kus on tarvis teostada e-kirja teel häireedastust lihtsas tehnoloogilises protsessis või kui põhilist rakendust leiab Node-RED.

FBD ja Node-RED keskkondade kombinatsioon pakkus rohkelt uusi võimalusi, kuid kontrolleri kasutamise soovitamiseks ettevõttes, peab põhirõhk olema suunatud Node-RED rakendusele.

Valminud lõputööd saab kasutada lähteülesande ja juhendmaterjalina ka mõne muu kontrolleri ja riistvara testimiseks. Edasiarendusena tuleks valminud lahendusega ühendada SCADA süsteem ja lähemalt analüüsida Modbus andmesidega seadmete ühendamist valitud kontrolleriga.

Kasutatud kirjandus

[1] Hill S., Chandler S., ja Beaton P. 4G vs. LTE: The Differences Explained, 29.03. 2021. https://www.digitaltrends.com/mobile/4g-vs-lte/ (09.04.2021).
[2] Why CODESYS? https://www.codesys.com/the-system/why-codesys.html (12.05.2021).
[3] e-Teatmik: IT ja sidetehnika seletav sõnaraamat. http://www.vallaste.ee.
[4] AKIT - Andmekaitse ja infoturbe leksikon. https://akit.cyber.ee.
[5] Lehtla T. ja Rosin A. Programmeeritavate kontrollerite tarkvara ja käsustik. http://www.ene.ttu.ee/leonardo/loogika/LOOGS9.pdf (09.04.2021).
[6] Lehtla M. Sissejuhatus digitaaltehnikasse: Programmeeritavate kontrolleritega seotud mõisteid. http://www.tud.ttu.ee/im/Madis.Lehtla/WEB/Sissejuhatus_ digitaaltehnikasse/Sonastikud/programmeeritavad_kontrollerid.html (09.04.2021).
[7] Hack Reactor, What is JavaScript Used For? https://www.hackreactor.com/blog/ what-is-javascript-used-for (09.04.2021).
[8] Node-RED: Nodes. https://nodered.org/docs/user-guide/editor/workspace/nodes (29.04.2021).
[9] Node.js. https://nodejs.org/en/ (29.04.2021).
[10] Unified Architecture. https://opcfoundation.org/about/opc-technologies/opc-ua/ (31.03.2021).
[11] Pt100 sensor. https://www.peaksensors.co.uk/what-is/pt100-sensor/ (31.03.2021).
[12] RJ45 Definition. https://techterms.com/definition/rj45 (10.05.2021).
[13] Phoenix Contact Starter kit. https://www.phoenixcontact.com/online/portal/ lt?uri=pxc-oc-itemdetail:pid=1046568&library=lten&tab=1 (27.04.2021).
[14] SIMATIC S7-1200 Starter Kits. https://assets.new.siemens.com/siemens/assets/api/ uuid:635a624b-a412-45f7-a9f0-6db531ce6974/difa-b10080-00flsimatics7-1200starterkits207x277ohnepreisenus-14.pdf (27.04.2021).
[15] Mehhatroonikaseadmed, Täiturid. https://www.tthk.ee/MEH/Taiturid_1.html (29.03.2021).
[16] Engineering and visualisation tools. https://www.weidmueller.com/int/products/ automation_software/engineering_and_visualisation_tools/index.jsp (18.02.2021).
[17] Weidmüller Product Catalogue, UC20-WL2000-AC. https://catalog.weidmueller.com /catalog/Start.do;jsessionid=005A113FE154336B1BF1C1EE6AA38568?ObjectID=1 334950000&page=Product (06.04.2021).
[18] International Standard - IEC 61131-3 Programmable controllers - Part 3 Programming languages. 2003.
[19] Weidmüller Product Catalogue, UR20-16DI-P. https://catalog.weidmueller.com /catalog/Start.do;jsessionid=87D88C3792A00C5CF014B0ABC84C5942?ObjectID= 1315200000&page=Product (23.02 2021).
[20] Weidmüller Product Catalogue, UR20-16DO-P. https://catalog.weidmueller.com /catalog/Start.do;jsessionid=005A113FE154336B1BF1C1EE6AA38568?ObjectID=1 315250000&page=Product (06.04.2021).
[21] Weidmüller Product Catalogue, UR20-4AI-UI-12. https://catalog.weidmueller.com /catalog/Start.do;jsessionid=87D88C3792A00C5CF014B0ABC84C5942?groupId=(%22group19772348029079%22)&ObjectID=1394390000&page=Product (23.02.2021).
[22] Teltonika RUT955. https://teltonika-networks.com/downloads/en/rut955/RUT955-Datasheet.pdf (09.04.2021).

[23] Knapp E. D. , Langill J. T. Industrial Network Security: Securing Critical Infrastructure Networks for Smart Grid, SCADA, and Other Industrial Control Systems., 2nd edition. USA: Waltham, MA: Elsevier, 2014.
[24] Ramp Up the Security of Industrial Networks. https://www.moxa.com/en/articles/ramp-up-the-security-of-industrial-networks (29.03.2021).
[25] DesRuisseaux D. Practical Overview of Implementing IEC 62443 Security Levels in Industrial Control Applications. https://download.schneider-electric.com/files?p_Doc_Ref=998-20186845 (29.03.2021).
[26] Weidmüller UC20-WL2000-AC help documentation. https://mdcop.weidmueller.com/mediadelivery/asset/900_199492 (20.02.2021).
[27] Controllers U-control Manual. https://mdcop.weidmueller.com/mediadelivery/asset/900_77934 (24.02.2021).
[28] Firmware change log of the device: UC20-WL2000-AC. https://mdcop.weidmueller.com/mediadelivery/asset/900_79182 (24.02.2021).
[29] Function Block Diagram (FBD) Programming Tutorial. https://www.plcacademy.com/function-block-diagram-programming/ (08.03.2021).
[30] Programming style guide for SIMATIC S7-1200/ S7-1500. https://support.industry.siemens.com/cs/attachments/109478084/81318674_Programming_Styleguide_DOC_v20_en.pdf (10.03.2021).
[31] Kester W. ADC Input Noise: The Good, The Bad, and The Ugly. Is No Noise Good Noise?, *Analog Dialogue*, veebr 2006, https://www.analog.com/en/analog-dialogue/articles/adc-input-noise.html# (04.05.2021)
[32] SIMATIC S7-1200 Easy Book. https://euroec.by/assets/files/siemens/s71200_easy_book_en-US_en-US.pdf (02.04.2021).
[33] RTD Temperature vs. Resistance Table. https://web.mst.edu/~cottrell/ME240/Resources/Temperature/RTD%20table.pdf (25.04.2021).
[34] Formula for 4mA-20mA Analog Scaling. https://support.industry.siemens.com/tf//WW/en/posts/formula-for-4ma-20ma-analog-scaling/167143?page=0&pageSize=10 (24.04.2021).
[35] Temperature transmitter Pixsys 2000.35.010. http://www.pixsys.net/en/products/signal-converters/temperature-transmitter-din-head (04.05.2021).
[36] Moore M. What is Industry 4.0? https://www.techradar.com/news/what-is-industry-40-everything-you-need-to-know (29.04.2021).
[37] Node-RED. https://nodered.org/ (29.03.2021).
[38] node-red-contrib-switch-break. http://flows.nodered.org/node/node-red-contrib-switch-break (29.03.2021).
[39] node-red-node-email. http://flows.nodered.org/node/node-red-node-email (11.05.2021).
[40] Nodemailer: SMTP transport. https://nodemailer.com/smtp/#tls-options (11.05.2021).

Lisad

I Kasutuslood

Tegevjuhi poolt esitatud nõuded:

1. Mina kui tegevjuht tahan kompaktset stendi, et see ei võtaks kontoris palju ruumi.
2. Mina kui tegevjuht tahan mobiilset stendi, et seda oleks ühel inimesel lihtne transportida.
3. Mina kui tegevjuht tahan seinale kinnitatavat stendi, et see ei võtaks kontoris põranda või lauaruumi.
4. Mina kui tegevjuht tahan laua peal kasutatavat stendi, et seda oleks võimalik esitada kliendi juures või messil.
5. Mina kui tegevjuht tahan ohutut stendi, et selle kasutamine ei kujutaks ohtu inimestele ja seadmetele.
6. Mina kui tegevjuht tahan stendi kasutada töötaja koolitamiseks, et integreerida töötaja kiiremini ettevõtte tootmisprotsessi.
7. Mina kui tegevjuht tahan, et stend annaks ülevaate ettevõttes kasutatavatest tehnoloogiatest, et integreerida töötaja kiiremini ettevõtte tootmisprotsessi.

Automaatikainseneri poolt esitatud nõuded:

8. Mina kui automaatikainsener tahan ohutut stendi, et selle kasutamine ei kujutaks ohtu inimestele ja seadmetele.
9. Mina kui automaatikainsener tahan stendis lihtsalt seadmeid vahetada, et kiirendada testimisprotsessi.
10. Mina kui automaatikainsener tahan, et seadmete juhtmed oleks tähistatud, et ei tekiks seadmete vahetamisel arusaamatusi.
11. Mina kui automaatikainsener tahan, et kontrolleril oleks Ethernet TCP/IP võrguliides, et seda saaks ühendada internetiga.
12. Mina kui automaatikainsener tahan, et kontrolleris kasutatav programmeerimiskeel vastaks standardile IEC 61131-3, et sinna saaks luua programmi vastavalt vajalikule funktsionaalsusele.
13. Mina kui automaatikainsener tahan, et süsteem mõõdaks analoog signaali, et simuleerida programmilõike mis vajavad analoogsisendit.
14. Mina kui automaatikainsener tahan, et süsteem võimaldaks mõõta 4…20mA signaali, et kasutada vastavat analoogandurit testimis ja arendustöös.
15. Mina kui automaatikainsener tahan, et süsteem võimaldaks mõõta 0…10V signaali, et kasutada vastavat analoogandurit testimis ja arendustöös.
16. Mina kui automaatikainsener tahan, et süsteem mõõdaks digitaalseid mõõteväärtusi, et simuleerida programmilõike, mis vajavad diskreetset sisendit „1“ või „0“.
17. Mina kui automaatikainsener tahan, et kontroller edastaks alarme e-maili teel, et stendi kasutamisega ei tekiks lisakulu SIM kaardi näol.
18. Mina kui automaatikainsener tahan, et kontroller kasutaks Modbus TCP/IP andmesidet kolmanda seadmega, et testida seadme funktsionaalsust.

SCADA inseneri poolt esitatud nõuded

19. Mina kui SCADA insener tahan, et kasutajaliides kuvaks kõiki alarme, et omada ülevaadet süsteemi riketest.
20. Mina kui SCADA insener tahan, et saaks muuta alarmi seadeväärtusi kasutajaliidese abil, et süsteemi paremini häälestada.
21. Mina kui SCADA insener tahan, et keskkond võimaldaks erinevate kasutajate loomist, et iga kasutaja oleks tuvastatav.

22. Mina kui SCADA insener tahan, et keskkond võimaldaks erinevate kasutaja rollide loomist, et kasutaja saaks teha ainult tema rollile ette nähtud tegevusi.
23. Mina kui SCADA insener tahan, et keskkond võimaldaks raportite koostamist, et kasutaja saaks olulisi kokkuvõtteid süsteemist.
24. Mina kui SCADA insener tahan, et süsteem võimaldaks alarmide edastamist e-mailile, et paremini hallata alarmide edastust.
25. Mina kui SCADA insener tahan, et kõik alarmid oleks kasutajaliidese avalehel näha, et saada koheselt ülevaade kui süsteemis on rike.
26. Mina kui SCADA insener tahan, et kasutajaliidesest saaks muuta e-kirja adressaati, et paremini hallata alarmide edastust.
27. Mina kui SCADA insener tahan, et alarme saaks kasutajaliidesest välja lülitada, et paremini hallata üksikuid alarme.
28. Mina kui SCADA insener tahan, et kasutajaliidesest saaks e-kirja edastuse välja lülitada, et paremini juhtida häireedastust.

II Kontrolleri UC20-WL200-AC spetsifikatsioon

Technical data

Dimensions and weights

Depth	76 mm	Depth (inches)	2.992 inch
Height	120 mm	Height (inches)	4.724 inch
Mounting dimension - height	128 mm	Net weight	232 g
Width	52 mm	Width (inches)	2.047 inch

Connection data

Type of connection	PUSH IN

General data

Rail	TS 35	UL 94 flammability rating	V-0

Power supply

Current consumption from I_{IN} (power segment of the field bus coupler), typ.	116 mA	
Current consumption from I_{IN} (power segment of the fieldbus coupler), typ.	max.	116 mA
	min.	0 mA
	nominal	116 mA
Feed current for I_{IN} (input current path)	max.	5,000 mA
	nominal	5,000 mA
	min.	0 mA
Feed current for I_{IN} (input current path), max.	5 A	
Feed current for I_{OUT} (output current path)	max.	5,000 mA
	nominal	5,000 mA
	min.	0 mA
Feed current for I_{OUT} (output current path), max.	5 A	
Feed current for the system, max.	4,000 mA	
Supply voltage for outputs	24 V DC +20 %/-15 %	
Supply voltage system and inputs	24 V DC +20 %/-15 %	

System data

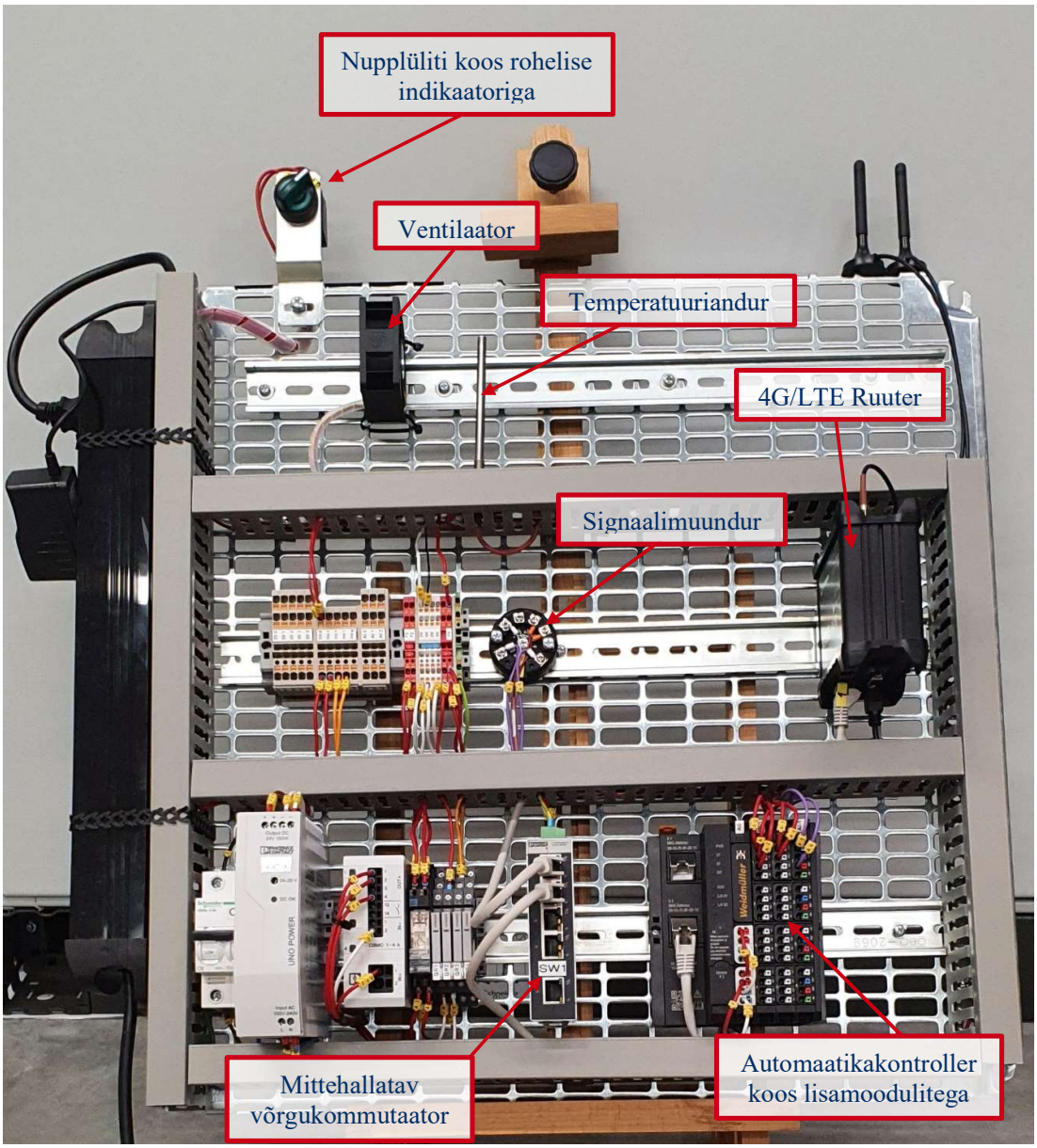
Configuration interface	Micro USB 2.0	Connection type	2 x RJ45 plug-in connectors
Engineering tool	u-create web	Interface	2x Ethernet TCP/IP, 1x Micro USB
Memory (Flash)	4 GB, 32 GB via microSD	Module type	Controller
Processor	Dual Core ARM Cortex A9, 624 MHz, 512 Mbyte RAM	Programming environment	web-based
Real-time clock	Battery buffered	Transmission speed backplane bus, max.	48 Mbit
max. number of modules	64		

Classifications

ETIM 6.0	EC001603	ETIM 7.0	EC001603
ECLASS 9.1	27-24-26-08	ECLASS 10.0	27-24-26-07
ECLASS 11.0	27-24-26-07		

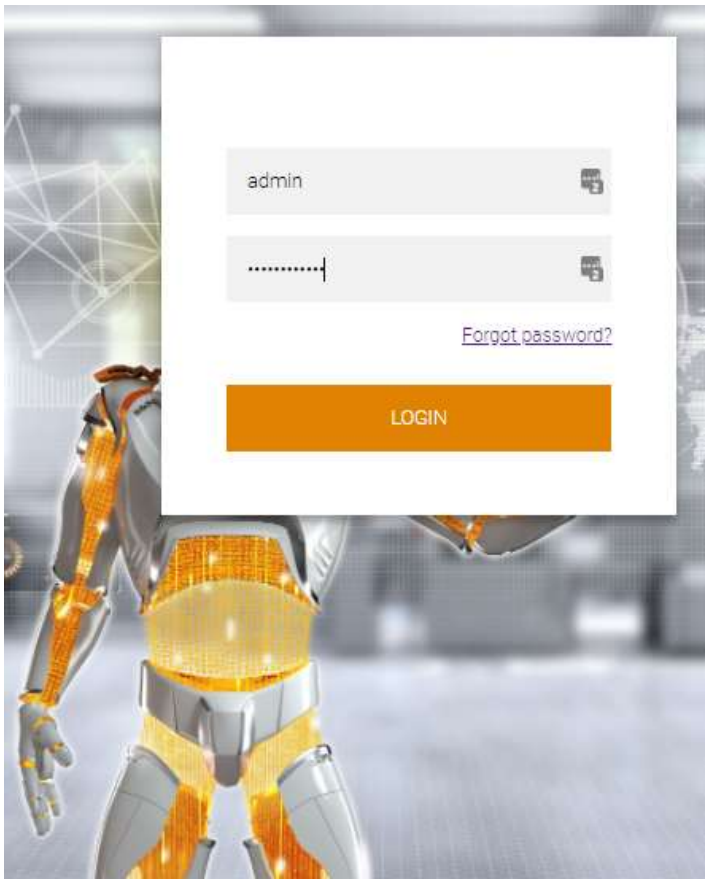
Joonis II.1. Kontrolleri spetsifikatsioon

III Automaatjuhtimissüsteemi testimise stend

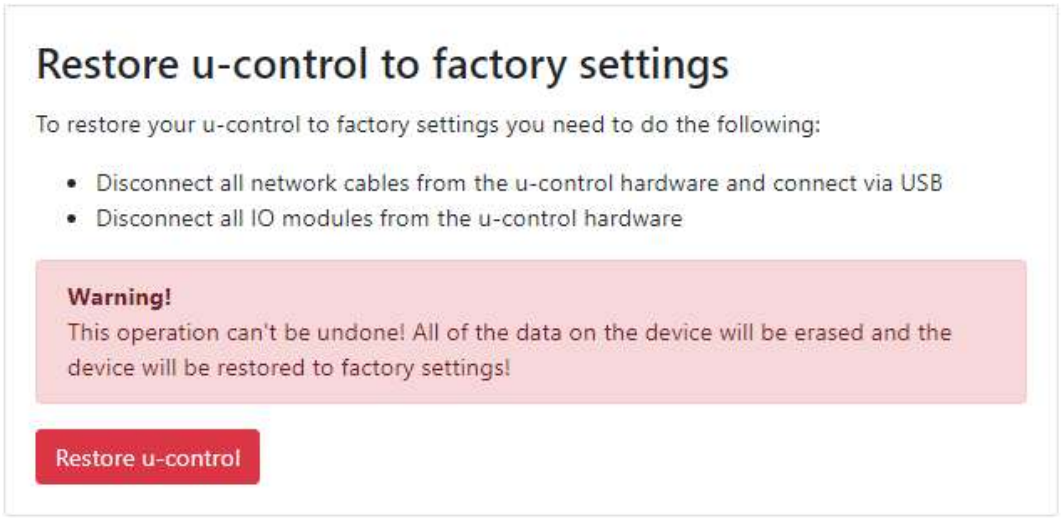


Joonis III.1. Automaatjuhtimissüsteemi testimise stend

IV Kasutaja tuvastus ja tehaseseadete taastamise juhend

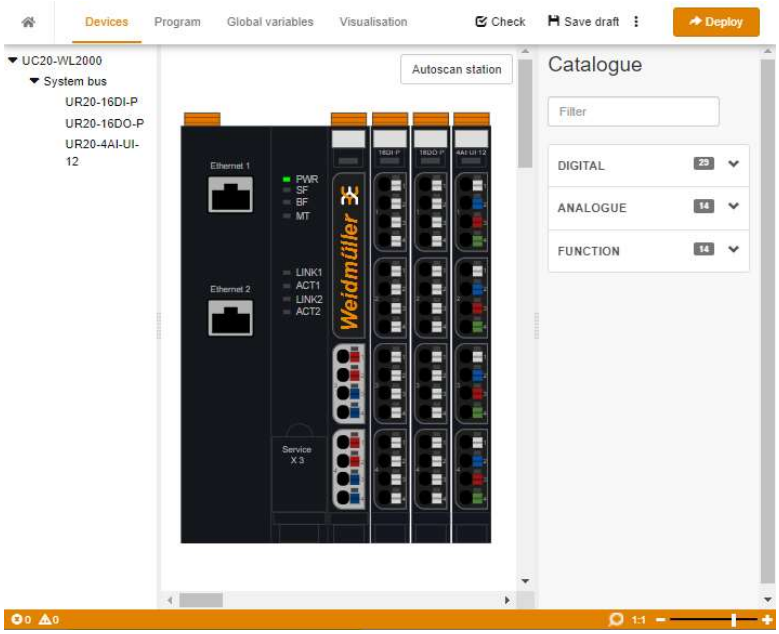


Joonis IV.1. Kasutaja tuvastus

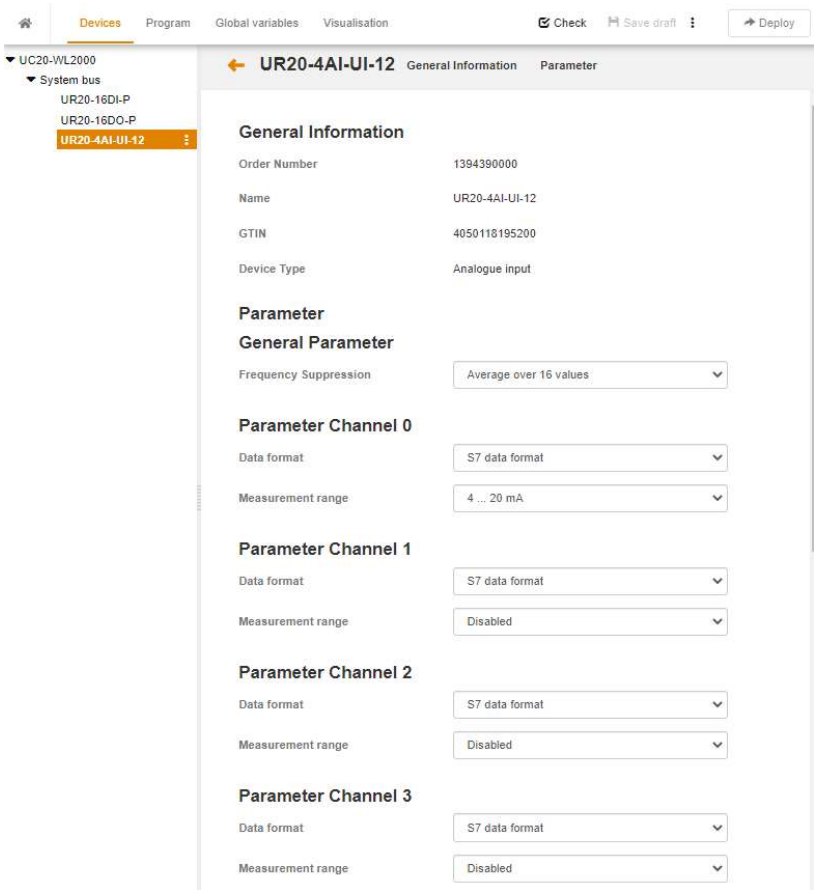


Joonis IV.2. Tehaseseadete taastamise juhend

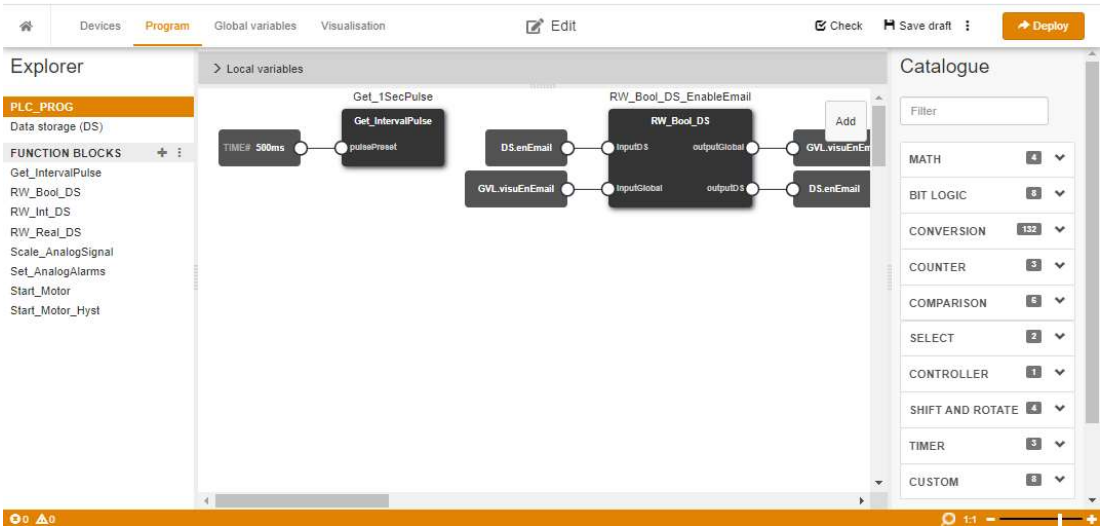
V Arenduskeskkondade töölehed



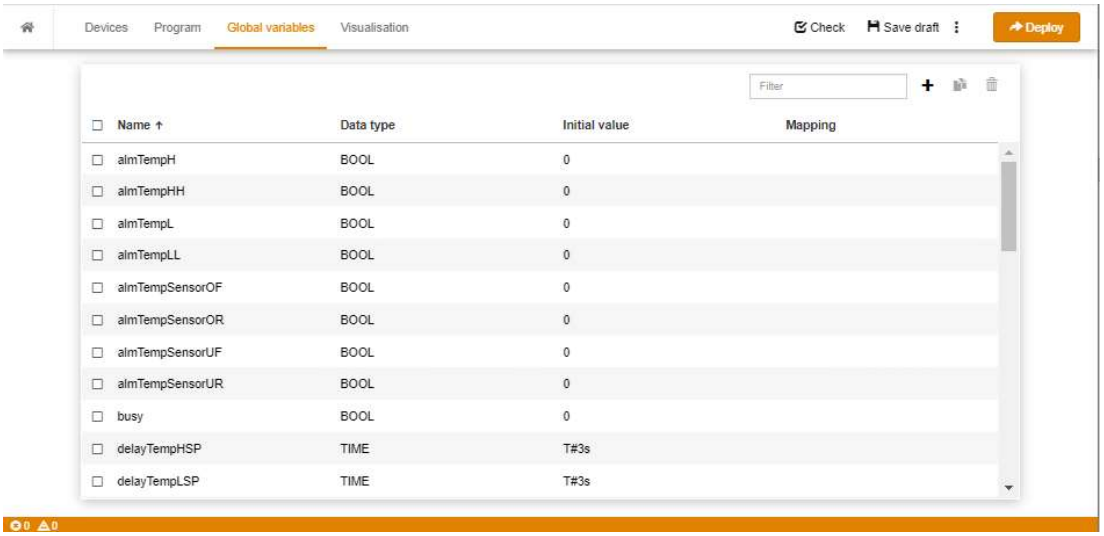
Joonis V.1. FBD programmi arenduskeskkonna seadmete alamleht redigeerimisvaates



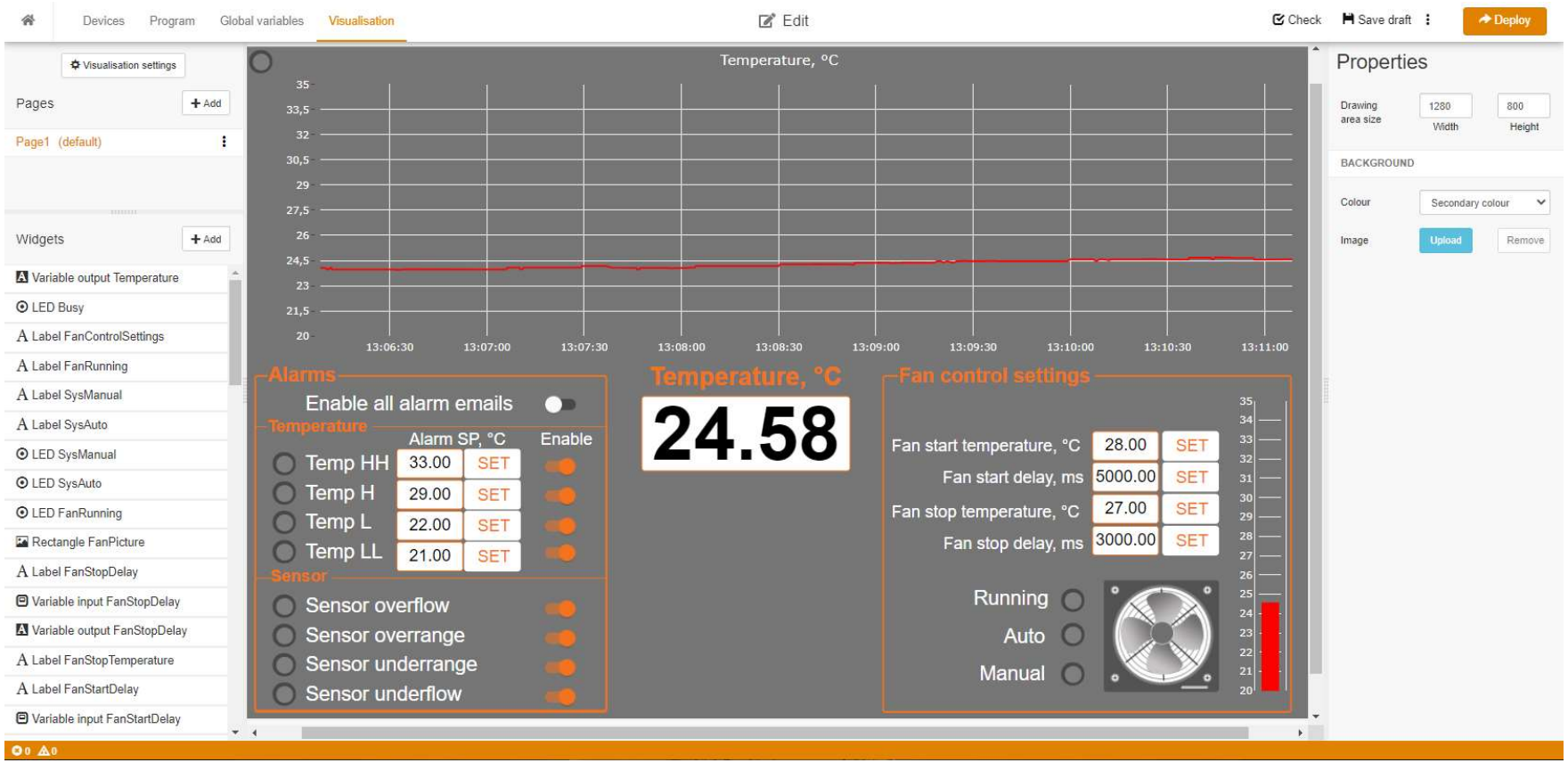
Joonis V.2. FBD arenduskeskkonna seadmete analoogmooduli alamleht redigeerimisvaates



Joonis V.3. FBD arenduskeskkonna programmi alamleht redigeerimisvaates



Joonis V.4. FBD arenduskeskkonna globaalsete muutujate alamleht redigeerimisvaates

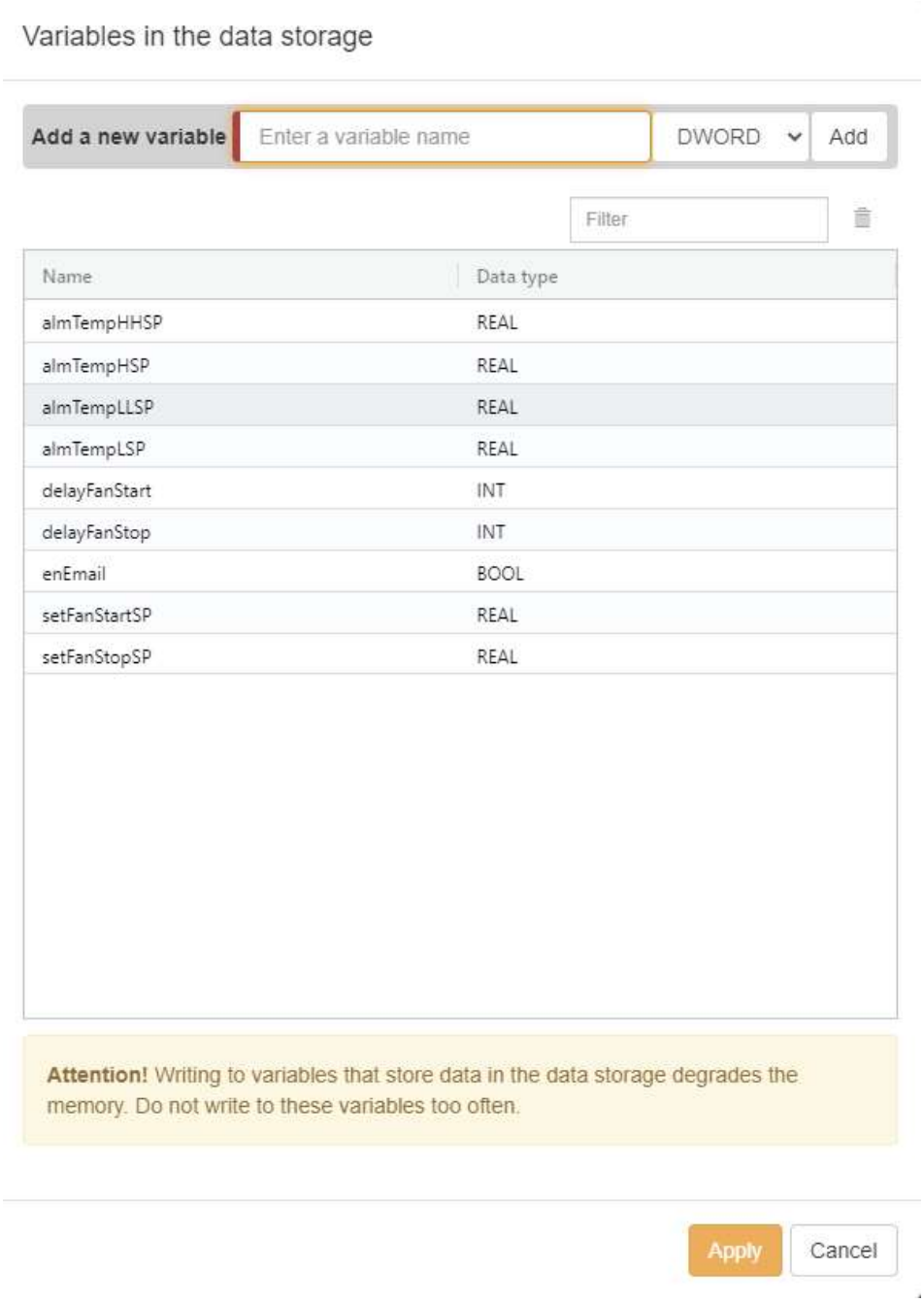


Joonis V.5. Visualiseerimiskeskkond redigeerimisvaates

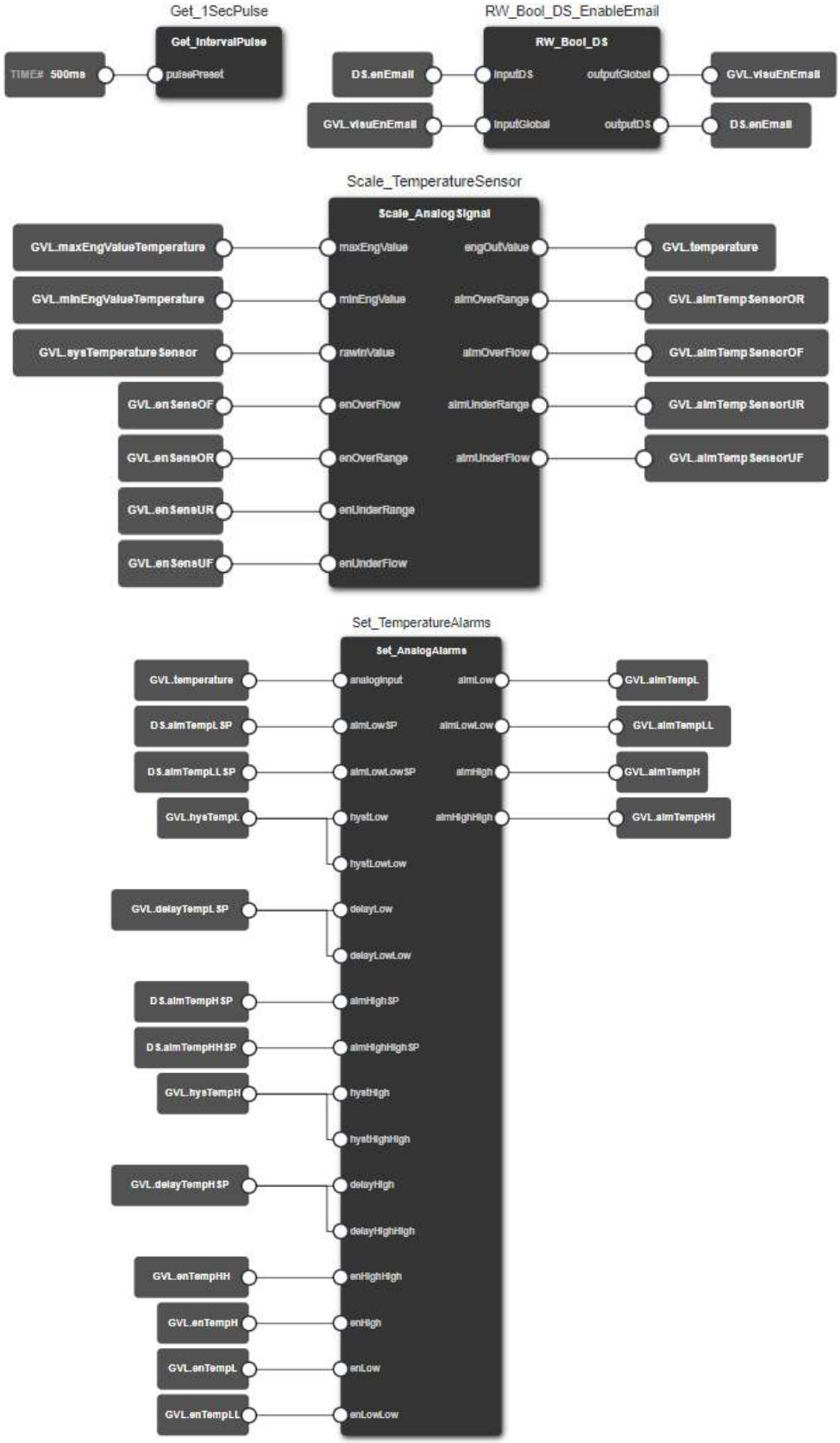
VI Kontrolleri täielik FBD programm

Name ↑	Data type	Initial value	Mapping
almTempH	BOOL	0	
almTempHH	BOOL	0	
almTempL	BOOL	0	
almTempLL	BOOL	0	
almTempSensorOF	BOOL	0	
almTempSensorOR	BOOL	0	
almTempSensorUF	BOOL	0	
almTempSensorUR	BOOL	0	
busy	BOOL	0	
delayTempHSP	TIME	T#3s	
delayTempLSP	TIME	T#3s	
enSensOF	BOOL	1	
enSensOR	BOOL	1	
enSensUF	BOOL	1	
enSensUR	BOOL	1	
enTempH	BOOL	1	
enTempHH	BOOL	1	
enTempL	BOOL	1	
enTempLL	BOOL	1	
fanRunning	BOOL	0	UR20-16DI-P@1 channel_2
hysTempH	REAL	1	
hysTempL	REAL	1	
maxEngValueTemperature	INT	50	
minEngValueTemperature	INT	0	
pulse1Sec	BOOL	0	
qFanStartCmd	BOOL	0	UR20-16DO-P@2 channel_0
sysModeAuto	BOOL	0	UR20-16DI-P@1 channel_0
sysModeManual	BOOL	0	UR20-16DI-P@1 channel_1
sysTemperatureSensor	INT	0	UR20-4AI-UI-12@3 channel_0
temperature	REAL	0	
visuAlmTempHHSP	REAL	0	
visuAlmTempHSP	REAL	0	
visuAlmTempLLSP	REAL	0	
visuAlmTempLSP	REAL	0	
visuDelayFanStart	INT	1000	
visuDelayFanStop	INT	3000	
visuEnEmail	BOOL	0	
visuSetFanStartSP	REAL	0	
visuSetFanStopSP	REAL	0	

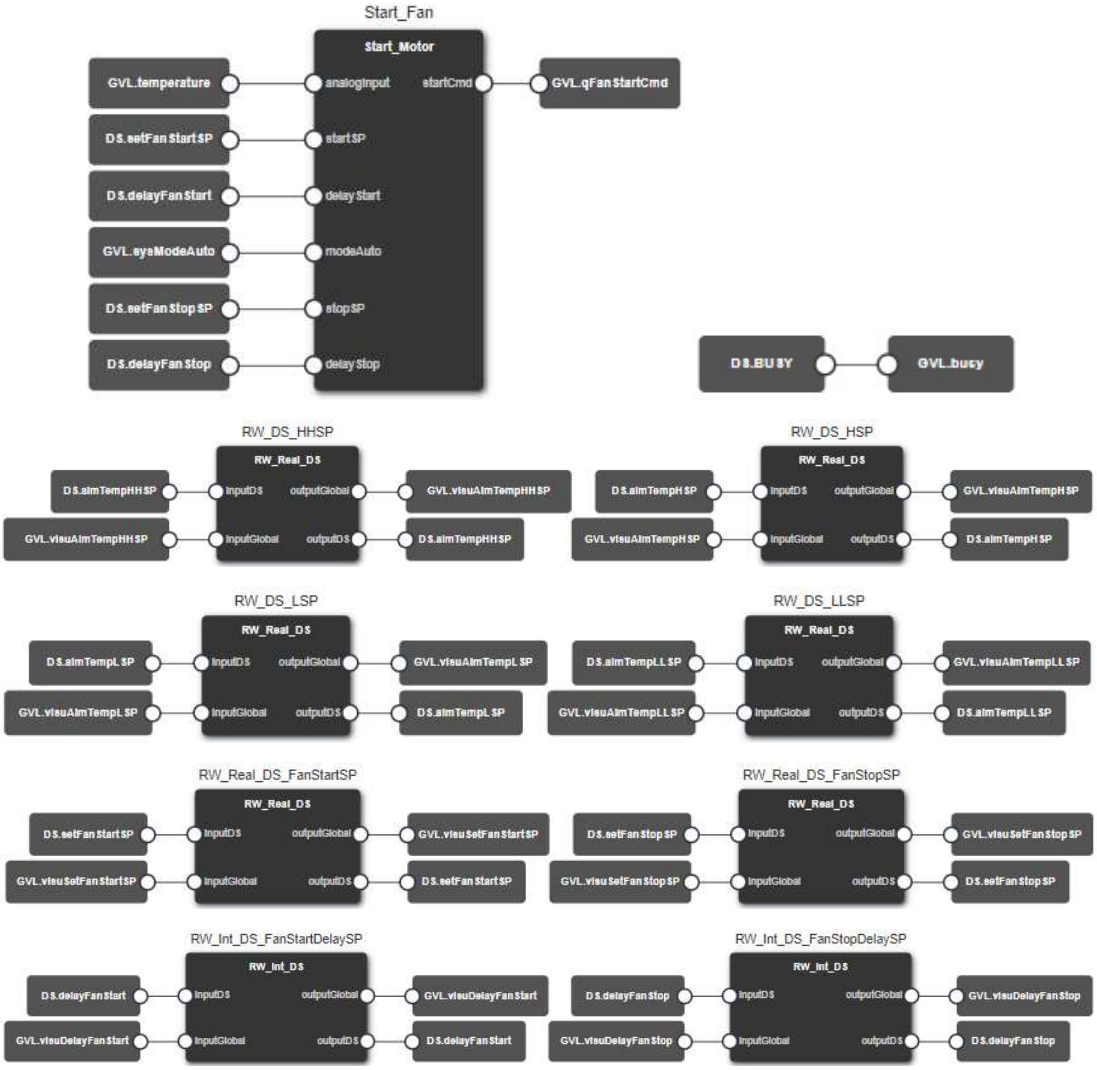
Joonis VI.1. Globaalsed muutujad



Joonis VI.2. Püsimälu muutujad



Joonis VI.3. Funktsiooniplokkskeem 1



Joonis VI.3 järg

VII Analoogsignaali töötluse funktsiooniplokk “Scale_AnalogSignal”

Local variables

Filter

Name	Section	Data type	Initial value
MAX_RAW_VALUE	VAR	INT	27648
MIN_RAW_VALUE	VAR	INT	0
maxEngValue	VAR_INPUT	INT	0
minEngValue	VAR_INPUT	INT	0
engOutValue	VAR_OUTPUT	REAL	0
rawInValue	VAR_INPUT	INT	0
OVER_RANGE	VAR	INT	32511
UNDER_RANGE	VAR	INT	-4864
almOverRange	VAR_OUTPUT	BOOL	0
almOverFlow	VAR_OUTPUT	BOOL	0
almUnderRange	VAR_OUTPUT	BOOL	0
almUnderFlow	VAR_OUTPUT	BOOL	0
ALM_DELAY	VAR	TIME	T#5s
TON_O1	VAR	TON	
TON_O2	VAR	TON	
TON_U1	VAR	TON	
TON_U2	VAR	TON	
enOverFlow	VAR_INPUT	BOOL	0
enOverRange	VAR_INPUT	BOOL	0
enUnderRange	VAR_INPUT	BOOL	0
enUnderFlow	VAR_INPUT	BOOL	0

Joonis VII.1. Funktsiooniploki „Scale_AnalogSignal“ muutujad

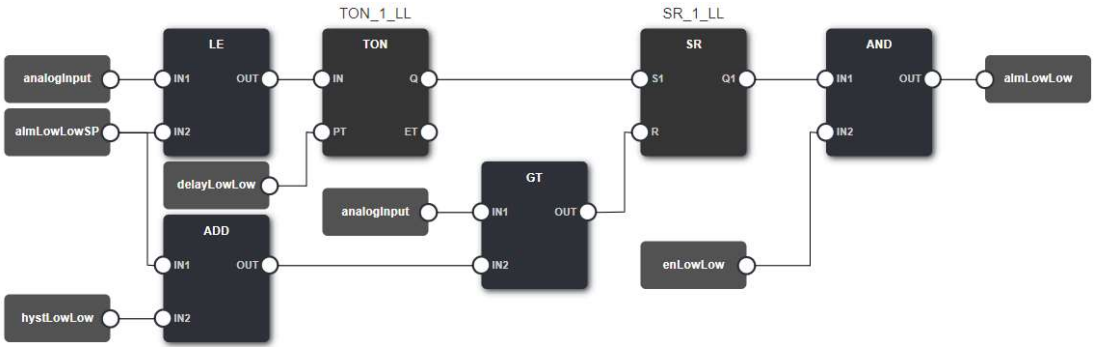
Joonis VII.2. Funktsiooniploki „Scale_AnalogSignal“ programm

VIII Analoogalarmi funktsiooniplokk “Set_AnalogAlarms”

Local variables — Filter

Name	Section	Data type	Initial value
analogInput	VAR_INPUT	REAL	0
almLowSP	VAR_INPUT	REAL	0
almLowLowSP	VAR_INPUT	REAL	0
hystLow	VAR_INPUT	REAL	0
hystLowLow	VAR_INPUT	REAL	0
delayLow	VAR_INPUT	TIME	T#3s
delayLowLow	VAR_INPUT	TIME	T#3s
almHighSP	VAR_INPUT	REAL	0
almHighHighSP	VAR_INPUT	REAL	0
hystHigh	VAR_INPUT	REAL	0
hystHighHigh	VAR_INPUT	REAL	0
delayHigh	VAR_INPUT	TIME	T#3s
delayHighHigh	VAR_INPUT	TIME	T#3s
almLow	VAR_OUTPUT	BOOL	0
almLowLow	VAR_OUTPUT	BOOL	0
almHigh	VAR_OUTPUT	BOOL	0
almHighHigh	VAR_OUTPUT	BOOL	0
SR_1_LL	VAR	SR	
TON_1_LL	VAR	TON	
TON_1_L	VAR	TON	
SR_1_L	VAR	SR	
TON_1_HH	VAR	TON	
SR_1_HH	VAR	SR	
TON_1_H	VAR	TON	
SR_1_H	VAR	SR	
enHighHigh	VAR_INPUT	BOOL	1
enHigh	VAR_INPUT	BOOL	1
enLow	VAR_INPUT	BOOL	1
enLowLow	VAR_INPUT	BOOL	1

Joonis VIII.1. Funktsiooniploki „Set_AnalogAlarms“ muutujad



Joonis VIII.2. Funktsiooniploki „Set_AnalogAlarms“ programmilõik

Joonis VIII.2 järg

IX Ventilaatori juhtimise funktsiooniplokk “Start_Fan” muutujad

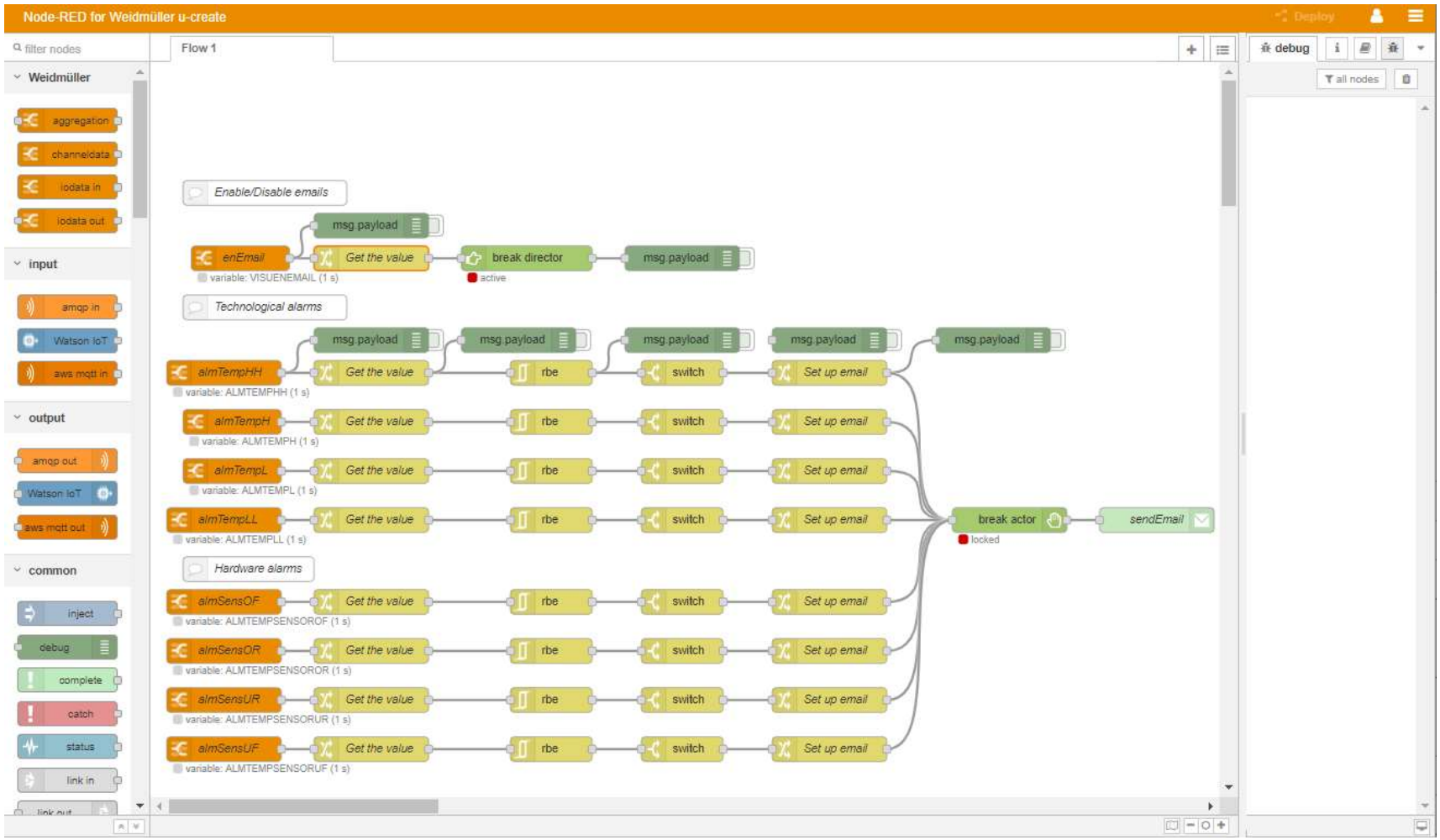
Local variables

Filter

Name	Section	Data type	Initial value
analogInput	VAR_INPUT	REAL	0
startSP	VAR_INPUT	REAL	0
delayStart	VAR_INPUT	INT	0
modeAuto	VAR_INPUT	BOOL	0
startCmd	VAR_OUTPUT	BOOL	0
stopSP	VAR_INPUT	REAL	0
delayStop	VAR_INPUT	INT	0
TON_1	VAR	TON	
TON_2	VAR	TON	
RS	VAR	RS	

Joonis IX.1. Funktsiooniploki „Start_Fan“ muutujad

X Alarmide töötlus Node-RED keskkonnas



Joonis X.1. Node-RED keskkond ja programm

XI Kasutajaliidese esilehel kuvatud muutujad

Tabel XI.1. Kasutajaliidese muutujad

Muutuja	Kirjeldus	Funktsioon
Kahendmuutujad (*Boolean*)		
almTempHH	Temperatuur kõrge kriitiline	R
almTempH	Temperatuur kõrge hoiatus	R
almTempL	Temperatuur madal hoiatus	R
almTempLL	Temperatuur madal kriitiline	R
almTempSensorOF	Sisend ületäitunud	R
almTempSensorOR	Sisend üle mõõtepiirkonna	R
almTempSensorUR	Sisend alla mõõtepiirkonna	R
almTempSensorUF	Sisend alatäitunud	R
fanRunning	Ventilaator töös	R
sysModeAuto	Süsteem automaatjuhtimisel	R
sysModeManual	Süsteem käsijuhtimisel	R
visuEnEmail	Alarmi e-kirjade edastuse lubamine (keelamine)	R/W
enTempHH	Temperatuur kõrge kriitiline alarmi desaktiveerimine	R/W
enTempH	Temperatuur kõrge hoiatus alarmi desaktiveerimine	R/W
enTempL	Temperatuur madal hoiatus alarmi desaktiveerimine	R/W
enTempLL	Temperatuur madal kriitiline alarmi desaktiveerimine	R/W
enSensorOF	Sisend ületäitunud alarmi desaktiveerimine	R/W
enSensorOR	Sisend üle mõõtepiirkonna alarmi desaktiveerimine	R/W
enSensorUR	Sisend alla mõõtepiirkonna alarmi desaktiveerimine	R/W
enSensorUF	Sisend alatäitunud alarmi desaktiveerimine	R/W
busy	Püsimällu kirjutamine hõivatud	R
Ujukoma muutujad (*Float*), ühik		
temperature	Temperatuur, °C	R
visuAlmTempHHSP	Temperatuur kõrge kriitiline seadeväärtus, °C	R/W
visuAlmTempHSP	Temperatuur kõrge hoiatus seadeväärtus, °C	R/W
visuAlmTempLSP	Temperatuur madal hoiatus seadeväärtus, °C	R/W
visuAlmTempLLSP	Temperatuur madal kriitiline seadeväärtus, °C	R/W
visuSetFanStartSP	Ventilaator start temperatuuri seadeväärtus, °C	R/W
visuSetFanStopSP	Ventilaator stopp temperatuuri seadeväärtus, °C	R/W
Täisarvulised muutujad (*Integer*), ühik		
visuDelayFanStart	Ventilaator start viide seadeväärtus, ms	R/W
visuDelayFanStop	Ventilaator stopp viide seadeväärtus, ms	R/W

XII Funktsiooniploki “RW_Real_DS” muutujad

Local variables

Filter

Name	Section	Data type	Initial value
startupBool	VAR	BOOL	1
outputGlobal	VAR_OUTPUT	REAL	0
inputDS	VAR_INPUT	REAL	0
RS_1	VAR	RS	
output	VAR	REAL	0
inputGlobal	VAR_INPUT	REAL	0
outputDS	VAR_OUTPUT	REAL	0
fromVisu	VAR	REAL	0
outputNE	VAR	BOOL	0

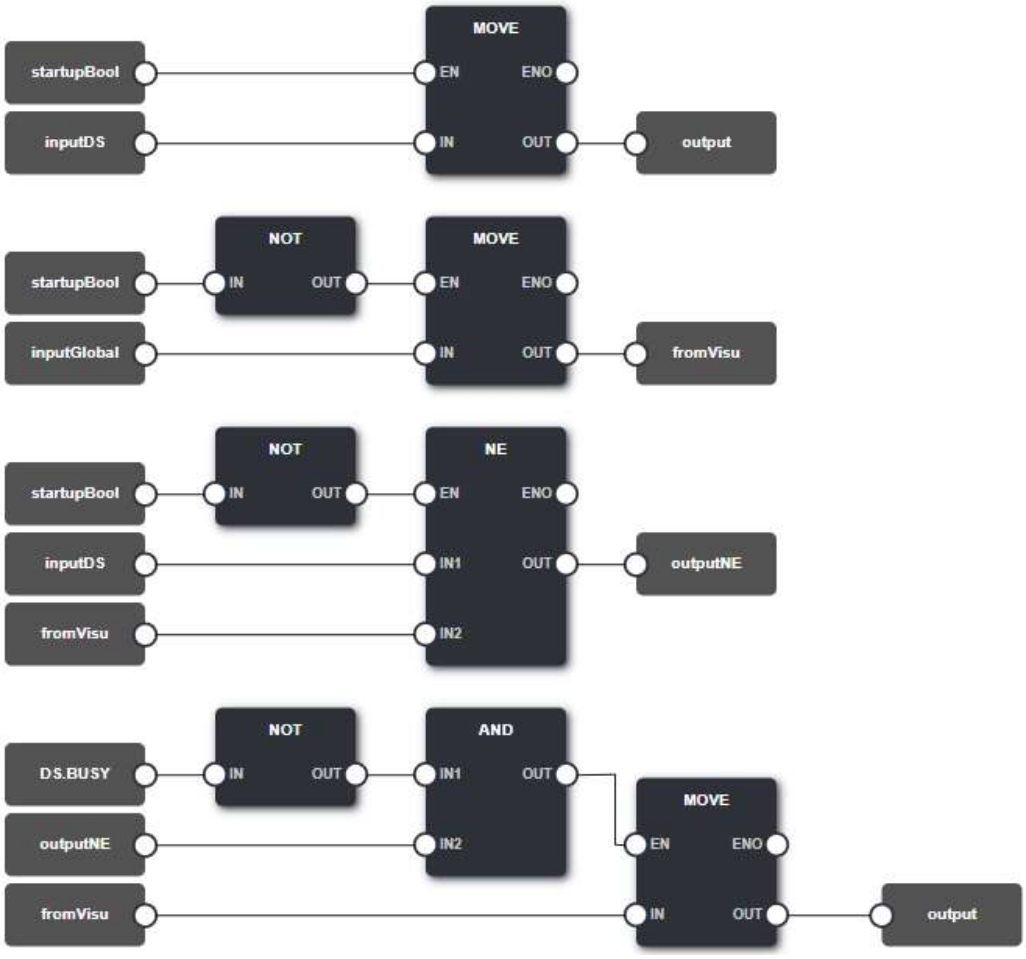
Joonis XII.1. Funktsiooniploki „RW_Real_DS“ muutujad

XIII Funktsiooniplokk „RW_Int_DS“ muutujad ja programmilõik

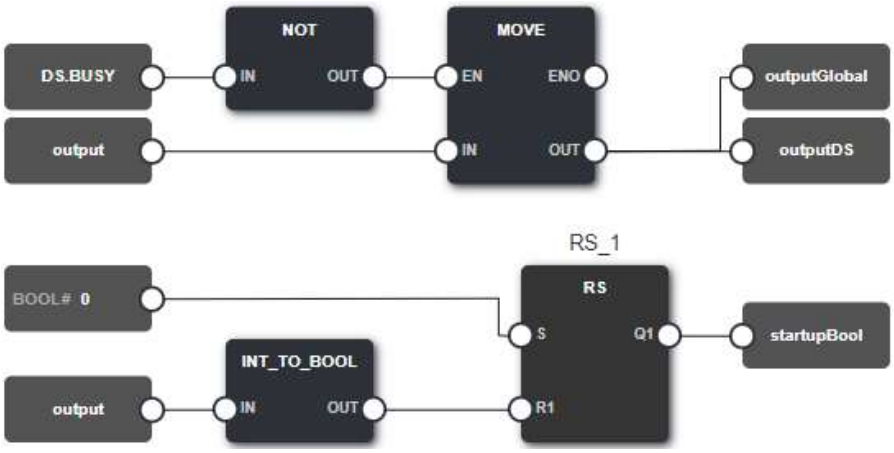
Local variables | Filter

Name	Section	Data type	Initial value
inputDS	VAR_INPUT	INT	0
startupBool	VAR	BOOL	1
output	VAR	INT	0
outputGlobal	VAR_OUTPUT	INT	0
outputDS	VAR_OUTPUT	INT	0
fromVisu	VAR	INT	0
inputGlobal	VAR_INPUT	INT	0
outputNE	VAR	BOOL	0
RS_1	VAR	RS	

Joonis XIII.1. Funktsiooniploki „RW_Int_DS“ muutujad



Joonis XIII.2. Funktsiooniploki „RW_Int_DS“ programm



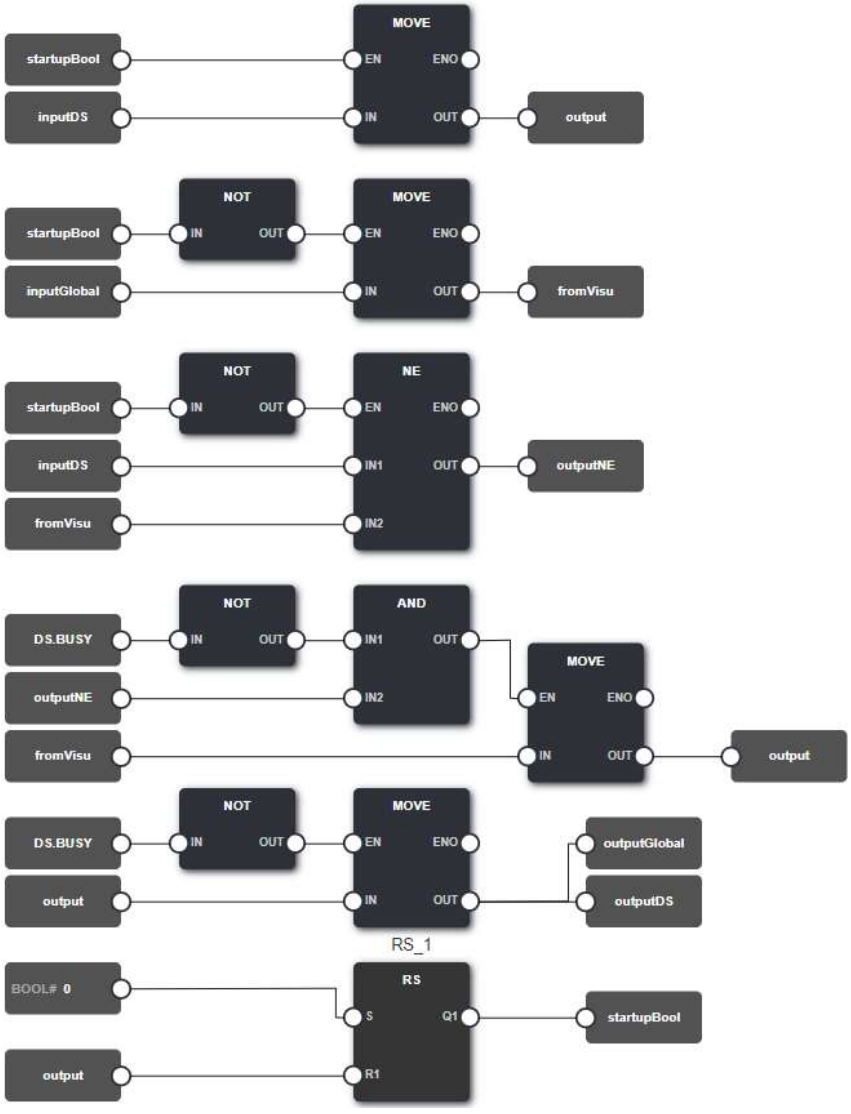
Joonis XIII.2 järg

XIV Funktsiooniplokk „RW_Bool_DS“ muutujad ja programmilõik

Local variables — Filter

Name	Section	Data type	Initial value
inputDS	VAR_INPUT	BOOL	0
startupBool	VAR	BOOL	1
output	VAR	BOOL	0
outputGlobal	VAR_OUTPUT	BOOL	0
outputDS	VAR_OUTPUT	BOOL	0
fromVisu	VAR	BOOL	0
inputGlobal	VAR_INPUT	BOOL	0
outputNE	VAR	BOOL	0
RS_1	VAR	RS	

Joonis XIV.1. Funktsiooniploki „RW_Bool_DS“ muutujad



Joonis XIV.2. Funktsiooniploki „RW_Bool_DS“ programm

XV FBD programmi töölehe ruumiline kasutus

Joonis XV.1. Programmi ruumikasutus töölehel

XVI Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Indrek Pomerants,

(autori nimi)

annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose

Automaatjuhtimissüsteemi testimine kontrolleri UC20-WL2000-AC näitel,

(lõputöö pealkiri)

mille juhendajad on Ivar Koppel ja Kaido Jaanus,

(juhendajate nimed)

reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni.

Annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni.

Olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile.

Kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Indrek Pomerants

14.05.2021