Tallinna Tehnikaülikool
Elektriajamite ja jõuelektroonika instituut
Eesti Moritz Hermann Jacobi Selts

ARGO ROSIN

PROGRAMMEERITAVAD KONTROLLERID SIMATIC S7

Tallinn
2000

Kaane kujundanud Mart Peri

ISBN 9985-69-018-4

Saateks

Viimasel aastakümnel on tootmise automatiseerimine Eestis oluliselt arenenud. Sellele on aidanud kaasa mitmed firmad, kes vahendavad Eestisse nüüdisaegseid automaatikavahendeid ja tegelevad tootmise automatiseerimisega. Eriti tähtsa koha on automaatikavahendite hulgas omandanud programmeeritavad kontrollerid (Programmable Logic Controller, PLC), mis võimaldavad tootmisprotsesse juhtida, tootmisseadmeid kontrollida, diagnoosida ja teiste seadmetega koordineerida. Programmeeritavate kontrollerite eeliseks võrreldes jäiga skeemiautomaatikaga (nt. relee- või loogikaskeemidega) on paindlikkus, paigaldamise ja seadistamise lihtsus, suur töökindlus ning majanduslik tasuvus. Programmeeritavate kontrollerite funktsioone saab vabalt valida ja kombineerida protsessijuhtimisele orienteeritud intelligentsete moodulitega, neil on standardsed kommunikatsiooniliidesed ja nende programmeerimiskeeled on standardiseeritud (IEC 61131-3).

Antud raamat on sel alal esimene eestikeelne ja mõeldud Tallinna Tehnikaülikooli tugevvoolutehnika üliõpilastele loengute, harjutustundide ja praktikumide läbiviimiseks aines *Tootmise automatiseerimine*; ka kasutatakse seda inseneride täienduskoolituskursuse "Programmeeritavad tööstuskontrollerid" abimaterjalina.

Raamatus antakse ülevaade tootmise automatiseerimise põhimõistetest, riist- ja tarkvarast. Põhiliselt käsitletakse firma SIEMENS programmeeritavate kontrollerite SIMATIC S7 ehitust ja kasutamist tootmise automatiseerimiseks ning tarkvarapaketti STEP7. Kuna eri firmade kontrollerite tööpõhimõte ja programmeerimine on standardiseeritud (IEC 61131), siis antud raamatus omandatud teadmisi võib hõlpsasti rakendada mõne teise firma kontrolleri tundmaõppimisel.

Käesoleva raamatu koostamisel andsid kasulikke soovitusi ja tegid märkusi professorid Endel Risthein, Tõnu Lehtla ja Juhan Laugis ning dotsent Jaan Tomson Tallinna Tehnikaülikoolist ja firma SIEMENS AS töötajad Ats Alupere ja Harri Koppelmaa.

Autor

Sisukord

1 TOOTMISE ARENG JA AUTOMATISEERIMINE 6

1.1 Tootmise tehnoloogilise arengu põhietapid 6

1.2 Automaatliinid ja paindtootmissüsteemid 10

1.3 Tootmise hierarhiline juhtimine 14

1.4 Andmesidevõrgud ja -liidesed 17

1.5 Lokaalsed süsteemid ja nende elemendid 19

2 PROGRAMMEERITAVATE KONTROLLERITE RIISTVARA 21

2.1 Ehitus 21

2.2 Tööpõhimõte 24

2.3 Arengutendentsid 27

3 PROGRAMMEERITAVATE KONTROLLERITE TARKVARA JA PROGRAMMEERIMINE 28

3.1 Programmeerimiskeeled ja -viisid 29

3.2 Programmeerimiskeel STEP 7 33
3.2.1 Programmeerimise alused 33
3.2.2 Käsurea struktuur 36
3.2.3 Andmetüübid ja -vormingud 38
3.2.4 Käsustik 39
3.2.5 Loogikakäsud 40
3.2.6 Sulguvad ja avanevad kontaktid 42
3.2.7 NING-VÕI-loogikaühend 44
3.2.8 VÕI-NING-loogikaühend 45
3.2.9 Väljundoperandi kasutamine sisendoperandina 46
3.2.10 Mälufunktsioonid 47
3.2.11 Impulsi tõusev ja langev front 50
3.2.12 Laadimis- ja siirdamiskäsud 51
3.2.13 Võrdluskäsud 52
3.2.14 Loenduskäsud 53
3.2.15 Viivituskäsud 55
3.2.16 Aritmeetika-, teisendus-, nihke- ja loogikatehete käsud 58
3.2.17 Programmi struktuurikäsud 60
3.2.18 Andmeploki talitluskäsud 62
3.2.19 Sammprogrammjuhtimine 63
3.2.20 Analoogsignaalid ja pidevjuhtimine 65

3.3 Reguleerimistarkvara 68
3.3.1 Reguleerimise alused 68
3.3.2 Reguleerimistarkvara SIMATIC S7/M7/C7 kontrolleritele 70
3.3.3 PID- ja hägusjuhtimine 74

4 PROTSESSIJUHTIMISPROGRAMMI KOOSTAMINE 78

4.1 Programmeerimiseks ettevalmistamine 78

4.2 Programmeerimine 80

4.3 Programmi testimine ja protsessi diagnostika 80

5 JUHTIMISPROGRAMMIDE NÄITEID 83

5.1 Ventilatsiooni signalisatsioon 83

5.2 Tehase värava juhtimine 88

5.3 Hädastopp- ja käivituslüliti 92

5.4 Pidevjuhtimine 94

6 TÖÖ STEP 7 KESKKONNAS 97

6.1 Tarkvara LOGO!Soft Comfort 99

6.2 Tarkvara STEP7 MicroWIN 101

6.3 SIMATIC Manager (STEP7 for SIMATIC S7-300/400) 103
6.3.1 Projekti alustamine ja riistvara määratlemine 103
6.3.2 Sümbolite tabeli ja programmi koostamine 105
6.3.3 Programmi kontrollerisse laadimine ja testimine 108
6.3.4 Kontrolleri diagnostika 112

7 KASUTATUD KIRJANDUS 114

8 LISAD 115

8.1 Väikese jõudlusega kontrollerid 115

8.2 Keskmise jõudlusega kontrollerid 116

8.3 Suure jõudlusega kontrollerid 117

8.4 Tööstusarvuti 118

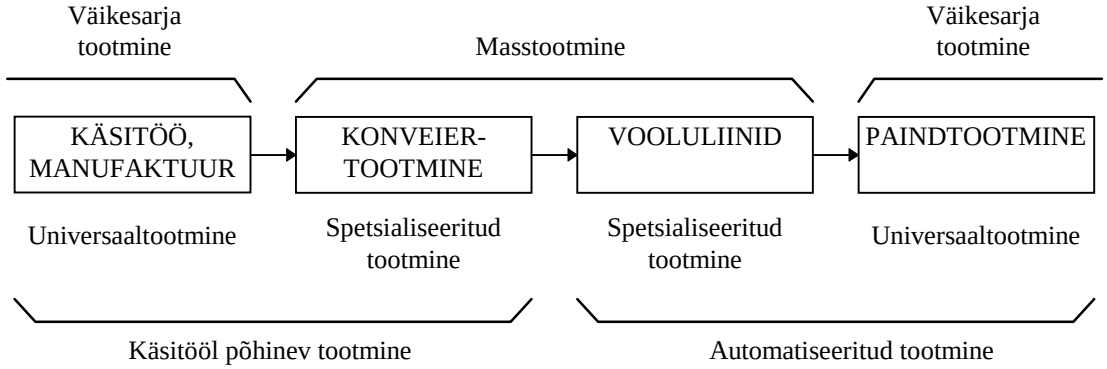
8.5 Operaatorpaneeliga kontrollerid 119

8.6 Tööjaamad 120

1 Tootmise areng ja automatiseerimine

1.1 Tootmise tehnoloogilise arengu põhietapid

Tootmine on sihipärane tegevus inimeste vajaduste rahuldamiseks, mille käigus inimesed loovad tootmisvahendite abil ainelist ehk materiaalset vara. Inimühiskonna ja tootmise tehnoloogilises arengus eristatakse pikemaid kindlate tunnustega etappe (joonis 1.1).



Joonis 1.1. Tootmise tehnoloogilise arengu etapid

Tootmise arengu alguseks võib sisuliselt nimetada aega, millal inimene leiutas esimese tööriista. Ühiskondliku arengu ja tootmise esimeseks ja pikimaks arenguetapiks loetakse agraarühiskonna ajajärku, mis algas u. 10000 aastat tagasi ja millal inimesed hakkasid tegelema paikse karjakasvatuse ja põllumajandusega. Agraarühiskonda iseloomustab põllumajanduslik tootmine ja käsitöö. Tootmine oli hajutatud ning hinnatud olid mitmekülgsete võimetega käsitöömeistrid, kes valmistasid unikaalseid esemeid. 18. sajandi keskpaigas muutus ühiskondade käitumine ja mõtlemislaad, mille tulemuseks oli tööstusrevolutsioon (aurumasina leiutamine 18. sajandi lõpus). Tekkis ***industriaalühiskond***, mille põhilisteks tunnusteks olid tootmise spetsialiseerumine, kontsentreerumine, maksimeerimine, tsentraliseerimine, standardiseerimine ja sünkroniseerimine. Eelmainitud tunnused eeldasid suurtööstuse väljaarendamist ja inimeste plahvatuslikku siirdumist suurematesse keskustesse (linnadesse). Töökohtade ja töövõtete spetsialiseerumise tulemusel kujunes välja konveiertootmine, kus iga inimene tegi töötamisel vaid mõnda üksikut lihtoperatsiooni. Töö hakkas toimuma sünkroonselt konveieri rütmiga. Järgmise sammuna toimus toodangu ja töövahendite standardiseerimine. Paralleelselt tööstusrevolutsiooniga toimunud energeetilisele revolutsioonile toimus masinate kiire areng. Masinate leiutamise ja ulatusliku mehhaniseerimise tulemusel hakkasid masinad inimest konveieri juurest välja vahetama. Masinate järjest laiema kasutuselevõtuga loodi eeldused automaatsete vooluliinide rakendamiseks. Kõik see soodustas tööstuse järjest suurenevat kontsentreerumist ja massilist tarbimist.

Tänapäeval kiire teadus- ja tehnikaarengu tulemusena on industriaalühiskond asendumas ***infoühiskonnaga***. Infoühiskonnas mängivad tähtsat rolli infoedastus ja -töötlus ning nende toimingute kiirus. Infoühiskond tähistab uut arenguetappi, mille tõi kaasa elektroonika ja arvutustehnika, sh. programmjuhtimise kasutuselevõtt. Selle tulemusel toimus tootmise iseloomu oluline muutumine. Arvutitega juhitavad robotid, arvjuhtimisega tööpingid on uue põlvkonna masinad. Tänu programmjuhtimisele on need masinad paindlikumad ja universaalsemad. Seega tõi programmjuhtimise kasutuselevõtt kaasa tähtsa muudatuse tootmises, s. o. jäiga automatiseeritud tootmise asendumise paindtootmisega. Tänu inimese loomulikule vajadusele mitmekesisuse ja vahelduse järele on paremini läbi löönud ettevõtted ja inimesed, kes on paindlikumad ja suudavad muutuvate oludega kiiremini kohaneda. Paindtootmise valdava kasutuselevõtuga toimub üleminek masstootmiselt sarja- ja väikesarjatootmisele, mis iseenesest, senikaua kui on vajadus masstootmise järele, ei tähenda automaatliinide kadumist.

Paindtootmist juhitakse hierarhiliste ja detsentraliseeritud juhtimissüsteemidega, mis koosnevad paljudest lokaalsetest alamsüsteemidest. Juhtseadmetena kasutatakse erineva võimsusega juhtraale (sh. tööstuskontrollerid), mis on varustatud vastava tarkvaraga ning võimaldavad inimese dialoogi automaatidega (arvutitega). Paindtootmissüsteemid on integreeritud ettevõtte automatiseeritud juhtimissüsteemi ning talitlevad koos projekteerimise ja tootmise tehnoloogilise ettevalmistuse automatiseertud süsteemidega.

Täielik paindtootmine, s. t. tootmise kõiki tasandeid hõlmav paindtootmine paneb sisuliselt aluse uuele ajajärgule tehnika arengus, milleks on tehisintellekti kasutamine. Täielik paindtootmine on tänapäeval juba reaalsus. Niisugusel tootmisel võib näiteks toote disainer või tehnoloog anda arvutile ette toote minimaalse arvu parameetreid või mustandjoonise jne., mille põhjal arvuti optimeerib ülesande, annab käskluse eri seadmetele vastavate programmide loomiseks. Tehisintellekt tähendab matemaatiliste, keemiliste, füüsikaliste jm. seoste põhjal uute seoste leidmise oskust, kus olulist rolli mängivad protsesside korduvuste põhjal järelduste tegemine ja juhuslikkuse integreerimine seadmesse või protsessijuhtimisse.

Järgnevalt selgitame mõningaid tootmise automatiseerimisega seotud mõisteid. ***Tootmise automatiseerimine*** on mehhanismide, seadmete, mehhaniseeritud protsesside jms. ühendamine terviklikuks juhtimissüsteemiks, milles energia, materjalide ja informatsiooni hankimine, töötlemine ja edastamine toimub inimese vahetu osavõtuta. Tootmise automatiseerimine vähendab tunduvalt inimese vaimsete ja kehaliste omaduste (kvalifikatsiooni, reageerimiskiiruse, meeleolu jm.) mõju tööle ning vabastab ta pidevast ühetoonilisest protsessi juhtimisest (mõõtmisest, tüürimisest, reguleerimisest, arvutamisest, operatsioonide ümberjaotamisest jne.).

Automatiseeritud tootmine on masintootmise kõrgeim vorm. Tehnika areng seisneb eraldi töötavate automaatseadmete asendamises automaattööpinkide ja -agregaatidega ning viimaste asendamises kompleksse automatiseerimisega, mida iseloomustavad elektroonika ja pneumaatika laialdane rakendamine, kõigi tootmisprotsessi põhi- ja abioperatsioonide mehhaniseeritus ja seostatus ning ühe või mitme juhtimisarvuti ja automatiseeritud juhtimissüsteemi olemasolu. ***Automatiseerimise tehniliste eeltingimuste*** hulka kuuluvad protsessi matemaatiline kirjeldatavus, töötlemis-, veo-, side- jm. vahendite mehhaniseeritus, tootmistehnoloogia automatiseerimisküpsus ja

töötajate kvalifikatsiooni vastavus automatiseeritud tootmise nõuetele. Tuleb märkida, et automatiseerimise otstarbekus oleneb kindlasti majanduslikust tasuvusest.

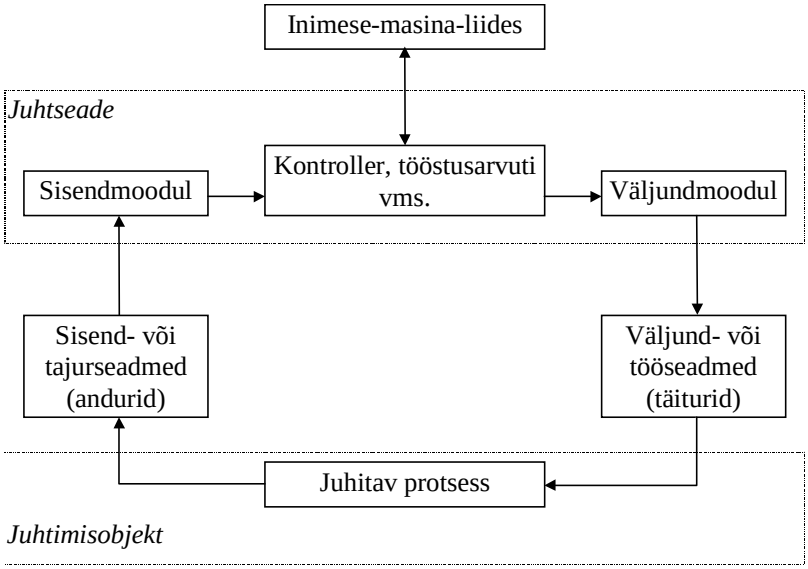
Tootmise automatiseerimises eristatakse automaatjuhtimist ja automatiseeritud juhtimist. ***Automaatjuhtimine*** on automaatika haru, mis käsitleb masinate, seadmete, tootmisprotsesside jm. inimese vahetu osavõtuta juhtimise meetodeid ja tehnilisi vahendeid. Juhitav objekt ja juhtimisseadmed moodustavad ***juhtimissüsteemi***. Juhtimissüsteemi tööeeskiri ehk juhtalgoritm määrab juhtimistoimingute järjekorra, laadi jne. Lihtsamate automaatjuhtimissüsteemide algoritm oleneb ainult süsteemi enda struktuurist ja ehitusest, keerulisemais süsteemides toimub juhtimine etteantava programmi või seaduspärasuse järgi. Juhtimissüsteem võib oma algoritmi muuta, kui objekti omadused või välised töötingimused (füüsikalised, mehaanilised jm. omadused) muutuvad. Töö iseloomu järgi eristatakse ***pideva*** ja ***katkelise*** (diskreetse) ***toimega automaatjuhtimist*** (joonis 1.2). Pideva toimega automaatjuhtimist kasutatakse näiteks keemiatööstuses ja katkelise toimega automaatjuhtimist autotööstuses. Juhtimissüsteemi ülesehituse järgi eristatakse *avatud* ja *suletud* kontuuriga automaatjuhtimist.

Avatud automaatjuhtimissüsteem ise oma toimimise tulemusi ei kontrolli, ***suletud*** ehk ***tagasisidestatud automaatjuhtimissüsteem*** kontrollib juhtimistulemusi järjepidevalt tagasiside abil. Suletud süsteemide põhiliigid on automaatreguleerimis-, järgiv- ja optimaaljuhtimissüsteemid. Viimaste algoritmid koostatakse nii, et tööseadmes kujuneks mingi(te) parameetri(te) poolest soodsaim olukord või tehnoloogiaprotsess.

Toime	Pideva toimega	Katkelise toimega
Mõõdetavad suurused	Rõhk, temperatuur, nivoo, vooluhulk, analüüs jne.	Kogus, hulk, aeg, positsioonid, kiirus jne.
Täiturid	Ventiilid, pumbad, kompressorid jne.	Mootorid, ajamid, klapid jne.
Protsesside liigid	Keemilised ja füüsikalised protsessid	Mehaanilised protsessid
Juhtimise viis	Pidevjuhtimine	Katkevjuhtimine (binaarsignaalid)
Juhtimissüsteem	Suletud kontuuriga	Avatud kontuuriga

Joonis 1.2. Pideva ja katkelise toimega automaatjuhtimine

Automatiseeritud juhtimissüsteem on tootmis-, majandus- vms. tegevuse juhtimise süsteem, mille puhul kasutatakse juhtseadmetena programmeeritavaid kontrollereid, tööstusarvuteid, personalarvuteid ja teisi andmetöötlusvahendeid (joonis 1.3). Tehnoloogilise protsessi automatiseeritud juhtimissüsteemis anduritelt saabuv informatsioon analüüsitakse juhtsüsteemis (arvutis, kontrolleris); seejärel sünteesitakse reguleerimiskorraldused automaattäiturile ja operaatorile, kes juhtimispuldil oleva kuvari või printeri kaudu vastavalt vajadusele sisestab või väljastab juhtimisinformatsiooni. Automatiseeritud juhtimissüsteemis sekkub inimene kas pidevalt või aegajalt tootmisprotsessi juhtimisse.



Joonis 1.3. Automatiseeritud juhtimissüsteemi struktuur

Kõik juhtimissüsteemid on jaotatud järgmisteks elementideks:

- juhitav protsess - tehnoloogiline tegevus nagu keevitamine, kuumutamine, saagimine jne.;
- sisendseadmed - lülitid, kontaktid, andurid, seadurid jne.;
- sisendmoodulid - kaitseahelad ja signaalimuundurid;
- kontrollerid või juhtarvutid - protsessor, mälu ja toiteplokk;
- juhtprogramm - protsessijuhtimise eeskiri;
- väljundmoodulid - kaitseahelad ja signaalimuundurid;
- inimese-masina-liides - juhtseadme ja protsessi jälgimiseks, juhtimiseks ja programmeerimiseks.

Juhtimissüsteemis tehnoloogiaprotsessist anduritega tuvastatav info muundatakse juhtseadmele vastuvõetavale kujule. Juhtseade töötleb seda infot mällu salvestatud programmi järgi ning väljastab juhtsignaalid täiturseadmetele, mis juhivad konkreetset protsessi. Juhtimissüsteemi juurde kuulub alati ***inimese-masina-liides*** ehk ***kasutajaliides*** (*Human-Machine-Interface, Man-Machine-Interface*), mis võimaldab inimesel osaleda juhtimiseks vajaliku (või täpsustava) informatsiooni sisestajana (programmeerijana) ning talitlusjärelvalve operaatorina või juhtimisprotsessi jälgijana.

1.2 Automaatliinid ja paindtootmissüsteemid

Automaatliinid on automaatsetest töömasinatest, teisaldusseadmetest ning juhtimis-, reguleerimis- ja kontrollseadmetest (-aparatuurist) koosnev automaatpingisüsteem toodete valmistamiseks inimese vahetu osavõtuta. Teenindava personali põhiülesanne on liini seadistamine ja häälestamine, töö jälgimine ja tekkivate seisakute korral normaalse tööseisundi taastamine ning töömaterjali etteandmine ja valmistoodete eemaldamine. Automaatliine kasutatakse väga mitmesugustes tööstusharudes. Tänapäeva automaatliine saab suhteliselt kergesti ümber kohandada ühetüübiliste toodete rühma piires ning nad õigustavad end sari- ja suursaritootmises. Väikesaritootmises kasutatakse paindlikke **a*utomaatpingisüsteeme*** millesse kuuluvad automaatvõllpingid, tooriku- ja detailivaheladustuse, detailikinnitusrakiste, detailiteisaldus- ja laadimisseadmete, lõikeriistamagasinide, kontrollseadmete ning automaatjuhtimisseadmete kogum teatavate kindlate detailide automaattöötlemiseks.

Automaatpingisüsteemid jagunevad

- *paindlikeks automaatpingisüsteemideks*, mis on ette nähtud väga mitmesuguste ühetüübiliste detailide väikesari- ja saritootmiseks. Nad koosnevad arvjuhtimispinkidest, vaheladudest, teisaldusseadmeist, tööstusroboteist ning tööriistade ja detailide magasinidest. Üldjuhtimist teostab keskarvuti, iga agregaati juhib kontroller ehk mikroarvuti.
- *ümberseadistatavateks automaatpingisüsteemideks*, mis on ette nähtud teatavate kindlate ühetüübiliste detailide sari- ja suursaritootmiseks. Nad koosnevad automaat- ja arvjuhtimispinkidest, detailimagasinidest, jäikadest või paindlikest detailiteisaldus- ja -laadimisseadmeist ning lõikeriistavahetusseadmeist.
- *jäikadeks automaatpingisüsteemideks*, mis on ette nähtud ühe detaili masstootmiseks. Nad koosnevad automaatpinkidest, detailimagasinidest, jäikadest detailiteisaldus- ja -laadimisseadmetest ning kontrollautomaatidest.

Automaatpingisüsteemidest, toorikute ja detailide, rakiste ja lõikeriistade keskladudest ning teisaldus- ja laadimisseadmeist koosnev kompleks, mille tööd organiseerivad ja juhivad keskarvutid, nimetatakse ***automaattootmiskompleksiks***.

Lähemalt tuleks vaadelda paindlikke automaatpingisüsteeme, mis on eelmainitud kolmest süsteemist oma ülesehituselt ja juhtimispõhimõttelt ilmselt kõige keerukam.

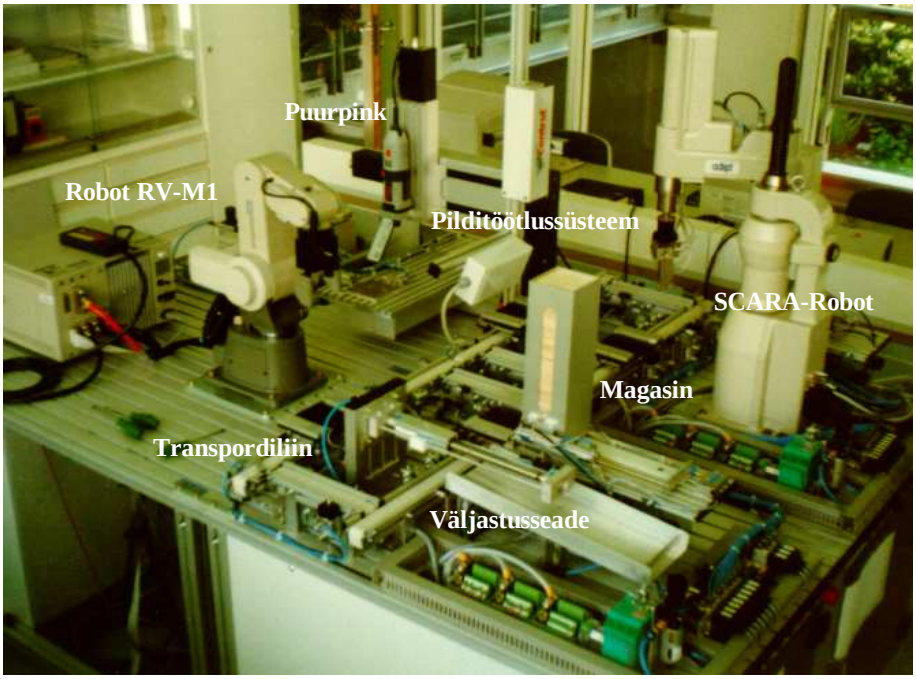
Paindtootmissüsteem on paindtootmise põhimõttel organiseeritud tootmisüksus ning sisaldab mitmeid erinevate ülesannetega juhtsüsteeme. Üks või enam paindlikku automaatpingisüsteemi moodustavad paindtootmissüsteemi.
Paindtootmine on automatiseeritud tööstuslik väikesaritootmine, mille puhul saab tööoperatsioone ja tehnoloogiat kergesti muuta. Tööoperatsioonide paindlikkust võimaldab universaalsete töökohtade (*paindtöötlemismoodulite*) ja seadmete (programmjuhtimisega tööpinkide ja tööstusrobotite) kasutamine. Tehnoloogilise paindlikkuse tagavad paindtöötlemismoodulite rööptöö, asendatavus ja reverseeritavus, ümberprogrammeeritavus, laonduse ja tööriistanduse automatiseerimine, transpordiseadmete paindlik töökorraldus (materjalide liikumisteede optimaalne valik ja operatiivne ümbersuunamine) ning programmjuhtimine. Paindtootmissüsteem võib kujutada endast paindtöötlemismoodulit, -jaoskonda, -tsehhi või ettevõtet. ***Paindtöötlemismoodul*** on

paindtootmissüsteemi väikseim iseseisev üksus, mis põhineb universaalseil programmjuhtimisega tehnoloogiaseadmeil ning on ette nähtud suure nomenklatuuriga toodete valmistamiseks. Paindtootmismoodul koosneb tööpingist, detailide ja tööriistade etteandeseadmeist (salvedest, kassettidest või konteinereist), transpordi- ja teisaldusseadmeist (konveiereist ehk transportliinidest, roboteist), kontroll- ja juhtseadmeist (kontrollereist, arvuteist ja juhtpultidest).

Paindtootmissüsteeme juhitakse *hierarhiliste* ja *hajutatud* (*detsentraliseeritud*) juhtimissüsteemidega, mis koosnevad paljudest kohalikest alamsüsteemidest. Juhtseadmeina kasutatakse mitmesuguse võimsusega juhtarvuteid ja kontrollereid, millede eritarkvara võimaldab inimese dialoogi automaatidega.

Ühel või mitmel robotil põhinevat paindtöötlemismoodulit nimetatakse ***robotsüsteemiks*** ehk ***robotmooduliks***. ***Robot*** on ümberprogrammeeritav isetoimiv masin, mida kasutatakse inimese liikumist, tajumist ja mõtlemist asendavais töödes (nt. esemete teisaldamisel, tööriista käsitsemisel ning keskkonna jälgimisel ja uurimisel). Eristatakse tööstus-, uurimis-, meditsiini-, põllumajandus- ja majapidamisroboteid. Roboti iseloomulikud tunnused on ühe- või mitmekäeline manipulaator ja programmjuhtimisseade.

Järgnevalt on näitena esitatud paindtootmissüsteem “FLAUSY” (joonis 1.4) [2], mida kasutatakse Kempteni Rakenduskõrgkoolis tudengite koolituseks − süsteem valmistab kolme eri toodet lauamänge: *Halma*, *Mühle* ja *Solitär*.



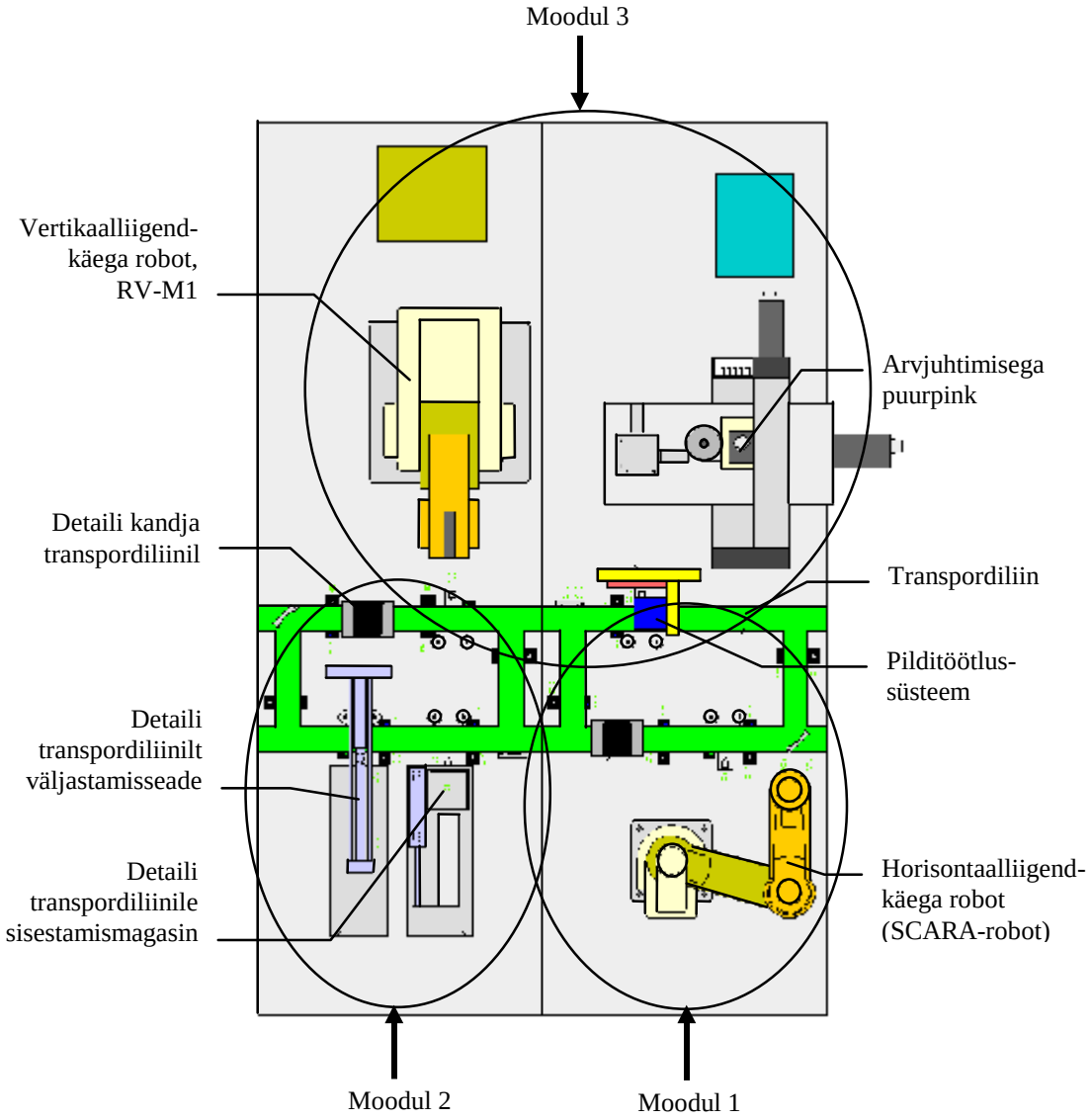
Joonis 1.4. Paindtootmissüsteemi “FLAUSY” üldvaade

Kõigepealt väljastatakse magasinist toode transpordiliinile; sealt edasi läheb edasi pilditöötlusseadmesse, kus määratakse kindlaks, mis tüüpi mustriga toode on. Kui toote muster on kindlaks määratud, tõstab robot *RV-M1* toote puurpingile ja puurpink

puurib tootele eelnevalt kindlaksmääratud mustriga augud. Kui muster on puuritud, tõstab robot *RV-M1* toote transpordiliinile, kus ta liigub *SCARA*-roboti juurde, mis asetab puuritud aukudesse tikud. Kui kõik tikud on aukudes, liigub toode mööda transpordiliini väljastamisseadmeni ja toode väljastatakse.

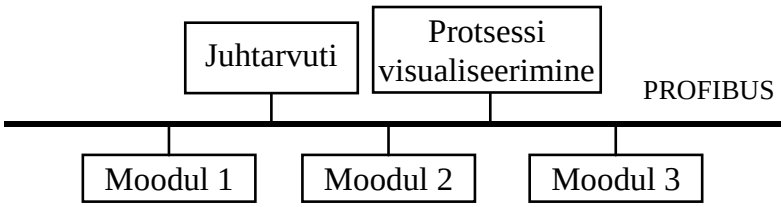
Kogu paindtootmissüsteem on jaotatud kolmeks paindtootmismooduliks (joonis 1.5), mida juhivad programmeeritavad kontrollerid. Igas moodulis on üks kontroller.

- Moodul 1 juhib tikumagasini, *SCARA*-robotit ning transpordiliini osa, mis jääb roboti ja pilditöötlusseadme töötsooni.
- Moodul 2 juhib detaili magasini, väljastamisseadet ja nende seadmete juurde kuuluvat transpordiliini osa.
- Moodul 3 juhib robotit *RV-M1*, puurpinki ja pilditöötlussüsteemi.



Joonis 1.5. Paindtootmissüsteemi “FLAUSY” moodulid

Kõikide moodulite, juhtarvuti ja protsessi visualiseerimisseadmete ühendamiseks kasutatakse andmesidevõrku *PROFIBUS*, mille kaudu toimub seadmete infovahetus (joonis 1.6). Juhtarvuti ülesandeks on koordineerida moodulite omavahelist tööd, protsessi visualiseerimise ülesandeks on esitada eemal paiknevale operaatorile reaalset protsessi arvuti ekraanil.

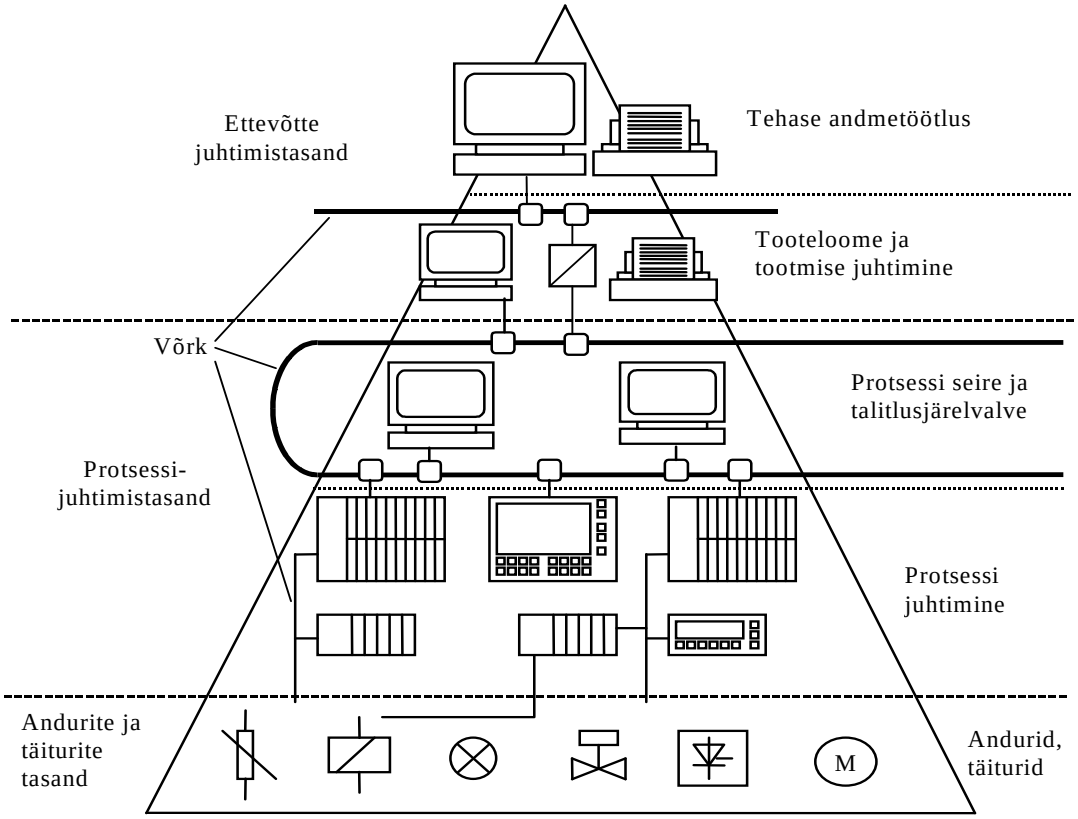


Joonis 1.6. Andmesidevõrk *PROFIBUS*

1.3 Tootmise hierarhiline juhtimine

Tootmise juhtimine ja automatiseerimine ei toimu ainult ühel kindlal tasandil, vaid hierarhiliselt mitmel erineval tasandil. Selline ***tootmise hierarhiline jaotamine ja juhtimine*** kujutab endast piltlikult *püramiidi*, mis sisaldab eri suunitluse ja ülesannetega *juhtimistasandeid* (ehk *kihte*). Tootmise hierarhiline juhtimine jaguneb kolmeks põhiliseks juhtimistasandiks (joonis 1.7) nagu:

- ettevõtte ehk tootmise juhtimistasand,
- protsessijuhtimistasand ja
- andurite ja täiturite tasand.



Joonis 1.7. Tootmise automatiseerimise püramiid

Ettevõtte juhtimistasandi peamised ülesanded on tootmis- ja töökorraldus, laomajandus, raamatupidamine ja rahandus ning side alumise tasandiga. Protsessijuhtimistasandi peamine ülesanne on tootmisprotsessi automaatne või automatiseeritud juhtimine, kontroll ja diagnostika ning side kõrgema ja madalama juhtimistasandiga. Andurite ja täiturite tasandi ülesanne on tootmisprotsessist tuleva info esitamine protsessijuhtimistasandile sobival kujul ning sealt tulevate käskude täitmine.

Eelmainitud kolm põhitasandit võib jaotada omakorda väiksemateks alamtasanditeks (joonise 1.7 parempoolne külg), millest kõige ülemine võiks olla tehase andmetöötlustasand, kuhu koguneb kogu tehasest tulev informatsioon ning mille

põhjal kontrollitakse ja analüüsitakse kogu tehase tööd majanduslikest, tehnilistest jm. aspektidest. Tooteloome ja tootmise juhtimistasandil toimub toodete väljatöötamine, arendamine, ümberkujundamine jne. ning nende tootmisse juurutamine ja tootmise juhtimine. Juhtseadmetena kasutatakse võimsaid juhtarvuteid ja servereid. Protsessiseire ja talitlusjärelvalve tasandil toimub tootmise osaprotsessi(de) talitluse jälgimine ja kontroll eri seadmetega, nagu tekstipaneelide, operatsioonipaneelide ja arvutitega. Protsessijuhtimistasandil saavad juhtseadmed (kontrollerid, tööstusarvutid) protsessianduritelt infot ning jagavad täiturseadmetele ja ajamitele vastavaid käsklusi. Sellel tasandil antakse ette protsessijuhtimiseks vajalikud parameetrid. Seega on tootmise hierarhia talitluseks vaja andmesidesüsteeme, milleks kasutatakse eri protokolliga andmesidevõrke.

Tootmise totaalne ehk ***täielik automatiseerimine*** kogu hierarhia ulatuses tähendab tootmises kõigi loetletud tasandite automatiseerimist ning sidumist omavahel ehk seadmete ühendamist ning ühildumist nii riist- kui tarkvaraliselt. Täielik automatiseerimine kaotab piirid

- arvutite ja programmeeritavate kontrollerite,
- protsessi talitlusjärelvalve ja juhtseadmete,
- tsentraalsete ja detsentraalsete seadmete ning
- katkeliste ja pidevate protsessi vahel.

Täielikku automatiseerimist või nimetada ka avatud automatiseerimiseks, kuna seda saab alati laiendada ning eri suundades arendada. Täieliku automatiseerimise eelisteks on

- koolituskulude vähenemine,
- riistvara hinna vähenemine,
- projekteerimise produktiivsuse tõus,
- madalad hoolduskulud,
- eri seadmete riskivaba ühendamine,
- paindlikkus jne.

Täieliku automatiseerimise aluseks ja eelduseks on informatsiooni liikumise võimalikkus tootmishierarhia nii vertikaal- kui ka horisontaalsihis. Informatsiooni liikumiseks piki tasandeid kasutatakse eri *andmesidevõrke*, risti tasandeid aga *võrgulüüse* ehk ühendusseadmeid eri andmesidevõrkude vahel.

Ka firma *Siemens* pakub oma seadmete SIMATIC baasil tootmise täielikku automatiseerimist, kuna viimased võimaldavad juhtida protsesse tootmise kõige alumisest kuni kõige ülemise hierarhia tasandini välja. Selline täielik automatiseerimise integratsioon tootmises on võimalik tänu veatule andmetöötlusele, seadmete konfigureerimisele ja programmeerimisele, seadmete ühildumisele ja eri andmesidevõrkude veatule ühendamisele kogu tootmise ulatuses.

Firma Siemens pakub tootmise täielikuks automatiseerimiseks järgmisi komponente:

- programmeeritavaid kontrollereid **SIMATIC S7/M7/C7**
- seadmeid hajusjuhtimiseks **SIMATIC DP**
- kontrollerite SIMATIC S7/M7/C7 programmeerimistarkvara **SIMATIC Industrial Software**
- programmaatoreid **SIMATIC PG** juhtseadmete programmeerimiseks
- tööstusarvuteid **SIMATIC PC** protsesside juhtimiseks ja jälgimiseks
- **SIMATIC PC-based Control** (***Win AC*****)** tarkvara tööstusarvutite *SIMATIC PC* programmeerimiseks
- operaatorpaneeli ja tekstidispleid protsessi talitlusjärelvalveks koos tarkvaraga *ProTool* või *WinCC* üldnimetusega **SIMATIC HMI**
- andmesidesüsteeme *Ethernet*, *Profibus* ja *AS-Interface* juhtseadmetevaheliseks suhtlemiseks kogu tootmise ulatuses, üldnimetusega **SIMATIC NET**
- tehase protsessi juhtimis- ja jälgimissüsteemi **SIMATIC PCS 7**, mis võimaldab ühendada kõik eelmainitud komponendid tootmishierarhia nii vertikaal- kui ka horisontaalsihis.

1.4 Andmesidevõrgud ja -liidesed

Andmesidevõrgud on hajusstruktuuriga andmesidesüsteemid, mille seadmed (kontrollerid, arvutid jne.) ehk võrgu *sõlmed* saavad vahetada informatsiooni omavahel. Võrgul on kindel struktuur ehk topoloogia, mille järgi eristatakse *ring-*, *täht-*, *lineaarset* ja *puu-* ehk *hargstruktuuri* ning nende kombinatsioone. Võrgud koosnevad signaalide ülekandmiseks mõeldud füüsikalisest keskkonnast ehk *riistvarast*, s.o. sideliinidest ja -seadmetest ning *tarkvarast* ehk programmidest. Eri tüüpi ja eri tootjate poolt valmistatud seadmeid saab ühendada võrku tingimusel, et neil on ühesugused füüsilised ja programmilised liidesed ning andmeportsioni edastusvormingud ehk *kaadrid* (*frames*) ja et andmevahetus toimub ühesuguse (standardiseeritud) protokolli järgi. Võrguprotokollide standardiseerimiseks on internatsionaalse standardiseerimisorganisatsioon välja töötanud 7-kihilise avatud võrgu mudeli [1]. Eri tootjate poolt loodud ja tootmise hierarhia tasandite võrkude võrdlemiseks kasutatakse näitajaid nagu võrguseadmete ehk -sõlmede maksimaalne arv, signaali maksimaalne ülekandekaugus, andmeedastuskiirus, kommunikatsioonimeetod jne. Võrku ühendatud seadmed jaotatakse *aktiivseteks* ehk *ülemseadmeteks* (*master devices*) ja *passiivseteks* ehk *alluvseadmeteks* (*slave devices*). Aktiivseadmed juhivad võrgus toimuvat andmeedastust. Nad võivad algatada andmevahetust välise nõudluseta, programmi järgi. Aktiivseadmeteks võivad olla personaalarvutid, programmeeritavad kontrollerid jne. Passiivseadmed aga ise andmevahetust ei algata, vaid võtavad vastu teateid ja vastavad aktiivseadmete infopäringutele. Passiivseadmeteks on tavaliselt muundurid, andurid ja täiturid. *Võrdõigusliku* kommunikatsiooni puhul omavad kõik seadmed õigust algatada infovahetust, kuid seda siin ei käsitleta.

Tootmise hierarhiline struktuur nõuab eri võrgusüsteemide kasutamist vastavatel tasanditel (vähemalt 3) ja nende omavahelist ühendamist (Joonis 1.7). Kõige ülemine tasand on ettevõtte juhtimistasand ja seda tüüpi võrku nimetatakse tööstuses *Ethernet-tööstusvõrguks*. Ethernet-tööstusvõrk on ettenähtud automaatikasüsteemide, personalarvutite ning tööjaamade omavaheliseks ühendamiseks ja andmesideks. Võrk on avatud ja seega ka laiendatav. Ettevõtte juhtimistasandil nõutakse enamasti suurt andmeedastuskiirust (10 Mbit/s või 100 Mbit/s) ja häirekindlust ning seda antud võrk ka omab. Protsessijuhtimistasandil paiknev võrk (Profibus-DP, Profibus-FMS, CAN, InterBus jne.) ühendab kõiki automaatikasüsteemides kasutatavaid seadmeid nagu programmeeritavaid kontrollereid, arvjuhtimisseadmeid, roboteid, tööpinke, paindtootmissüsteeme jne. Sageli nimetatakse protsessijuhtimistasandil paiknevat võrku ka *tööväljavõrguks*. Nõuded andmeedastuskiirusele sõltuvad selle võrgu puhul protsessijuhtimistasandil paiknevate seadmete hulgast, toimuvate protsesside kiirusest jne. Seetõttu on selle võrgu infoedastuskiirus valitav ja asub piirides 9,6…12000 kbit/s. Kõige madalamal andmeedastustasandil ehk andurite-täiturite tasandil kasutatakse sageli *lihtseadmevõrku* (Profibus-PA, AS-Interface, CAN SLIO jne.), et vältida suure hulga juhtmete vedamist andurite ja täiturite ühendamiseks juhtseadmega. Selle võrgu puhul ühendatakse teatud hulk vastavaid andureid ja täitureid ühele ja samale kahejuhtmelisele siinile, mille kaudu kas küsitakse nende olekut või väljastatakse neile mingi juhttoime. Lihtseadmevõrgul, näiteks AS-Interface puhul on 31 alamseadme suurim päringuaeg 5 ms.

Enamike eespool mainitud töövälja tasandi tööstusvõrkude aluseks on andmeside liidesed RS485, RS422, V.24 (TTY, Current Loop) ja RS232. Levinuim on andmesideliides RS485, mis on liidese RS422 täiustatud versioon; seda kasutatakse praegusel ajal standardliidesena eri seadmete vahel.

RS485 on välja töötatud andmesideks kuni 32 seadme vahel ja sobib master/slave tüüpi *multidrop* võrkudesse. Soovitatav vahemaa RS485 liidesel baseeruva andmesidevõrgu jaoks on kuni 1200 m. Liidese suureks eeliseks on see, et andmete ülekandmise suunda saab muuta. Sellist ülekandeviisi nimetatakse sageli *pool-dupleks* (*half-duplex*) ülekandeks ühes juhtmepaaris. RS485 on ülekandemeetod, mis on paljude tuntud andmesiinide (Profibus, Bitbus ja Interbus-S) aluseks.

V.11./RS422 on standardliides tööstuslikuks kasutamiseks. See on loodud multidrop andmesiini jaoks peaarvuti ja terminaalide vahel. Liides on tasakaalustatud, kasutab 4-soonelist juhet ja on suhteliselt immuunne häiretele. See liides muudab juhtmepaaris polaarsust sõltuvalt sellest, kas 1 või 0 on üle kantud. RS422 oli esialgu väljatöötatud 10 seadme jaoks, kuid tänaseks võib ühendada juba kuni 32 seadet. Maksimaalne soovitatud vahemaa on kuni 1200 m (100 kbit/s) või 50 m (10 Mbit/s).

Tavalise vaskjuhtme kasutamisel suurte kauguste vahel on ülekanne *usaldamatu*, sest häired on suured. Sageli vähendatakse info ülekandmiskiirust. Järeleproovitud ja testitud meetod, mida on kasutatud pikaajaliselt, on elektrivoolu ülekanne. Vooluring (*current loop*) on vanim teadaolev meetod. V24 impulsid määrab elektrivoolu olemasolu või puudumine juhtmepaaris (voolu olemasolul “1” ja voolu puudumisel “0”). Et mõlemat juhtmepaari vooluga varustada, on saatja ühendatud kui aktiivne ja vastuvõtja kui passiivne või vastupidi. Vooluringi abil saavutatakse usaldatavam ühendus, kuid see on suhteliselt tundlik häiretele, kui ta pole tasakaalustatud. Enamik TTY-liideseid kasutab tänapäeval *balansseeritud* vooluringi ehk voolu suuna muutust juhtmepaaris (±10 mA) sõltuvalt sellest, kas signaal on “1” või “0. See tähendab, et vool on liinis ka siis, kui infoülekannet ei toimu (olekus “0” on vool -10 mA). See meetod on väga usaldusväärne info ülekandmiseks kuni 18 km kaugusele.

1.5 Lokaalsed süsteemid ja nende elemendid

Tootmishierarhia iga tasand jaguneb üheks või mitmeks ***lokaalseks süsteemiks***. Lokaalse süsteemi moodustab üks või enam alamsüsteemi. Lokaalsed süsteemid koosnevad väga paljudest eri elementidest, mida üldiselt võib jaotada

- passiivseadmeteks ja
- aktiivseadmeteks.

Passiivseadmeteks nimetatakse seadmeid, mis on protsessijuhtimisega passiivselt seotud ja ise mingeid otsuseid vastu ei võta. Passiivseadmed on nt. andurid, täiturid, muundurid, juhtpuldid jne., mis andmevõrkudes töötavad alamseadmetena.

Andur on seadis, mis muundab mõõdetava füüsikalise suuruse mingiks teiseks nt. elektriliseks suuruseks, mida on parem mõõta, võimendada, muundada või kasutada juhtimiseks. Andureid liigitatakse mitmete tunnuste põhjal. Üheks võimaluseks on liigitada neid sisend- ja väljundsuuruste järgi. Tööstusautomaatikas rakendatakse peamiselt elektrilise väljundiga mehhaaniliste, termiliste, optiliste ja elektromagnetiliste suuruste andureid. Teiseks võimaluseks on andurite jaotamine nende otstarbe järgi vedelike kiirus- ja kuluanduriteks, asendi-, kiirus-, kiirendus-, voolu-, jõu-, momendi- ja rõhuanduriteks. Millist andurit tööstusautomaatikas kasutada, sõltub konkreetsest protsessist.

Täitur on distants-, automaatjuhtimis- ning reguleerimisseadmeis kasutatav elektriline, pneumaatiline, hüdrauliline, elektropneumaatiline või elektrohüdrauliline ajam, mis regulaatorist saadava signaali järgi muudab reguleeritava seadise asendit, olekut jne. Seega täidab täitur mingilt seadmelt või inimeselt saadud korraldusi.

Lokaalsetes süsteemides nimetatakse ***aktiiv***- ehk ***juhtseadmeteks*** seadmeid, mis aktiivselt juhivad ja kontrollivad protsessi. Aktiivseadmed saavad oma info kas passiivseadmetelt või teistelt aktiivseadmetelt ning juhivad kas passiivseadmeid või teisi aktiivseadmeid. Aktiivseadmeid võib jaotada

- infoesituse,
- signaalitöötluse,
- ehituse ning
- programmi realiseerimise ehk juhtimispõhimõtte järgi.

Märkimist väärib aktiivseadmete jaotamine programmi realiseerimise ja juhtimispõhimõtte järgi, kus eristatakse:

- riistvaralist ja
- tarkvaralist programmeerimist.

Riistvaralise programmeerimise eripäraks on see, et programmi esitamiseks ühendatakse juhtimiseks kasutatavad lülitid ja releed omavahel juhtmete abil mingit funktsiooni täitvaks seadmeks. Seega toimib süsteem riistvaratehnikal baseeruva programmjuhtimise puhul vastavalt süsteemi elementide elektrilistele ühendustele.

Riistvaralist programmi on algsest erineva funktsiooni täitmiseks raskem ja kulukam ümber programmeerida (ümber ehitada).

Tarkvaralisel programmeerimisel põhinevate aktiivseadmete puhul paikneb andureid ja täitureid siduv funktsioon ehk programm juhtseadme mälus ning seda on vajadusel lihtne programmaatori abil muuta ja parandada. Tarkvaralise programmeerimise puhul ei sõltu juhtseadme ehitus seega programmist. Tarkvaratehnikal põhinevate seadmete eelisteks võrreldes riistavaratehnikal põhinevatega on paindlikkus, töökindlus ja majanduslik tasuvus, kuna neid iseloomustab vabalt kombineeritav funktsionaalsus, integreeritud kommunikatsioon, intelligentsed moodulid protsess-orienteeritud juhtimiseks, sarnane ehk muutumatu konfiguratsioon, standardiseeritud programmeerimine ja kõikide funktsioonide programmeerimise võimalus.

Tarkvaralise programmeerimise majanduslik tasuvus hakkab toimima alates protsessi teatud suurusest ja keerukusest ning väljendub selles, et

- vähenevad elektrikilpide ja -kappide mõõtmed, mis vähendab hinda;
- väheneb juhtmete ja elektromehaaniliste ühenduste arv, mis suurendab töökindlust ja vähendab hinda;
- juhitava protsessi laiendamisel või muutmisel on lihtne muuta juhtprogrammi ning asendada või juurde lisada olemasolevatele riistvaraplokkidele uusi plokke, mis vähendab muudatustega seonduvaid kulutusi.

Seoses elektroonika ja mikroprotsessortehnika arenguga ning elektroonikaseadmete hinna pideva langusega hakkab tarkvaralise programmeerimise majanduslik tasuvus ilmnema järjest väiksemate ja lihtsamate protsesside juures. Nii näiteks on kõige väiksema SIEMENSi kompaktkontrolleri LOGO! hind juba praegu (1999. aastal) võrreldav kolme aegrelee hinnaga, kusjuures tal on juhtimiseks palju suuremad võimalused.

Tööstusautomaatika aktiivseadmeteks võivad olla näiteks

- loogikamoodulid – mikrokontrollerid, mis võimaldavad kuni mõnekümne anduri või täiturseadmega protsessi juhtida;
- programmeeritavad kontrollerid, mis jaotatakse oma jõudluse poolest kolme tasandisse; väiksema jõudlusega seadmed sarnanevad oma võimaluste poolest loogikamoodulitega, suurema jõudlusega seadmed aga väikese jõudlusega tööstusarvutitele;
- tööstusarvutid, mis sarnanevad oma töökiiruselt ja jõudluselt tavalisele personaalarvutile;
- personaalarvutid,
- tööjaamad või multiprotsessorsüsteem suurte andmehulkade töötlemiseks ja säilitamiseks.

Järgnevates peatükkides pannakse põhirõhk lokaalsetes süsteemides kasutatavateprogrammeeritavate kontrollerite iseärasustele, riist- ja tarkvarale ning nende kasutamisele tsükliliste protsesside juhtimiseks tööstuses.

2 Programmeeritavate kontrollerite riistvara

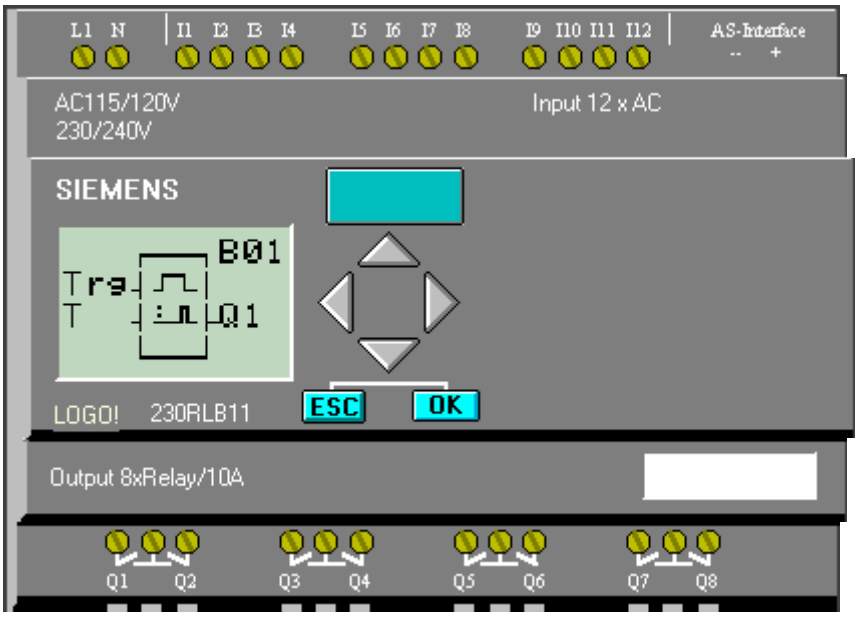
Programmeeritavad kontrollerid on programmjuhtimisseadmed, mis juhivad keerukat tehnoloogilist protsessi või seadet varem koostatud programmi (eeskirja ehk mällu salvestatud tarkvara) järgi. Programmeeritavate kontrollerite tootjad valmistavad neid eri suuruste ja tehniliste võimalustega. Millist seadet valida, sõltub sellest, kui palju mitmesuguseid andureid ja täitureid (passiivseadmeid) tuleb seadmega ühendada ning kui keerukas on mällu salvestatav programm. Seadmete valik sõltub juhitava protsessi keerukusastmest.

2.1 Ehitus

Programmeeritavad kontrollerid jagunevad oma ehituselt kaheks põhigrupiks:

- kompaktkontrollerid,
- moodulkontrollerid.

Kompaktkontrollerid on programmeeritavate kontrollerite peres kõige väiksemad. Nende eelisteks on väikesed mõõtmed ja nad sobivad ideaalselt protsesside juhtimiseks, milles vajatakse kuni 50 andurit või täiturit.



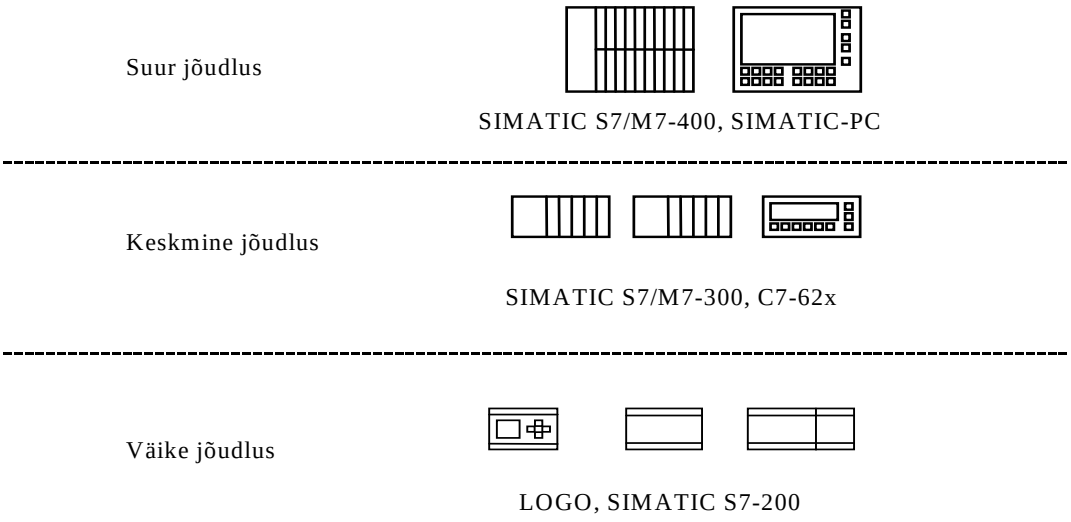
Joonis 2.1. Kompaktkontroller LOGO!

Kompaktkontrollerid erinevad teistest kontrolleripere liikmetest sellega, et nii toiteplokk kui ka protsessor ja sisend-/väljundplokk paiknevad kõik ühes ümbrises. Enamik kaasaegseid kompaktkontrollereid omavad kasutajaliidest seadme esipaneelil,

mis võimaldab programmaatori või vastava tarkvara puudumisel teda programmeerida. Üheks tuntumaks kompaktkontrolleriks võib lugeda firma Siemensi poolt toodetavat LOGO!-sarja (joonis 2.1). See sari sisaldab mitmesuguste tehniliste parameetritega kontrollereid, mida saab kasutada valgustuse, elektrikardinate, väravate, uste, ventilatsiooni, soojenduse, väikeste mootorite, presside, kompressorite, pumpade jpm. seadmete juhtimiseks.

Moodulkontrollerid jagunevad oma jõudluse ja üleehituse järgi järgmiselt (joonis 2.2):

- mikrokontrollerid (S7-200), mis võimaldavad lahendada juhtimisülesandeid, milles ei kasutata suurt kiirust ja mille sisendite/väljundite maht ei ole tavaliselt üle 500 punkti (anduri/täituri). S7-200 seeria kontrollereid saab kasutada nii kompakt- kui ka moodulkontrolleritena, lisades protsessorploki külge (mis sisaldab ka toiteplokki) digitaal/analoog- sisend- või väljundplokke;
- keskastme kontrollerid (S7-300, M7-300 ja C7-6xx), mis võimaldavad juhitavas protsessis kasutada kuni mõni tuhat andurit ja täiturit ning seega ka suuremat töökiirust; siia kuuluvad ka väikese jõudlusega tööstusarvutid;
- kõrgastme kontrollerid (S7-400, M7-400), mis võimaldavad juhitavas protsessis kasutada kuni mõnikümmend tuhat andurit ja täiturit ning väga suurt töökiirust; need kontrollerid sarnanevad oma ehituselt ja jõudluselt personaalarvutitega.



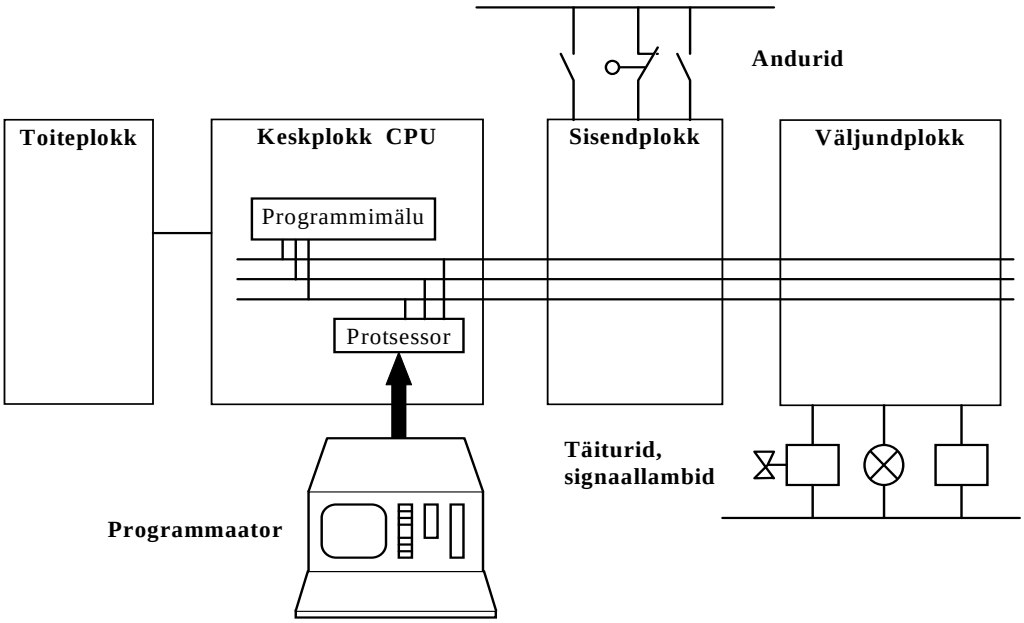
Joonis 2.2. SIMATIC-sarja kontrollerite jaotus jõudluse järgi

SIMATIC-M7 sarja kontrollereid nimetatakse tööstusarvuteiks kuna neil on personalarvuti funktsionaalsus nii protsessori kui ka töökiiruse, mälumahu jm. parameetrite poolest.

Sõltumata seadmete suurusest ja jõudlusest koosnevad kõik programmeeritavad kontrollerid järgmistest eri otstarbega sõlmedest (joonis 2.3) [9]:

- keskplokk koos protsessori ja programmimäluga,
- toiteplokk,
- sisend/väljundplokid ja
- siinisüsteem.

Need sõlmed võivad olla realiseeritud kontrolleris kas eraldi moodulitena või ühtse tervikuna. Seega koosneb juhtsüsteem alati anduritest, täituritest ja juhtseadm(et)est ehk passiiv- ja aktiivseadmetest.



Joonis 2.3. Programmeeritav kontroller

2.2 Tööpõhimõte

Programmeeritava kontrolleri tööpõhimõte on järgmine. Teatud ajahetkel saabub anduritelt signaal kontrolleri sisendplokki. Keskplokis töötav protsessor kontrollib iga teatud ajavahemiku järel mällu salvestatud programmi järgi sisendite olekut sisendplokis. Sõltuvalt sisendplokist saadud informatsioonile saab protsessor programmi põhjal otsustada, millise väljundi olekut väljundplokis tuleb muuta. Programmimälus on iga käsu jaoks oma koht ehk *pesa*. Mälupesad nummerdatakse järjekorras ning neid numbreid kasutatakse mälupesa aadressina. Protsessor pöördub aadressiloenduri abil üksikute mälupesade poole ning saadab info käsuregistrisse. Seega hakkab protsessor töötlema igalt uuelt aadressilt leitud mälupesa sisu, mis on viidud käsuregistrisse. Iga käsu järel suurendatakse aadressiloenduri sisu ühe võrra (joonis 2.4) [9].

Protsessor käib korduvalt ehk tsükliliselt läbi kõik programmimällu kirjutatud käsud, s. t. pöördub tsükliliselt vastava aadressiga mälupesade poole. Seda tüüpi juhtimist nimetatakse ***tsükliliseks programmjuhtimiseks***. Programmi täitmist esimesest käsust kuni viimase käsuni nimetatakse programmi *tsükliks* ja aega, mis selleks kulub nimetatakse *tsükliajaks*.

Programmjuhtimise tsükkel koosneb järgmistest etappidest:

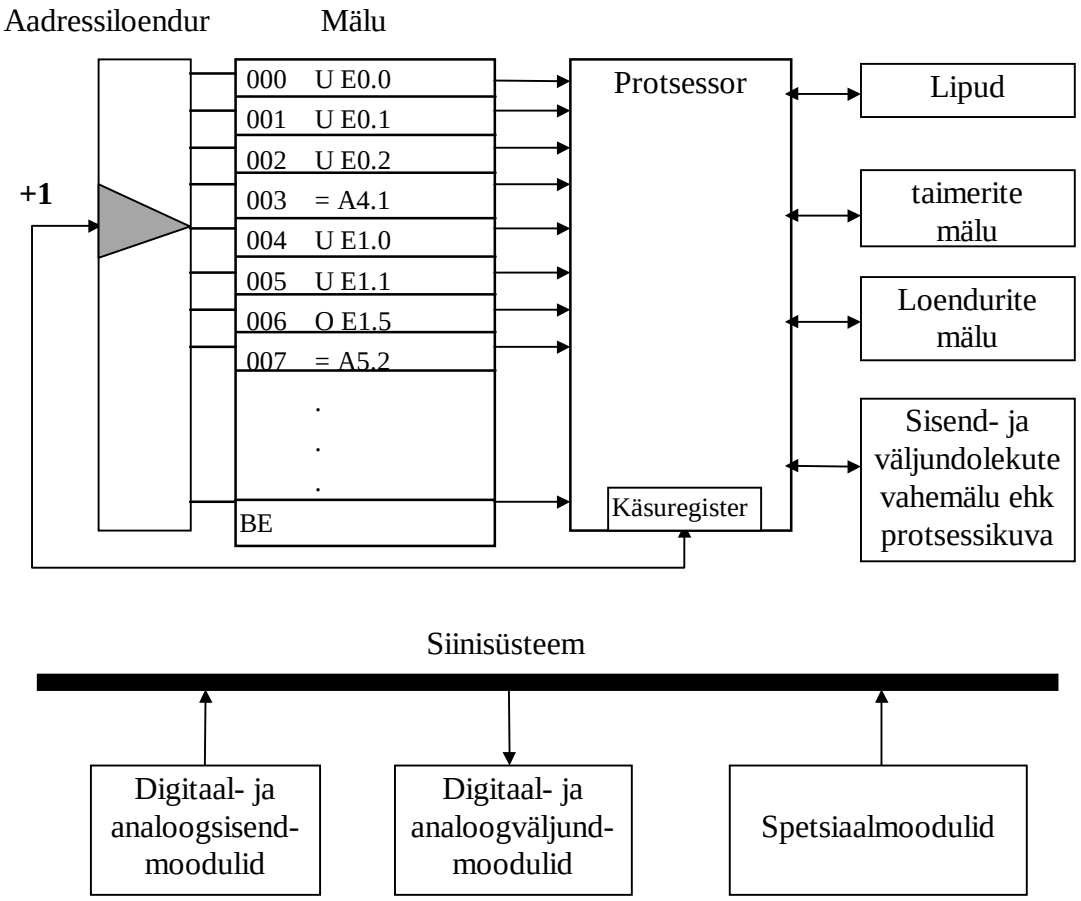
1. Protsessor pärib kõigi sisendite olekuid ning salvestab need vahemällu. Vastav päringuprogramm on salvestatud kontrolleri juhtimissüsteemi.
2. Protsessor töötleb kasutajaprogrammi. Loeb käske, töötleb ja täidab neid.
3. Protsessor saadab väljundite uued olekud väljundkaardile.

Programmimäluna kasutatakse pooljuht- või ketasmälu. Mälud koosnevad *mälupesadest*. Mälupesade hulk määrab mälumahu, mis on $N \cdot M$ bitti, kus N tähistab mälupesa suurust bittides ja M mälupesade arvu.

Juhtseadmetes kasutatavaid pooljuhtprogammimälusid jaotatakse

- vabalt programmeeritavateks ja
- jäigalt programmeeritavateks.

Vabalt programmeeritavad mälud on *muutmälud*. Muutmälu (*RAM, random access memory*) on lugemis-kirjutamismälu, kuhu saab infot salvestada ja lugeda vastavalt kasutaja soovile. Muutmällu salvestatud info säilib ainult senikaua kui mälul on toitepinge. Info säilitamiseks seadme väljalülitamisel kasutatakse juhtseadmetes primaarelemente või akut. Vabalt programmeeritavate mälude hulka loetakse ka poolpüsimälud EEPROM-EAPROM, kuhu saab vastava programmaatori olemasolul infot elektriliselt salvestada ja sealt kustutada; need on *ümberprogrammeeritavad* mälud.



Joonis 2.4. Programmimälu ja protsessori tööpõhimõte

Jäigalt programmeeritavateks nimetatakse püsimälusid, kuna sinna salvestatakse valmistaja või kasutaja poolt programm vaid üks kord. Neist saab infot lugeda, kuid seda muuta pole võimalik (*ROM, read only memory*). Jäigalt programmeeritavateks nimetatakse ka poolpüsimälusid EPROM ja REPROM, kuna neis info kustutamine toimub ultraviolettkiirguse abil. Poolpüsi- ja püsimäludes säilib informatsioon ka toitepinge puudumisel.

Juhtseadmete programmaatorites kasutatakse ümbrikketasmälu (*floppy disk*) või kõvaketasmälu (*hard disk*).

Juhtseadme protsessori ja sisend/väljundliideste (plokkide) vahel kasutatakse info vahetuseks *siinisüsteemi*, mille abil saab omavahel ühendada palju eri komponente. Siinisüsteem koosneb kolmest siinist: aadressi-, andme- ja juhtsiinist. Aadressi- ja andmesiinid on kaheksasoonelised ja infot edastatakse neis üks bait korraga. Aadressisiini mööda saab edastada aadresse 0…255 (nt. sisend- ja väljundplokkide aadresse). Sisend/väljundplokid reageerivad protsessori pöördumistele vaid sel juhul kui nad on ühendatud siiniga. Kui mingi sisendplokk tuvastab siinil oma aadressi, väljastab ta andmesiinile ühe baidi kaheksa biti olekud. Kui andmesiinile saabub väljastamiseks kaheksa bitti, antakse need bitid väljundploki kaudu edasi väljundseadmetele. Juhtsiini kaudu edastatakse signaale, mida kasutatakse juhtseadme töö juhtimiseks ja kontrolliks.

Juhtprogrammi koostab kontrolleri kasutaja *programmaatori* abil. Viimane ühendatakse juhtplokiga siis, kui tuleb salvestada uus programm või muuta varem salvestatud programmi. Programmaatoriga saab jälgida ka programmi tööd ning otsida ja parandada selle vigu.

Toiteplokk on seade protsessori, sisend/väljundmoodulite ja spetsiaalmoodulite toiteks alalispingega 5 V. Andurite ja täiturite toiteks kõrgema pingega (nt. 24 või 220 V) kasutatakse lisatoiteallikaid või -plokke.

2.3 Arengutendentsid

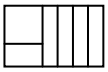
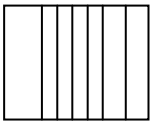
Maailmas on palju tööstuskontrollereid tootvaid firmasid, kuid toodetavate kontrollerite põhiomadused langevad suures osas kokku, s. t. nad on ehituselt ja tööpõhimõttelt sarnased. Maailmaturul omab liidripositsiooni turuosaga 25 % SIEMENS, kelle kontrolleriperekonnad SIMATIC S5 ja S7 on laialt tuntud. Selle tõttu käsitletakse käesolevas raamatus peamiselt SIMATIC-kontrollerite tarkvara ja programmeerimist, mis aga ei erine kuigi oluliselt teiste tuntud kontrollerite programmeerimisest, kuna viimane vastab standardile **IEC 61131**.

Programmeeritavad kontrollerid on jõudnud kõikidesse tööstusvaldkondadesse, büroodesse, elumajadesse jne. Kontrollerite eelisteks võib lugeda nende paigaldamise, seadistamise ja programmeerimise lihtsust ning majanduslikku tasuvust.

Arvutustehnika edasine areng lubab eeldada, et tööstuskontrollerid lähenevad üha enam personaalarvutitele. Tööstusarvutitest nagu SIMATIC PC rääkimata, mis ei jää praegu oma võimaluste ja tehniliste näitajate poolest millegagi alla kaasaegsetele personaalarvutitele. Juba praegu on mõnede firmade tööstuskontrollerid (SIMATIC C7) realiseeritud ühtse tervikuna operaatorpaneeliga. Eelmainitud lahendus laiendab tunduvalt kontrolleri kasutusvõimalusi ja vähendab talitluskulusid, kuna sellise seadme kogumaksumus on odavam kui eraldi seadmete puhul. Samuti väheneb siiniga ühendatud seadmete arv ja seetõttu ka elektriliste, elektromagnetiliste ja mehaaniliste häirete ning vigade tõenäosus. Samuti vähenevad kulutused tark- ja riistvarale. Tööstuskontrollerite tähtsaim arenguperspektiiv on vaieldamatult areng andmesidevõrkude ja -siinide ning infovahetuse kaudu (täielik automatiseeritus). See areng võib edaspidi kaasa tuua muutusi kontrolleri ehituses. Seega omavad tööstuses juba praegu või hakkavad peagi omama üha suuremat tähtsust universaalseadmed, mida saab rakendada eri ülesannete lahendamiseks ja protsesside juhtimiseks, sest nad võimaldavad paindlikku tootmist.

3 Programmeeritavate kontrollerite tarkvara ja programmeerimine

Sõltuvalt automaatikasüsteemi vajadustest võib SIMATIC kontrollerite programmeerimiseks kasutada eri tarkvara pakette. Enamasti kasutatakse eri jõudlusega kontrollerite programmeerimiseks eri tarkvarapakette, kuna kontrollerite kasutamise valdkonnad, vajadused ja võimalused on erisugused. Lisaks põhitarkvarale saab kontrolleri programmeerimiseks kasutada lisatarkvarapakette (joonis 3.1), mis on mõeldud spetsiaalseks otstarbeks nagu reguleerimiseks, diagnostikaks, simuleerimiseks, programmeerimiseks, häälestamiseks, optimeerimiseks jne.

Kontrolleri tüüp	Vajalik tarkvarapakett	Lisatav tarkvarapakett
SIMATIC S7-200 	STEP7-Micro/DOS STEP7-Micro/WIN	Puudub
SIMATIC S7-300 	STEP7-Mini	Puudub
SIMATIC S7-300  SIMATIC S7-400 	STEP7	S7-SCL CFC S7-GRAPH S7-HiGraph FuzzyControl Standard PID Control Modular PID Control DOCPRO TeleService S7-PDIAG S7-PLCSIM PRODAVE MPI PC-DDE-Server

Joonis 3.1. Vajalikud ja lisatavad tarkvarapaketid

Lisatavate tarkvarapakettide kohta saab täpsustavat informatsiooni seadmete kataloogidest. Mõningaid lisatavaid tarkvarapakette käsitletakse ka peatükis 3.3.

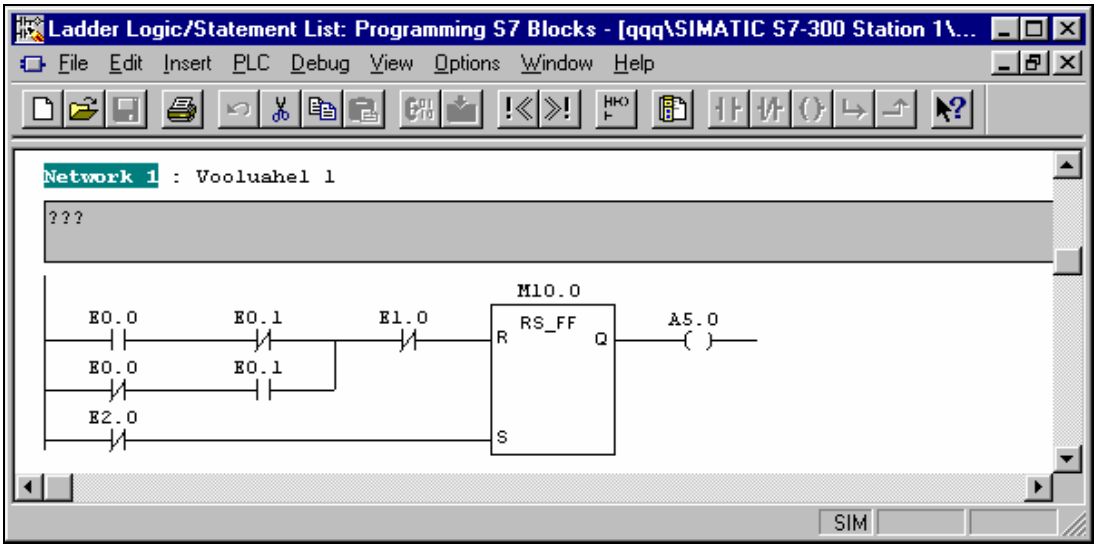
3.1 Programmeerimiskeeled ja -viisid

Maailmaturul tuntud firmade kontrollerite tarkvara struktuuri ja programmeerimise põhimõtted on määratud rahvusvahelise standardiga IEC 61131-3. Standardi järgi on kontrolleri programmeerimisel lubatud kasutada nt. järgmisi programmeerimiskeeli:

- kontaktaseskeemi *Ladder diagram (LD)* või *Ladder Logic (LAD)*
- algoritmi plokkskeemi ehk sammprogrammi *Sequential Function Charts (SFC*)
- loogikaskeemi *Function Block Diagram (FBD)*
- kõrgkeelt *Structured Text (ST)* (**C++** või **Pascal**'iga sarnane)
- käsulisti *Instruction List (IL)* või *Statement List (STL)*

Kontaktaseskeemina programeerimine kujutab endast programmi graafilist esitust ja on mõeldud relee-kontaktor-juhtimislülituste hõlpsaks teisendamiseks kontrolleri programmiks. Kontaktaseskeem sarnaneb tavalise elektriskeemiga. Kui tavalises elektriskeemis kontakte, lüliteid ja täitureid ühendavad juhtmed paiknevad vertikaalsihis, siis kontaktaseskeemi puhul paiknevad nad horisontaalsihis. Kontaktaseskeemina programmeerimise eelisteks on *kasutajasõbralikkus* ja lihtsus. Tavaliselt saab kontaktaseskeemina programmeerimisel kasutada kõiki juhtimisoperatsioone ja käske nagu teisteski esitusviisides.

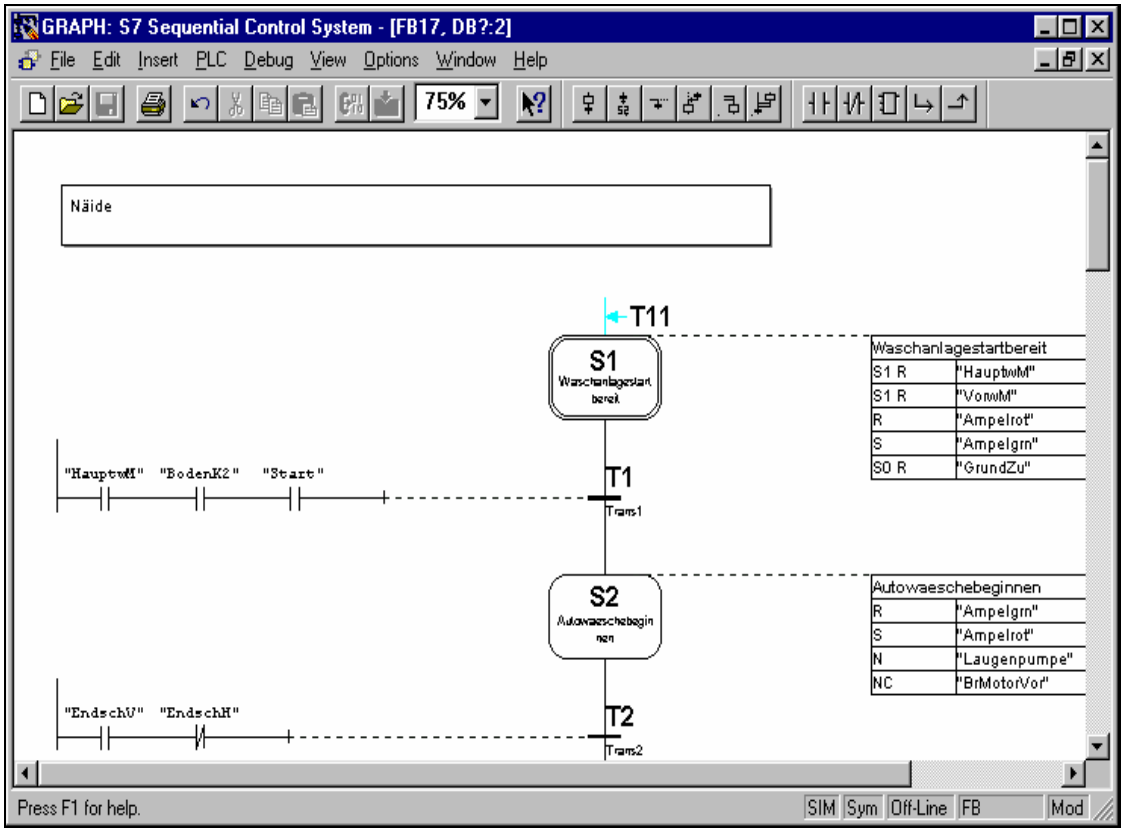
Joonisel 3.2 on näha kontaktaseskeemina esitatud juhtimisprogramm ja kontrolleri sisend- ja väljundoperandide tähised.



Joonis 3.2. Kontaktaseskeemina programmeerimine

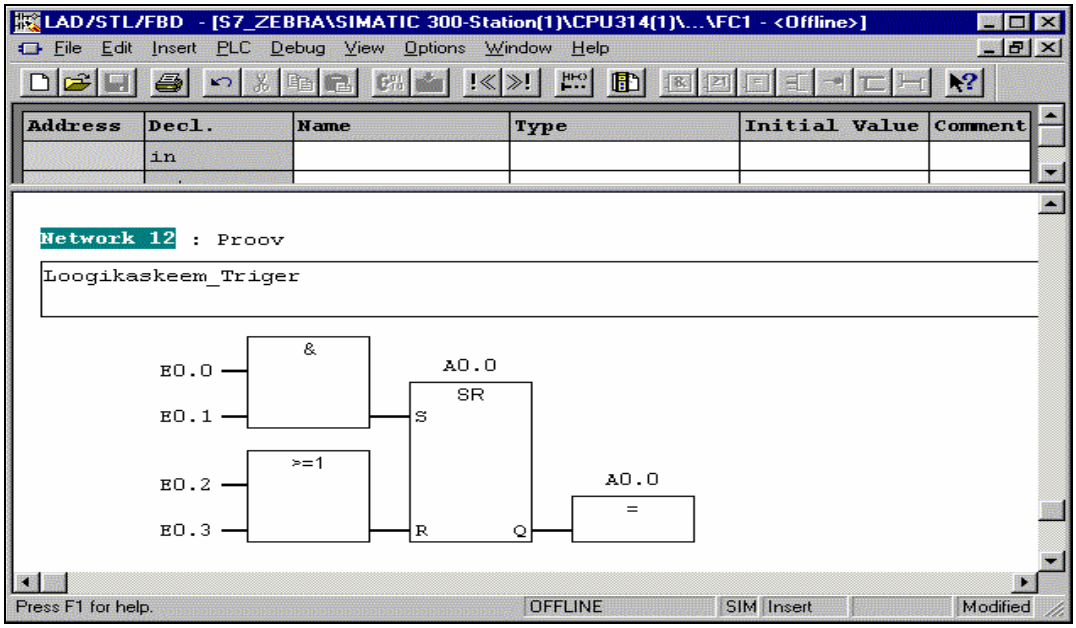
Programmeerimine algoritmi plokkskeemina kujutab endast graafilist programmeerimisviisi. Kaks põhikomponenti on seejuures operatsiooni- ja tingimusplokid. Tingimusplokid määravad ära järgneva operatsiooniploki. Iga operatsiooniplokk sisaldab üht või mitut jadamisi täidetavat operatsiooni. Tingimused

määravad algoritmi kulgemise ja seepärast võib tingimusplokke nimetada ka ülekande- ehk siirdeplokiks. Kui üldine juhtalgoritm on operatsiooni- ja tingimusplokkidena määratud, tuleb neid plokke programmeerida kas kontaktaseskeemina, käsulistina või kõrgkeeles. Joonisel 3.3 on esitatud operatsioonid tähega S ja tingimused tähega T.



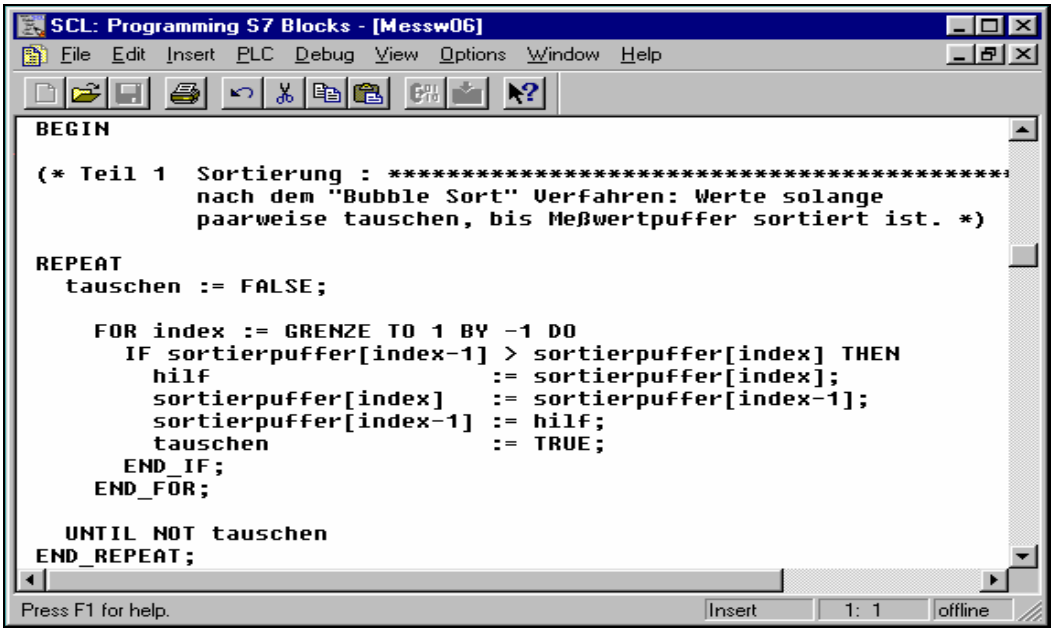
Joonis 3.3. Algoritmigraafskeemina programmeerimine

Loogikaskeemina programmeerimine on samuti graafiline programmeerimine (joonis 3.4). Loogikaskeemina programmeerimisel on põhielemendiks *loogikaelement*. Sisendid paiknevad elementide vasakul, väljundid paremal küljel. Loogikaskeemide puhul eristatakse põhiloogikafunktsioone ja kasutaja loogikafunktsioone. Põhifunktsioonid on tarkvara tootja poolt evitatud enamlevinud loogikafunktsioonid. Kasutaja loogikafunktsioonid on programmeerija poolt mingi konkreetse ülesande täitmiseks defineeritud ja programmeeritud plokid. Sellist programmeerimisviisi kasutatakse enamasti keerukate süsteemide ja juhtsüsteemis sageli esinevate juhtahelate programmeerimiseks.



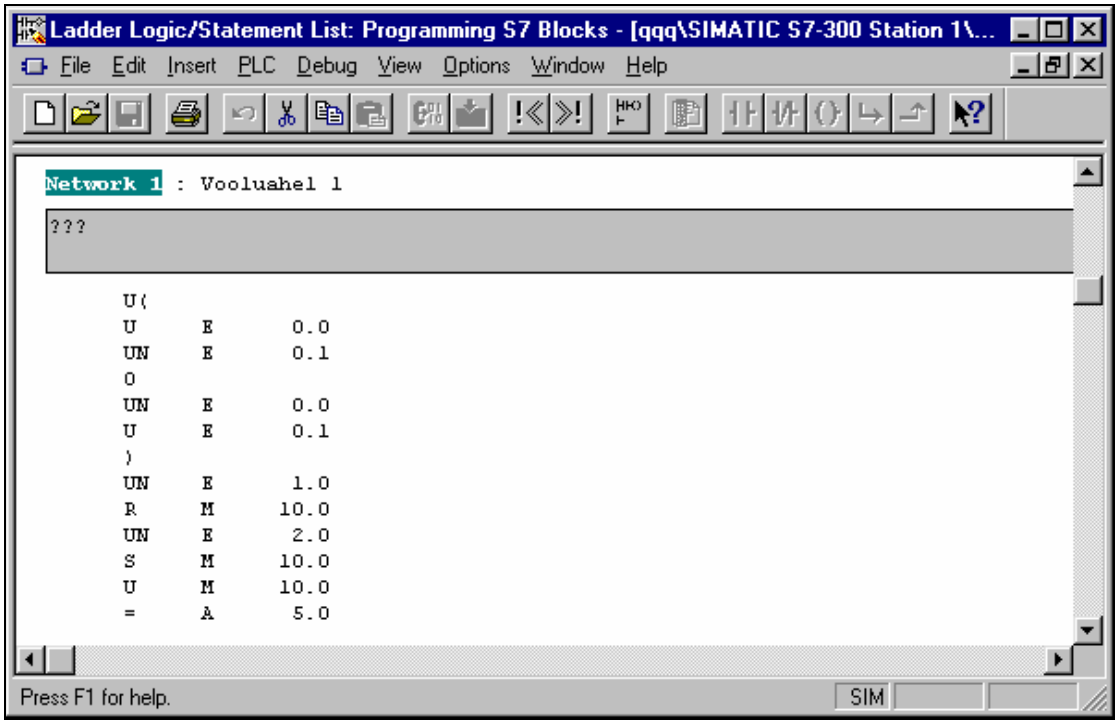
Joonis 3.4. Loogikaskeemina programmeerimine

Üldlevinud ***kõrgkeeles programmeerimine*** (joonis 3.5) sisaldab kõiki tänapäeva kõrgkeeltes esinevaid põhikäske nagu valikukäske (IF-THEN-ELSE and CASE OF) ja tsüklite käske (FOR, WHILE and REPEAT) jne. Tänu selle võimaluse lülitamisele tööstuskontrolleritesse on inimesel, kes kasutab teiste automaatika süsteemide programmeerimisel kõrgkeeli C++, Pascal või Basic, suhteliselt lihtne programmeerida ilma ümberõppimiseta ka tööstuskontrollereid. Tänu kasutatavale kõrgkeelele muutuvad tööstuskontrollerid teiste automaatjuhtimisseadmetega ühtseks tervikuks.



Joonis 3.5. Kõrgkeeles programmeerimine

Käsulistina programmeerimine (joonis 3.6) kujutab madala nivoo keeles programmeerimist. Käsulistis esitatakse programm, vastupidi kontaktaseskeemile ja loogikaskeemile, tekstina. Käsulistina programmeerides lahendatakse kogu juhtimisülesanne üksikute käskude jadana. See võimaldab teha lühemaid ja kiiremaid programme, kuna saab paremini kasutada mälu mahtu. Võrreldes teiste keeltega on ta raskemini õpitav, kuna teistes keeltes (nagu kontaktaseskeem või loogikaskeem) hõlmab üks plokk sageli mitut konkreetseks operatsiooniks vajalikku käsulisti käsku.



Joonis 3.6. Käsulistina programmeerimine

Rahvusvahelise standardi IEC 61131 eelisteks on see, et ta lubab ühe ja sama kontrolleri puhul kasutada mitmeid eri programmeerimisviise. See tähendab, et igaüks leiab enda võimetele vastava programmeerimiskeele.

SIEMENSi programmeeritavate kontrollerite SIMATIC S7/M7/C7 puhul saab kõiki eelmainitud esitusviise kasutada vastava tarkvarapaketi omamise korral. Ka teiste firmade kontrollerite tarkvara võimaldab kasutada vähemalt kolme erinevat programmeerimisviisi (kontaktaseskeemi, loogikaskeemi ja käsulisti).

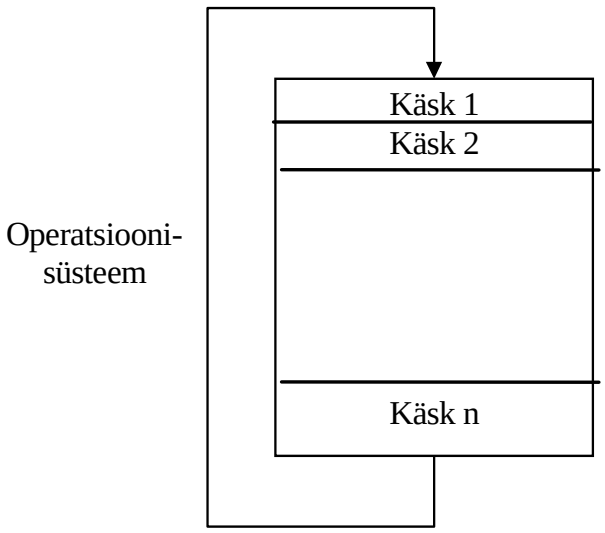
3.2 Programmeerimiskeel STEP 7

3.2.1 Programmeerimise alused

Kõikide juhtseadmete protsessorites, sh. kontrollerites töötavad kahte tüüpi programmid: *operatsioonisüsteem* ja *kasutajaprogrammid*. Igal protsessoril on operatsioonisüsteem, mis organiseerib protsessori tööd ning pole otseselt seotud juhitava protsessiga ehk juhtimisülesandega, mis hõlmab vastavate sisendite olekutele põhinevaid väljundite reaktsioone. Operatsioonisüsteem organiseerib näiteks protsessori käivitust, protsessikuvade sisend- ja väljundtabelite moodustamist, kasutajaprogrammi väljakutsumist, määrab katkestused, töötleb vigu, korraldab tööd mälupiirkondadega ning kommunikeerub programmaatorite ja teiste seadmetega. Kasutajaprogramm tuleb kasutajal endal programmeerida ja protsessorisse laadida. Kasutajaprogramm on kindla protsessi juhtimiseks, mis tähendab seda, et selles programmis saab määrata reaktsioonid vastavatele katkestustele, töödelda protsessist tulevaid andmeid, määrata tingimused protsessori lülitamiseks talitlusse ning programmi talitluskõrvalekallete töötlemiseks.

Kasutajaprogrammi võib omakorda jaotada *lineaarseks* ja *jaotatud* programmiks. Siit tulenevalt eristatakse ka lineaarset ja jaotatud programmeerimist.

Lineaarseks programmeerimiseks nimetatakse programmi koostamist, milles kõik programmikäsud on üksteise järel reas ning nende täitmine toimub samas järjekorras (joonis 3.7). Lineaarset programmeerimist kasutatakse lihtsate automaatikaseadmete, nagu loogikamoodulite (nt. LOGO!) puhul, millega juhitakse ka suhteliselt lihtsaid protsesse.



Joonis 3.7. Lineaarne programmeerimine

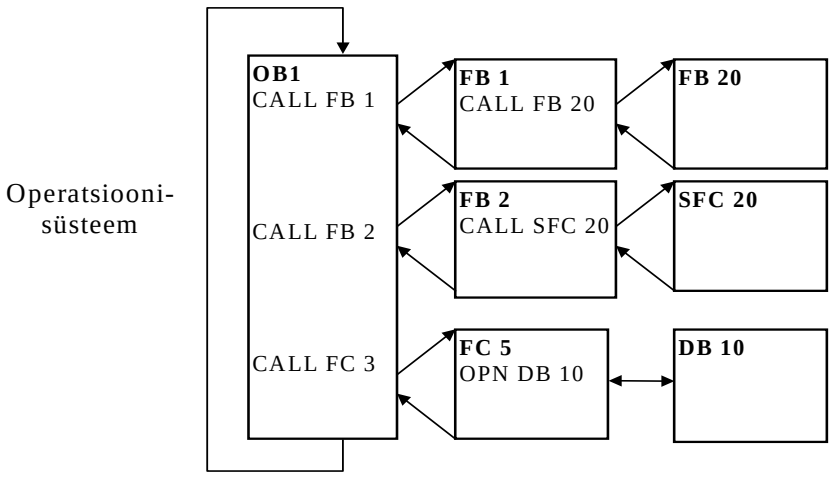
Jaotatud programmeerimiseks nimetatakse sellist programmi koostamist, milles koostaja jaotab juhtimisprotsessi operatsioonideks ja osaülesanneteks, mis on omavahel seotud kindlate tingimustega. Jaotatud programmeerimise eelisteks on

- programmeerimise lihtsus,
- alamprogrammide korduva kasutamise võimalus,
- programmi lugemise hõlpsus,
- programmi redigeerimise lihtsus,
- programmi testimise lihtsus, sest programmi erinevaid osi ehk plokke saab testida eraldi,
- protsessi määratlemise ja programmeerimise lihtsus.

Jaotatud programmeerimist kasutakse nt. programmeeritavate kontrollerite ja teiste kõrgkeeles programmeeritavate seadmete juures. Näiteks SIMATIC S7 kontrolleritarkvara võimaldab jaotatud programmeerimist, milles programmi üksikuid operatsioone ehk jaotisi nimetatakse plokkideks. SIMATIC S7 kasutajaprogramm koosneb plokkidest (nn. kasutaja programmplokkidest), käskudest ja aadressidest. Tabel 3.1 [12] annab ülevaate sellest, missuguseid plokke saab STEP 7 keeles kasutada ja millised on nende funktsioonid.

Enne kasutaja programmploki töötlemist peab toimuma vastav päring ehk *siire*. Programmi siirdumiseks kasutatakse STEP 7 erikäske. Plokki siirdumist saab programmeerida mõne teise ploki sees. Sellist siirdumist ühest plokist teise võib võrrelda siirdumisega (ülem)programmist alamprogrammi. Iga selline siire tähendab ploki vahetust. Päringu esitanud ploki (kust siirduti) aadress säilitatakse operatsioonisüsteemis, et alamprogrammi lõppedes saaks siirduda tagasi põhiprogrammi. Plokkidesse siirdumise korda ehk järjestust nimetatakse *hierarhiaks*. Teisiti öeldes, korda mis määrab, millised plokid võivad olla ühtedele plokkidele alam- ja teistele ülemprogrammplokkideks, nimetatakse hierarhiaks (joonis 3.8). Plokkide arv hierarhias sõltub protsessori tüübist.

Hierarhias kutsuvad teisi plokke välja OB-, FB- ja FC-plokid, kusjuures väljakutsutavateks võivad olla FB-, FC-, SFB- ja SFC-plokid.



Joonis 3.8. Hierarhiline jaotatud programmeerimine

Tabel 3.1 STEP 7 keeles kasutatavad põhiplokid [11]

Plokid	Kirjeldus
Juhtplokid *OB*	Juhtplokid määravad kasutajaprogrammi struktuuri. Need plokid võimaldavad sidet operatsiooni süsteemi ja kasutaja programmi vahel. Nad juhivad kontrolleri tööle käivitamist, programmi tööd tsüklilises ja katkestustalitluses ning veatöötlust. Siia kuuluvad nt. • OB1 - programmi tsükliline täitmine • OB10..OB17 - kellaajalise või päevalise programmi täitmine • OB20..OB23 - programmi täitmine teatud aja tagant • OB30..OB38 - tsükliline katkestamine teatud aja tagant • OB40..OB47 - riistvara katkestustele reageerimine • OB80..OB87 - vigadele reageerimine jne. Enamasti toimivad juhtplokid koos süsteemsete funktsioonide ja funktsioonplokkidega.
Süsteemsed funktsioon-plokid ja funktsioonid *SFB ja SFC*	Need standardplokid on tootja poolt eelnevalt programmeeritud, kasutajal pole vaja neid programmeerida. SFB'd ja SFC'd paiknevad kontrolleri protsessoris ning neid võib rakendada kasutajaprogrammis vastava päringuga. Kuna need plokid on operatsioonisüsteemi osad, siis erinevalt teistest plokkidest pole neid tarvis laadida kontrollerisse. Mõned näited: • SFB41 "CONT_C" pidevate protsesside juhtimiseks • SFB43 "PULSEGEN", mida kasutatakse koos PID-juhtimisega pulsilaiusmodulatsiooni teostamiseks jne.
Funktsioonid ja funktsioon-plokid *FC ja FB*	Need on loogikaplokid, mida kasutaja peab ise programmeerima. FB-plokkidele on eraldatud teatud mälupiirkond (andmeplokkide näol) parameetrite säilitamiseks. FC-plokkidel millel pole eelmainitud tüüpi mälupiirkonda. Funktsioonide puhul kasutatakse vahemuutujaid ja vahetulemeid, mis funktsiooni töötluse lõppedes kaotavad oma väärtuse, kuid funktsioonplokis saab staatilisi andmeid deklareerida ja hoida muutumatult kuni järgmise andmetöötluseni. Funktsioonplokkide puhul eristatakse *muutujatega* ja *muutujateta* plokke. Muutujatega funktsioonplokkide puhul ei anta sisend-, väljund-, mälu- jne. parameetreid ette aadressidena, vaid muutujatena, millele hiljem saab omistada aadresse. Muutujateta funktsioonplokkides kasutatakse konkreetseid aadresse sisend-, väljund-, mälu- jne. parameetrite iseloomustamiseks.
Andme-plokid *DB*	Need on mälupiirkonnad kasutaja andmete salvestamiseks, säilitamiseks ja lugemiseks. On kahte tüüpi kasutaja andmeplokke: 1. seotud ehk sõltuvad andmeplokid, mis on kindla funktsioonploki või funktsiooniga jäigalt seotud; 2. vabad andmeplokid, mille poole saab pöörduda igast plokist.
Süsteemsed andmeplokid *SDB*	Need andmeplokid on operatsioonisüsteemi enda andmete talletamiseks ja lugemiseks.

3.2.2 Käsurea struktuur

Juhtimisülesanne programmeeritakse üksikute käskude kaupa. Käsk on iseseisev ja sõltumatu programmi osa ning sisaldab juhtsüsteemi tööks vajalikku informatsiooni. Käsk on kasutajaprogrammi väikseim osa nii käsulisti kujul programmeerides kui ka programmimällu salvestatuna. Käsk on määratud käsuvorminguga (joonis 3.9) ning koosneb *operatsiooni koodist* ja *operandist*.

Juhtkäsk		
Operatsioon (mida teha?)	Operand (millega ja kus teha?)	
	Tunnus	Parameeter
U	*E*	*0.5*

Joonis 3.9. Juhtkäsk

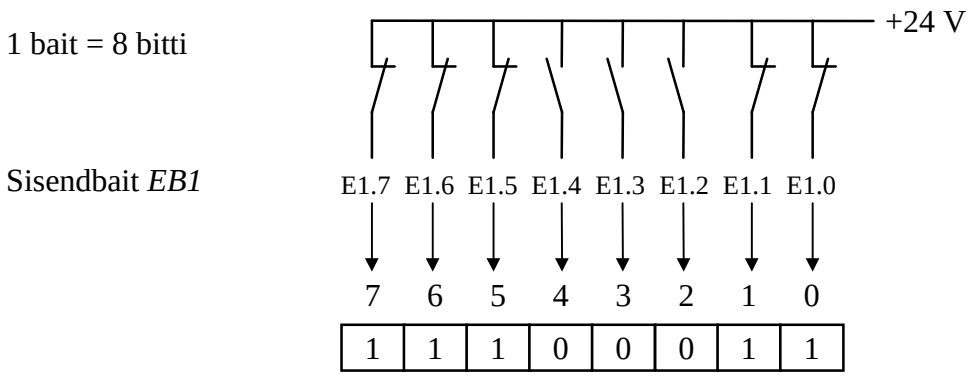
Tähis *U* vastab STEP7 keeles NING-tehtele, *E* vastab sisendile ja *0.5* on aadress. Sõnades võib öelda, et ning sisendis aadressiga 0.5 on olek “1”. Sisendaadressi *E 0.5* korral *0* tähistab sisendbaiti ja *5* selle sama baidi viiendat bitti.

Tabel 3.2 Operandide paiknemine mälus

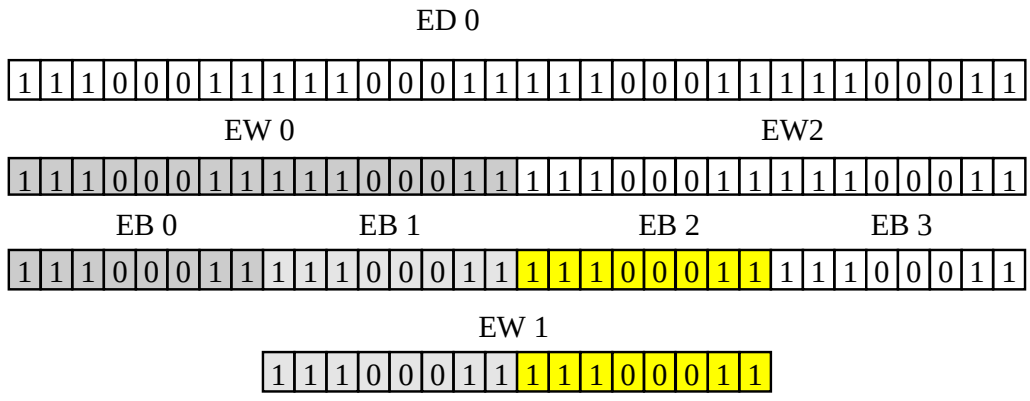
Operand	Mälupiirkond	Kirjeldus
E	Sisendite protsessikuva ehk vahemälu	Tsükli alguses loeb operatsioonisüsteem sisendsignaalide olekud ja asetab need sisendite protsessikuvasse
A	Väljundite protsessikuva ehk vahemälu	Tsükli kestel määrab programm väljundsuurused või -olekud ning asetab need väljundite protsesskuvasse. Tsükli lõppedes loeb operatsioonisüsteem väljundite protsessikuvas paiknevad andmed ning olekud ja kannab need väljunditesse
PE	Perifeersisend	Mälupiirkond otseseks kiireks protsessist mõõdetavate sisendsuurustele juurdepääsuks
PA	Perifeerväljund	Mälupiirkond otseseks kiireks protsessi saadetavate väljundsuurustele juurdepääsuks
M	Märgend	Mälupiirkond vaheolekute operatsioonideks
T	Ajad	Mälupiirkond viivitus- ehk aja-operatsioonideks
Z	Loendurid	Mälupiirkond loendusoperatsioonideks
DB	Andmeplokid	Andmetele reserveeritud mälupiirkond, kuhu võib programmi plokkidest pöörduda
L	Lokaalandmed	Mälupiirkond programmiplokkide (OB, FB, FC) temporaarsete andmete hoidmiseks. Kui programmi ploki töötlus on lõppenud, ei ole need andmed enam kättesaadavad

Käsu operatsiooni- ehk tehtekood määrab, mida tuleb teha. Operand sisaldab operatsiooni ehk juhtkäsu jaoks täpsustavat teavet, s.t. vastab küsimusele "kus ja millega teha?". Operandi moodustavad operandi tunnus ja parameeter. Operandi tunnuseks võib olla sisend (*E*), väljund (*A*), märgend (*M*) jne. erinevates informatsioonihulkades, nagu bitt, bait, sõna ja topeltsõna. Seega operandi tunnus määrab informatsiooni asukoha protsessori mälus (tabel 3.2). Parameeter on operandi aadress, mis määrab informatsiooni asukoha aadressi mälus.

Programmeeritavas kontrolleris kasutatakse bitt-, bait-, sõna- ja topeltsõnaaadresse. Operandi tunnustena kasutatakse sisendi puhul sõltuvalt sõnapikkusest vastavalt tähised *E*, *EB*, *EW* ja *ED*. Programmeeritavas kontrolleris toimivad vaid kahendsignaalid. Olek “0” tähistab pinge puudumist (0 V), olek “1” tähistab pinge olemasolu (24 või 220 V). Igal bitil on oma järjekorranumber ehk aadress. Baidi parempoolne bitt on aadressiga 0, ning vasakpoolne aadressiga 7. Samuti omab aadressi iga bait.
Joonisel 3.10 on toodud sisendbait aadressiga 1 (*EB1*). Baiti kuuluvaid bitte tähistatakse *E 1.0 - E 1.7*. Sisendite ja väljundite tähistamiseks kasutatakse vastavalt saksa keeles tähti *E* (*Eingang*) ja *A* (*Ausgang*) ja inglise keeles *I* (*Input*) ja *Q* (*Output*). Sõnade ja topeltsõnade kasutamisel tuleb arvestada, et nad omavad alati neisse kuuluva noorima baidi aadressi (joonis 3.11).



Joonis 3.10. Sisendbait



Joonis 3.11. Sõnavormingud

3.2.3 Andmetüübid ja -vormingud

Programmeerimiskeeles STEP 7 kasutatakse andmeid biti-, baidi-, sõna- ja topeltsõnapikkustena. Järgnevalt vaadeldakse mõningaid andmevorminguid (tabel 3.3), mida saab esitada eri andmeühikutes.

Tabel 3.3 Andmeühikud ja -vormingud

Andmeühik	Andmevorming	Piirkonnad
bitt	kahendarv	TRUE, FALSE
bait (8 bitti)	kahendarv, kuueteistkümnendarv, tekst, sümbolid, tähed	2#0…2#11111111 B#16#0…B#16#FF 'A'
sõna (16 bitti)	kahendarv, kuueteistkümnendarv, märgita baidid, märgiga täisarv, kuupäev, ajaintervall (h min s või min s ms) loendussuurus	2#0… …2#1111111111111111 W#16#0…W#16#FFFF B#(0,0)…B#(255,255) -32768…+32767 D#1990-01-01… …D#2168-12-31 S5T#0ms või S5T#10ms… …S5T#2h46m30s C#0… C#999
topeltsõna (32 bitti)	kahendarv, kuueteistkümnendarv, märgita baidid, märgiga täisarv, ujukomaarv ehk reaalarv, +/- märgiga ajaintervall, aeg mitmendast päevast	2#0… 2#11111111111111111111111111111111 DW#16#0……DW#16#FFFFFFFF B#(0,0,0,0)…B#(255,255,255,255) L# -2147483648…L#+2147483647 +1,175494e-38…+3.402823e+38 -1,175494e-38…-3.402823e+38 T# -24d20h31m23s647ms… …T#+24d20h31m23s647ms

Peale elementaarandmetüüpide võimaldab STEP 7 kasutada kompleksandmetüüpe ja kasutaja poolt määratletud andmetüüpe. Kompleksandmetüübid (*date_and_time, string, array, struct, FB'd ja SFB'd)* on andmed pikkusega üle 32 biti või sisaldavad ka teisi andmetüüpe. Kasutaja andmetüübid määrab kasutaja ise ning need võivad sisaldada nii kompleks- kui ka elementaarandmetüüpe. Viimastega saab lähemalt tutvuda käsiraamatute abil.

3.2.4 Käsustik

Iga programmi koostamise aluseks on ülesande kirjeldus, milleks võib olla tehnoloogiaskeem, juhtimise graafskeem või tekstina kirjeldatud algoritm. Ülesande kirjeldus tuleb viia seega kontrollerile mõistetavale kujule ehk *programmeerida*. Programmeerimiseks saab kasutada mitmeid kontrollerile arusaadavaid esitusviise (programeerimiskeeli). Käesolevas peatükis käsitletakse kolme enamkasutatavat programmi esitusviisi ehk keelt – loogikaskeemi, kontaktaseskeemi ja käsulisti.

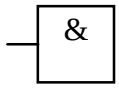
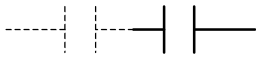
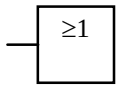
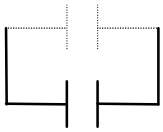
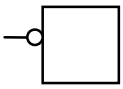
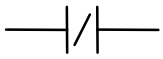
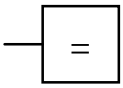
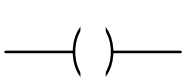
Programmi käskudeks on

- loogikakäsud,
- laadimis- ja siirdekäsud,
- võrdluskäsud,
- loendus- ja viivituskäsud,
- aritmeetikakäsud,
- teisenduskäsud,
- nihkekäsud sh. ringnihkekäsud,
- programmi struktuurikäsud,
- andmeploki talitluskäsud.

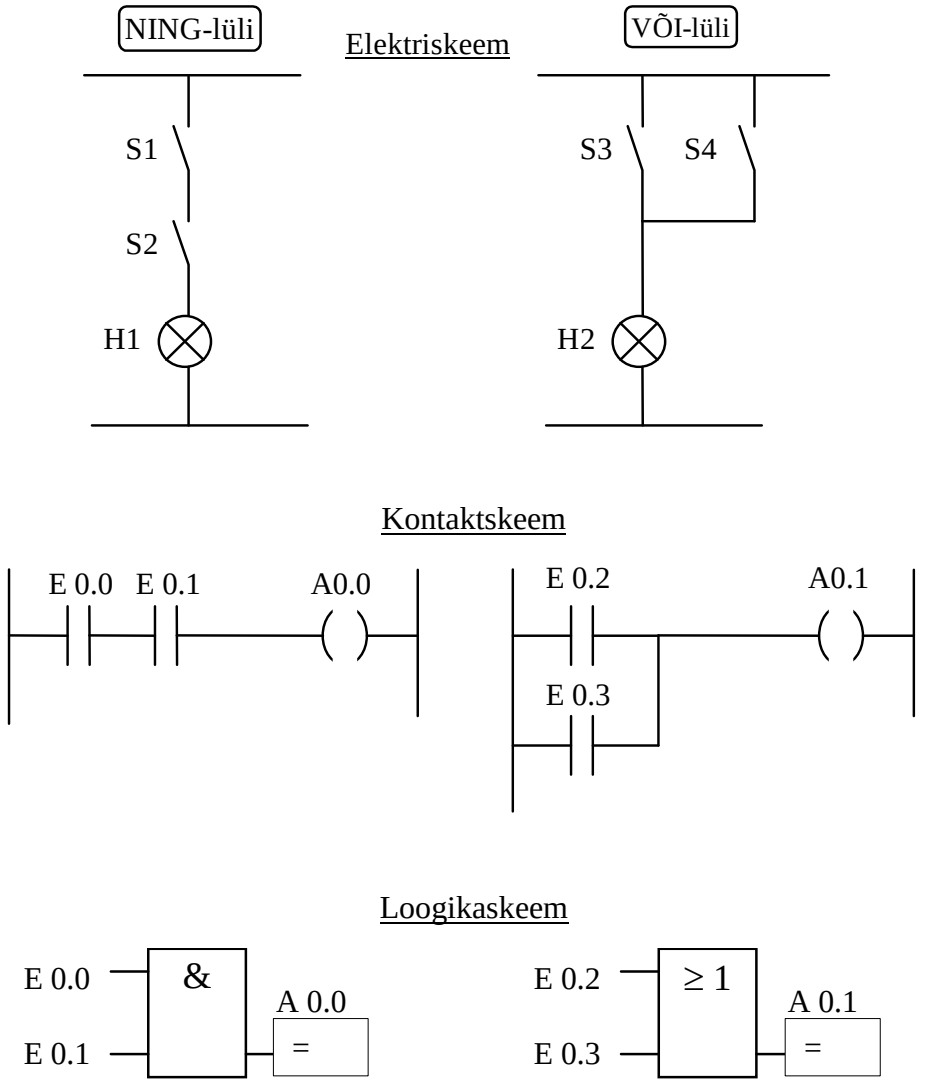
3.2.5 Loogikakäsud

Loogikakäske kasutatakse lihtsate binaarfunktsioonide programmeerimiseks (nt. NING, VÕI, EI ja nende kombinatsioonid). Programmeerimiskeel STEP 7 võimaldab programmeerida kuuel erineval viisil. Järgnevates peatükkides vaadeldakse põhiliselt kolme erinevat programmeerimisviisi – loogikaskeem, kontaktaseskeem ja käsulist (tabel 3.4).

Tabel 3.4 Loogikaelemendid

Loogikaskeem	Kontaktaseskeem	Käsulist	Nimetus
		U	NING
		O	VÕI
		N	EI
		=	VÄLJUND

Joonisel 3.12 on esitatud elektriskeemina, kontaktaseskeemina ja loogikaskeemina NING- ja VÕI-lüli. NING-lüli tööd iseloomustab see, et väljundis on olek “1” ainult siis, kui kõigis sisendites on olek "1". VÕI-lüli tööd iseloomustab see, et väljundis on olek "1", kui kasvõi ainult ühes sisendis on olek "1". Sisend ja väljundahelate kohale kirjutatakse operandide koodid. Kui võrrelda kahte joonist omavahel, võib öelda, et S1-le vastab kontrolleri sisend aadressiga 0.0, S2-le vastavalt sisend aadressiga 0.1 ning H1-le väljund aadressiga 0.0.



Joonis 3.12. NING- ja VÕI-lüli eri esitusviisides

Tabelis 3.5 on näidatud NING- ja VÕI-lüli esitus käsulisti kujul. Tabelis esitatud käsulist on analoogiline programmaatori ekraanil esitatuga.

Tabel 3.5 Käsulist

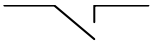
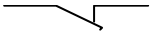
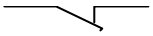
Aadress	Operatsioon	Operand		
		Tunnus	Parameeter	
0000	U	E	0.0	
0002	U	E	0.1	NING-lüli
0004	=	A	0.0	
0006				
0008	O	E	0.2	
000A	O	E	0.3	VÕI-lüli
000C	=	A	0.1	
000E	BE			

Peale eelmainitud lihtloogikakäskude võimaldab STEP 7 keel kasutada ka VÄLISTAV-VÕI käsku (tähistatakse käsulistina esitamisel **X** ja loogikaskeemina esitamisel plokki **XOR**) ja VÄLISTAV-VÕI-EITUS käsku (tähistatakse **XN** käsulistis). VÄLISTAV-VÕI lüli väljundis on sel juhul olek “1” kui ainult ühes sisendis on olek “1”. VÄLISTAV-VÕI-EITUS lüli väljundis on ainult sel juhul olek “1” kui kõigis lüli sisendites on ühesugune olek st. kas “0” või “1”.

3.2.6 Sulguvad ja avanevad kontaktid

Programmeerimisel on tähtis teada, kas kasutatavatel täituritel (nt. releedel) või anduritel on sulguvad või avanevad kontaktid. Kui kontrolleri sisendis on sulguv kontakt, siis selle kontakti rakendumisel on sisendis olek "1". Avaneva kontakti ühendamisel kontrolleri sisendiga, selle kontakti rakendumisel on sisendis olek "0". Kontroller ei võimalda mingil moel kindlaks määrata, kas sisendisse on ühendatud avanev või sulgev kontakt. Kontroller tunneb ära ainult oleku sisendis, st. kas "0" või "1". Seega küsitakse operandi olekut, kas operandi olek on "0" või "1".
Oleku "1" päringuks kasutatakse käsku *U* (*und – NING*) või *O* (*oder – VÕI*) ning oleku "0" päringuks *UN* (*und nicht – NING-EI*) või *ON* (*oder nicht – VÕI-EI*).
Järgmine tabel 3.6 aitab selgitada avatud ja suletud kontaktide kasutamist kontrolleri sisendis ja väljundis.

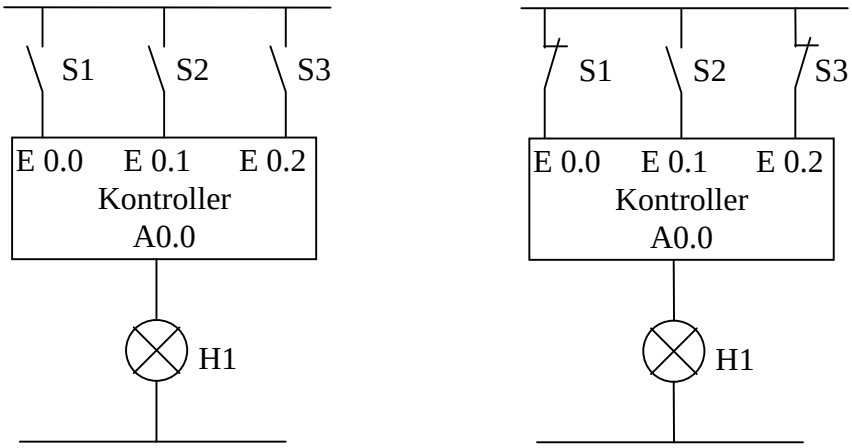
Tabel 3.6 Suletud ja avatud kontaktid

Anduri ehk kontakti tüüp	Andur ehk kontakt	Pinge sisendis	Signaali olek sisendis
Sulguv kontakt	on rakendunud	on	1
	ei ole rakendunud 	ei ole	0
Avanev kontakt	on rakendunud 	ei ole	0
	ei ole rakendunud 	on	1

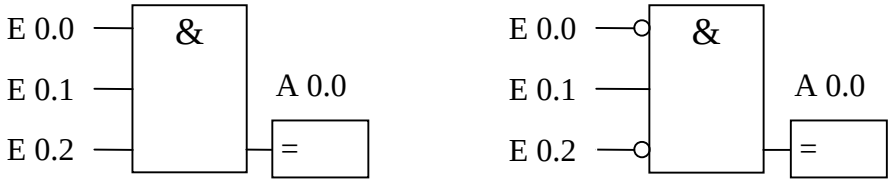
Märkus: Standardi VDE 0113 kohaselt tuleb masina peatamiseks lülitada tema toitepinge välja. Niisugune peatamine on turvaline, sest toimib ka maaühenduse, juhtmekatkestuse või pingekatkestuse korral. Seetõttu ***kasutatakse seiskamis- ja piirlülitites avanevaid kontakte ja neid ei tohi mingil juhul asendada sulguvate kontaktide ehk "0" signaali olekupäringuga programmis***.

Näide
Ülesanne on järgmine: lamp H1 (joonis 3.13) peab süttima, kui kontaktid S1, S2 ja S3 on rakendunud. Koostada tuleb vastav loogikaskeem, kontaktskeem ja käsulist. Ülesande lahendus on toodud joonisel 3.13 elektriskeemina.

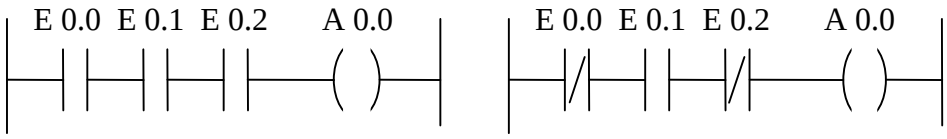
Elektriskeem



Loogikaskeem



Kontaktaseskeem



Käsulist

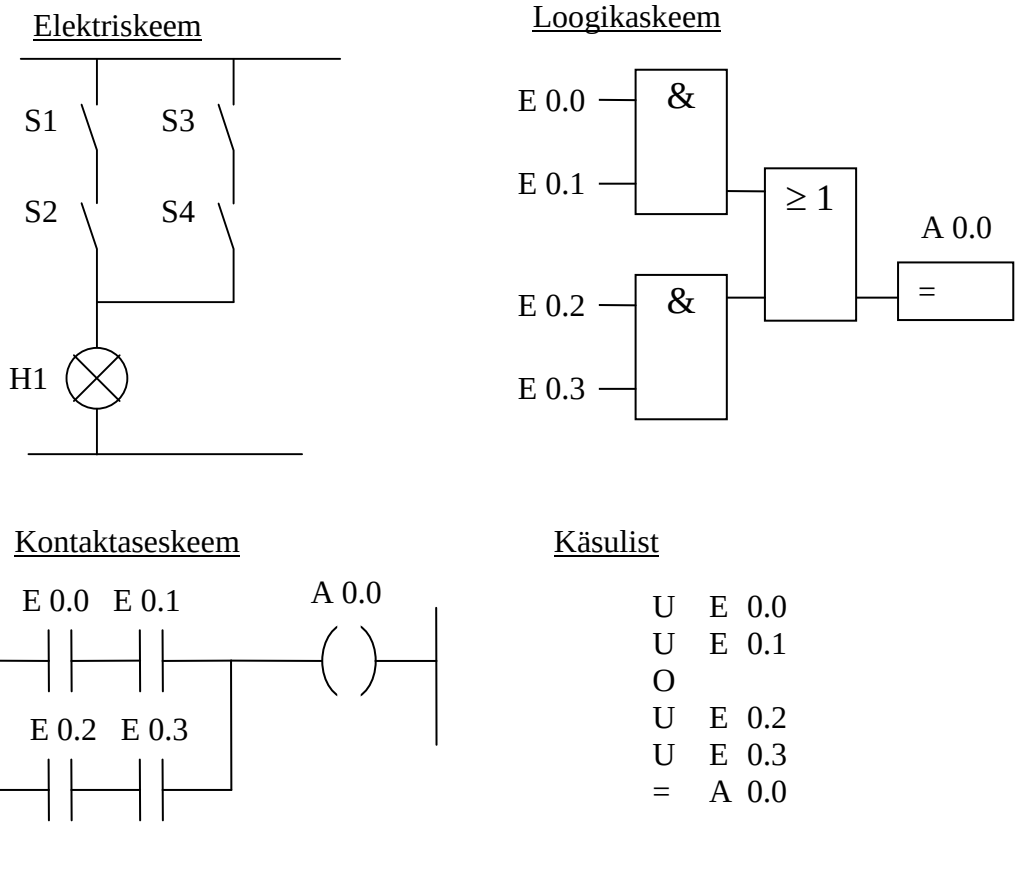
```
U   E  0.0
U   E  0.1
U   E  0.2
=   A  0.0
```

```
UN  E  0.0
U   E  0.1
UN  E  0.2
=   A  0.0
```

Joon 3.13. Avanevate ja sulguvate kontaktide kasutamise näide

3.2.7 NING-VÕI-loogikaühend

Programmeerimise puhul eristatakse termineid *NING-VÕI-loogikaühend (NV)* ja *VÕI-NING-loogikaühend (VN)*. NV kujutab endast rööbiti (VÕI-lüliga) ühendatud NING-lülisid, kus kõigepealt teostatakse NING-tehe ja seejärel VÕI-tehe, nagu on näha joonisel 3.14 elektriskeemina ja kolmes erinevas programmeerimise viisis.

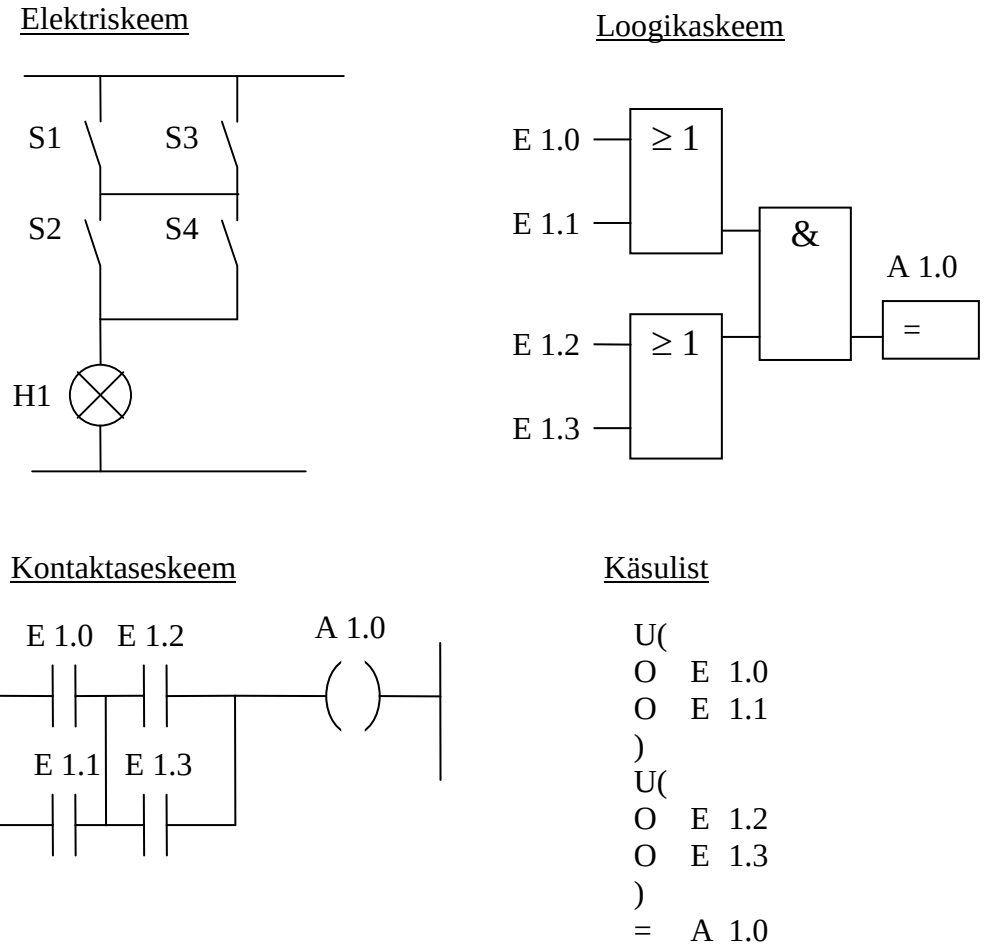


Joonis 3.14. NING-VÕI-loogikaühend

Joonisel 3.14 esitatud skeemi tööpõhimõte on järgmine: kui kontaktid S1 ja S2 või S3 ja S4 on rakendunud, põleb lamp H1; muudel juhtudel lamp ei põle.

3.2.8 VÕI-NING-loogikaühend

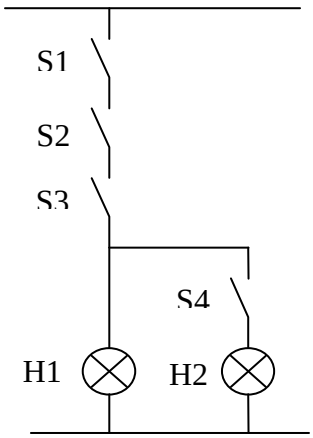
VÕI-NING-loogikaühendi (VN) skeem kujutab endast jadamisi (NING-lüliga) ühendatud VÕI-lülisid. VN programmeerimine ei erine oluliselt loogikaskeemi ja kontaktaseskeemi esitusviisi kasutamise korral NV-st, kuid erineb oluliselt käsulisti kasutamisel (joonis 3.15). VN programmeerimisel käsulistina tuleb kasutada sulgusid, kuna nii määratakse kontrolleri jaoks kindlaks, et VÕI-tehe tuleb teostada enne NING-tehet. Nagu teada, on kahendloogikas VÕI-tehe võrreldes NING-tehtega sekundaarne. Antud juhul teostatakse VÕI-tehe enne NING-tehet; võib enda jaoks põhjendada ja meelde jätta, kus ja millal kasutatakse sulgusid.



Joonis 3.15. VÕI-NING-loogikaühend

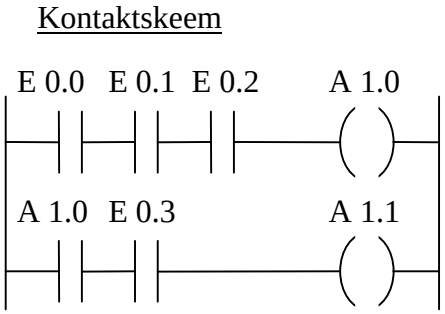
3.2.9 Väljundoperandi kasutamine sisendoperandina

Näide
Signaallamp H1 peab süttima, kui lülitid S1, S2 ja S3 on rakendunud; signaallamp H2 süttib, kui on rakendunud lülitid S1, S2, S3 ja S4 (joonis 3.16).



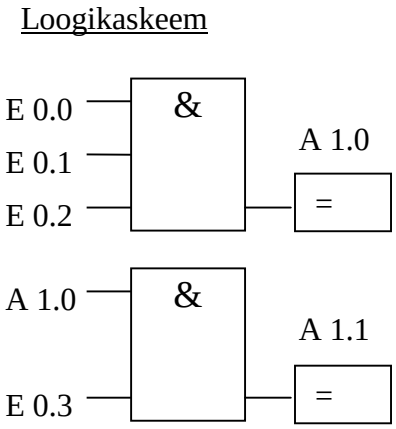
Joonis 3.16. Elektriskeem

Kuna SIEMENSi kontrollerite programmeerimisel võib väljundoperandi kasutada ka sisendoperandina, lahendub ülesanne kontaktaseskeemina järgmiselt (joonis 3.17):



Käsulist

```
U   E  0.0
U   E  0.1
U   E  0.2
=   A  1.0
U   A  1.0
U   E  0.3
=   A  1.1
BE
```

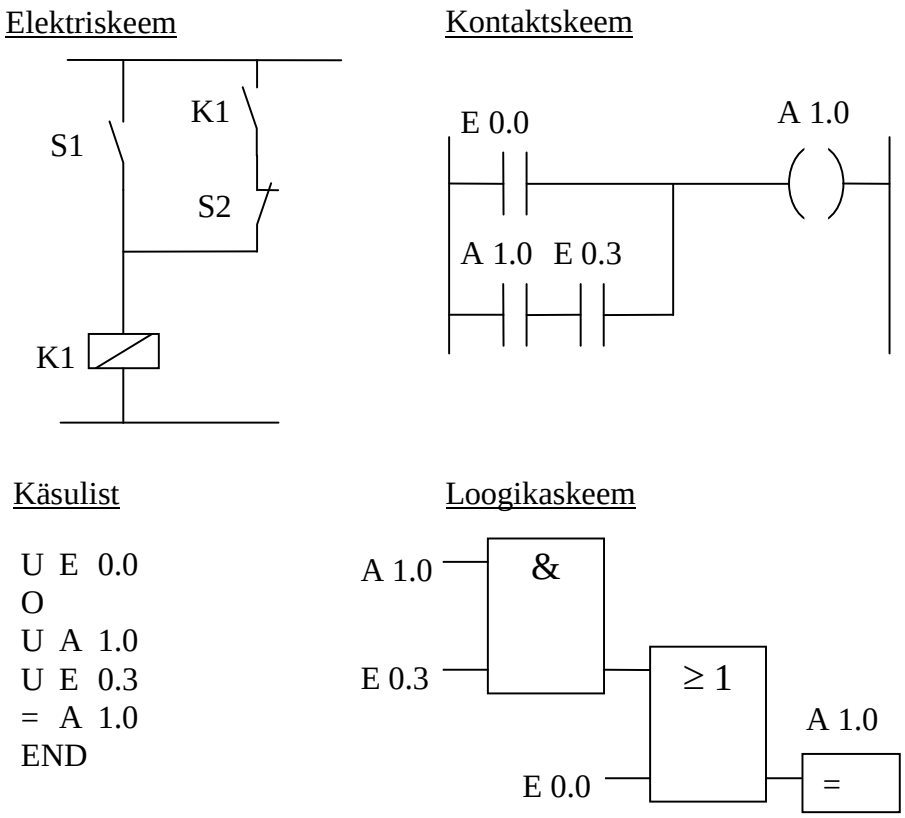


Joonis 3.17. Väljundoperandi kasutamine sisendoperandina

3.2.10 Mälufunktsioonid

Isekäivituskaitse (kaitse mootori isekäivitumise eest pärast pingekatkestust) on kõige lihtsam mälufunktsiooniga seade. Antud seadme puhul eristatakse kahte erinevat tüüpi mälufunktsiooni:

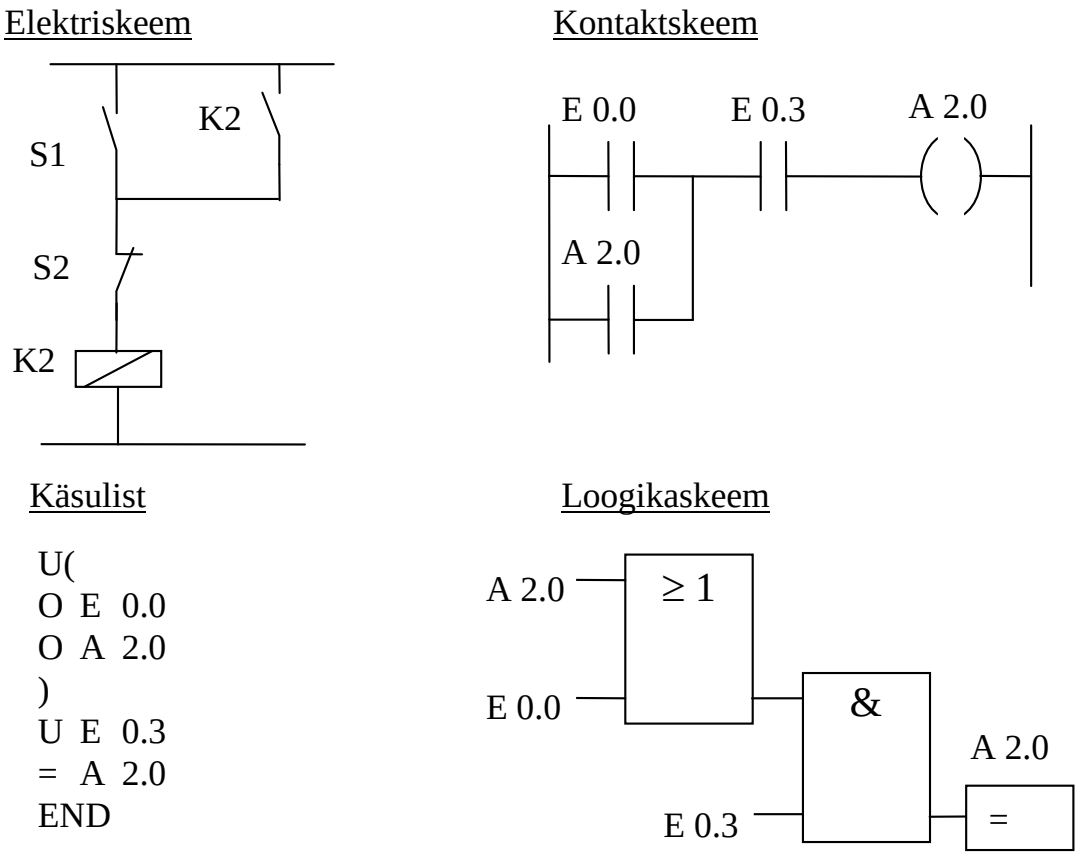
- Ülimusliku *sisselülitamisega* – kui korraga vajutada nii mootori käivitus- (S1) kui ka stoppnuppu (S2), siis kontaktor K1 rakendub ja mootor käivitub (joonis 3.18).



Joonis 3.18. Ülimusliku sisselülitamisega käivitusseade

Antud juhul kontakti S2 (E0.3) esitamiseks programmis ei kasutata “0”-signaali päringut, kuna ***seiskamis- ja piirlülitites kasutatakse avanevaid kontakte ja neid ei tohi mingil juhul asendada sulguvate kontaktide ehk "0" signaali olekupäringuga programmis***.

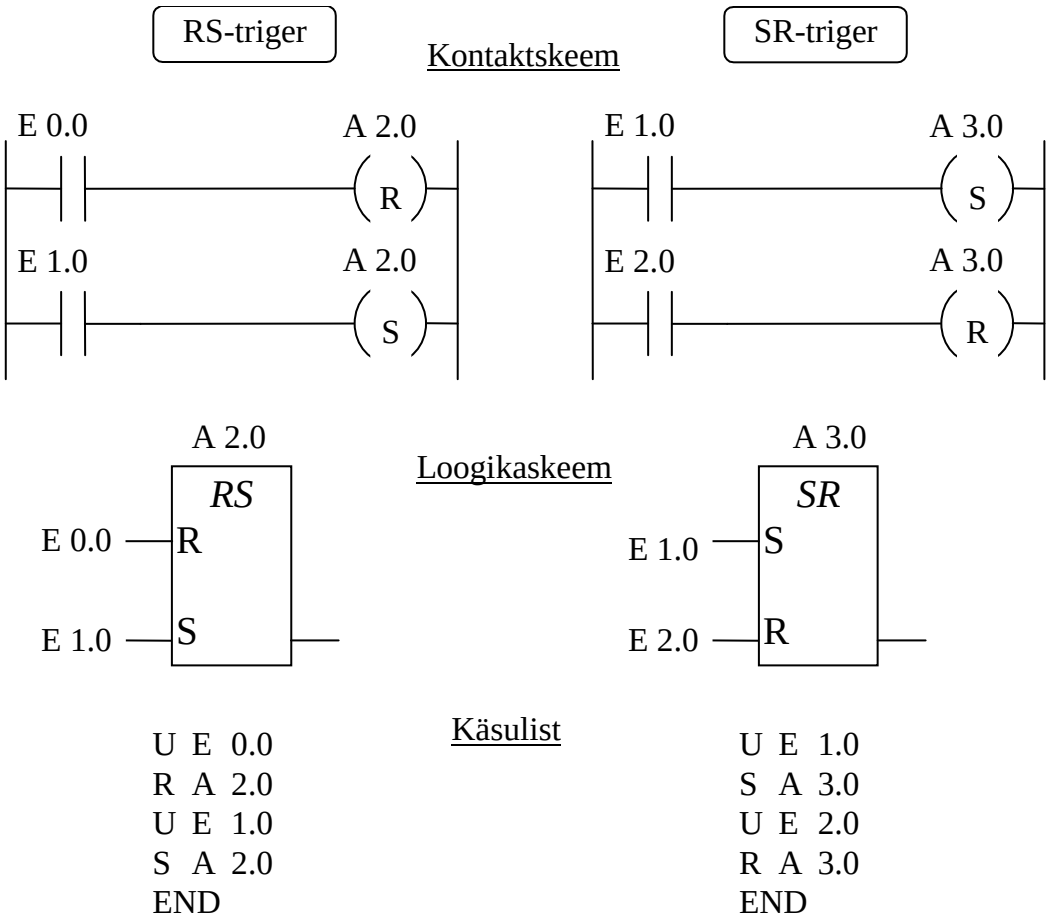
- Ülimusliku *väljalülitamisega* – kui korraga vajutada nii mootori käivitus- (S1) kui ka stoppnuppu (S2), siis kontaktor K2 ei rakendu ja mootor seiskub (joonis 3.19).



Joonis 3.19. Ülimusliku väljalülitamisega käivitusseade

Antud juhul kontakti S2 (E0.3) esitamiseks programmis ei kasutata “0”-signaali päringut, kuna ***seiskamis- ja piirlülitites kasutatakse avanevaid kontakte ja neid ei tohi mingil juhul asendada sulguvate kontaktide ehk "0" signaali olekupäringuga programmis***.

Tuntuim mälufunktsiooniga loogikaseade on triger. Programmeerimiskeeltes STEP 5 ja STEP 7 kasutatakse RS- ja SR-trigerit. RS-triger on ülimusliku sisselülitamisega, SR-triger on ülimusliku väljalülitamisega (joonis 3.20).

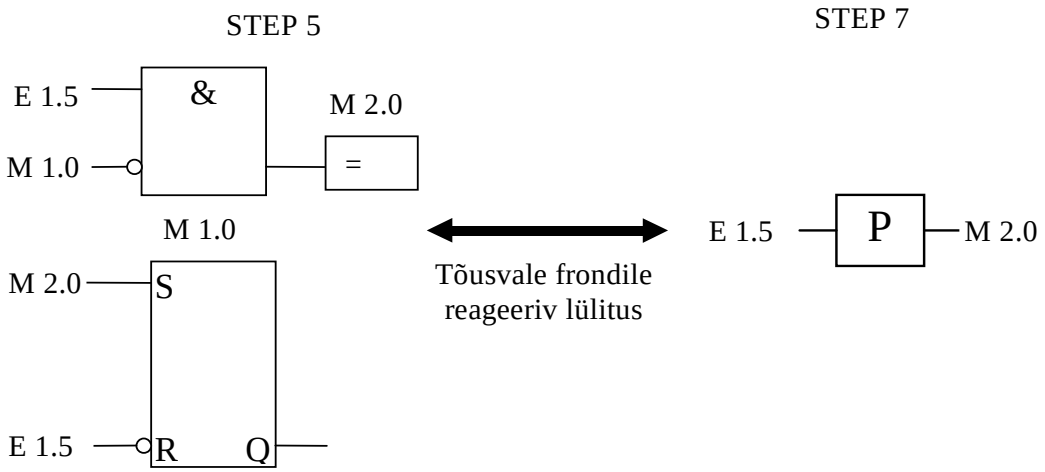


Joonis 3.20. RS- ja SR-trigerid

Mõlemal trigeril seatakse väljund olekusse “1”, kui sisendisse S (*set*) antakse signaal olekuga “1”; väljund seatakse olekusse “0”, kui sisendisse R (*reset*) antakse signaal olekuga “1”. SR-trigerit iseloomustab see, et kui samaaegselt anda trigeri R- ja S-sisendile signaal olekuga “1”, siis trigeri väljund viiakse olekusse “0”. RS-trigeri puhul põhjustab signaal olekuga “1” mõlemas sisendis trigeri väljundi siirdumise olekusse “1”.

3.2.11 Impulsi tõusev ja langev front

Digitaalelektroonikas mõistetakse tõusva frondi all signaali muutust olekust “0” olekusse “1” ja langeva frondi all signaali muutust olekust “1” olekusse “0”. Vanematel kontrolleritel on impulsi tõusvale ja langevale frondile reageeriva ahela programmeerimiseks kasutajal vaja programmeerida vastav ahel (joonis 3.21). STEP 7 keeles on juba tarkvara tootja poolt ette nähtud vastavad käsud.

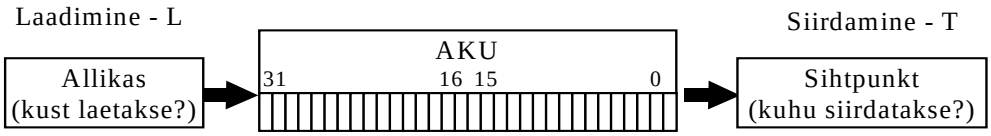


Joonis 3.21. Impulsi tõusvale frondile reageeriv lülitus

Joonisel 3.21 tähistab operand M 1.0 (sks. *Merker*) lippu ehk mälu osa kontrolleris. Antud juhul on tegemist M 1.0 nullinda bitiga mälubaidist 1 ja M 2.0 nullinda bitiga mälubaidist 2. Programmeeritavate kontrollerite puhul võib peale selle kasutada näiteks mälubaidi MB, mälusõna MW ja mälu-topeltsõna MD mäluregistrit. Impulsi langevale frondile reageeriv skeem erineb STEP 5 keeles selle poolest, et NING-lüli sisendisse E 1.5 tuleb ühendada EI-lüli ehk inversioon. STEP 7 keeles tähistatakse langevale frondile reageerivat elementi tähega ***N***. Käsulistis kasutatakse vastavalt positiivse frondi korral käsku ***FP*** ja negatiivse frondi korral ***FN***.

3.2.12 Laadimis- ja siirdamiskäsud

Programmeerimiskeel STEP 7 võimaldab baidi, sõna ja topeltsõna pikkust infovahetust, sisend- ja väljundkaartide, sisend- ja väljundprotsessi kuva, loendurite, taimerite, mälu, andmeplokkide jms. vahel. Selleks kasutatakse laadimis- (***L***) ja siirdamiskäske (***T***). Selline infovahetus ei toimu otse, vaid alati vahemälu (AKU) kaudu. Vahemälu kujutab endast ühte protsessori registrit. Laadimis- ja siirdamiskäsu toimet selgitab joonis 3.22.



Joonis 3.22. Laadimis- ja siirdamistoiming

Seega mingi info laetakse teatud allikast (nt. sisendmoodulist) vahemällu ning siiratakse sealt sihtpunkti (nt. väljundmoodulisse). Infoallikaks või sihtpunktiks võivad olla sisendid, väljundid, mälud, loendurid, taimerid, andmeplokid jne. Laadimis/siirdamiskäsud on *absoluutkäsud* ning neid töödeldakse kontrolleris iga töötsükli jooksul. Neid käske tähistatakse käsulistis sümbolitena ***L*** ja ***T***, millele järgneb operandi tunnus ehk otsene või kaudne aadress, st. kust laetakse ja kuhu saadetakse. Laadimis- ja siirdamiskäsu kasutamist selgitab näide joonisel 3.23.

```
L  EW 0     Laetakse sisendist aadressilt 0 sõnapikkune infohulk ja
T  AW 10    saadetakse väljundisse aadressile 10 sama infohulk

L  MD 10    Laetakse mälust aadressilt 10 topeltsõnapikkune infohulk ja
T  AD 4     saadetakse väljundisse aadressile 4 sama infohulk
```

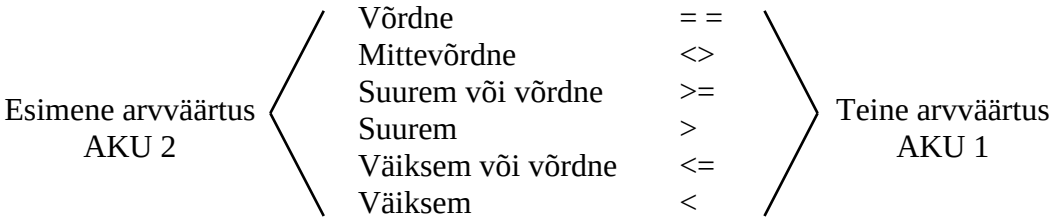
Joonis 3.23. Laadimis- ja siirdamiskäskude näide

Laadimis- ja siirdamiskäskude korral saab operandi tunnustena kasutada

- *protsessikuva* baidi- (EB, AB), sõna- (EW, AW) ja topeltsõnapikkusi (ED, AD) sisend- ja väljundandmeid;
- *välisseadmete* baidi- (PEB, PAB), sõna- (PEW, PAW) ja topeltsõnapikkusi (PED, PAD) sisend- ja väljundandmeid;
- *mälu* baidi- (MB), sõna- (MW) ja topeltsõnapikkusi (MD) andmeid;
- *andmeplokkide* baidi- (DBB, DIB), sõna- (DBW, DIW) ja topeltsõnapikkusi (DBD, DID) andmeid;
- *lokaalseid* (*temporary or dynamic local data*) baidi- (LB), sõna- (LW) ja topeltsõnapikkusi (LD) andmeid.

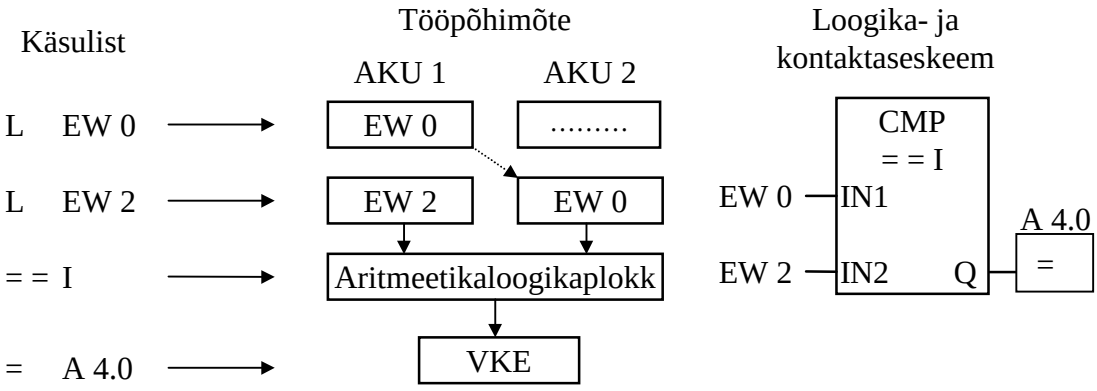
3.2.13 Võrdluskäsud

STEP 7 keeles saab võrrelda operande (arve), mille tunnuseks on sisend, väljund, konstant, mälu ja andmeplokk ja mis on esitatud kas sõna- (***I***) ja topeltsõnapikkuse täisarvu (***D***) või topeltsõnapikkuse reaalarvu (***R***) kujul. Kõigi nende arvutüüpide võrdlemiseks saab kasutada kuut erinevat võrdluskäsku (joonis 3.24), kusjuures võrreldakse esimest arvväärtust teise arvväärtuse suhtes.



Joonis 3.24. Võrdluskäsud

Võrdluskäsu abil võrreldakse kahte AKUs 1 ja AKUs 2 paiknevat kahendsuurust (joonis 3.25). Esimese laadimiskäsuga laetakse esimene operand AKUsse 1. Teise laadimiskäsu täitmisel nihutatakse kõigepealt esimene operand AKUst 1 AKUsse 2 ja seejärel laetakse teine operand AKUsse 1. Seega paikneb nüüd esimesena laetud operand AKUs 2 ja teisena laetud operand AKUs 1. Seejärel võrdleb aritmeetikaloogikaplokk biti kaupa mõlemas AKUs paiknevat arvväärtust. Kui võrdlustehe on tõene, on võrdlustulemuseks ehk lahendiks (VKE) olek “1” ehk joonisel 2.24 väljundi A 4.0 olek on “1”.



Joonis 3.25. Võrdlustehe

Sarnaselt võrdlustehetele toimivad ka aritmeetikakäsud nagu liitmine, lahutamine, korrutamine ja jagamine, mida vaadeldakse punktis 3.2.16.

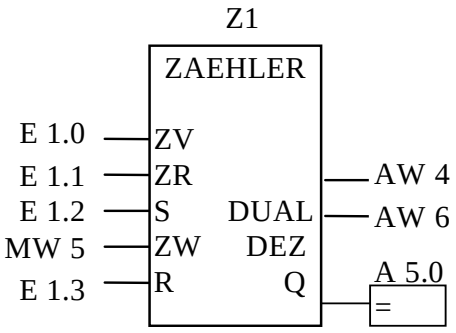
3.2.14 Loenduskäsud

Protsessides on sageli vaja loendada sündmusi või esemeid, nt. ajaimpulsse, tooteid, detaile vms. Sedalaadi loendamiseks kasutatakse programmeerimisekeeles STEP 7 loenduskäske ehk graafiliste esitusviiside puhul loendureid. Loenduritega (joonis 3.26) saab loendada arve alates 0 kuni 999.

Joonisel 3.26 toodud loenduri sisendite ja väljundite omadused on järgmised:

- *ZV* – sisendimpulsi tõusva frondiga suurendatakse loenduri loendustulemit ühe võrra. Loendustulemi ülempiir on 999.
- *ZR* – sisendimpulsi tõusva frondiga vähendatakse loenduri loendustulemit ühe võrra. Loendustulemi alampiir on 0.
- *S* – sisendimpulsi tõusva frondiga alustatakse loendamist (omistatakse loendurile loendustulemile algväärtus, mis paikneb sisendis ZW ning väljund Q viiakse olekusse “1”).
- ZW – sisendisse antakse loendustulemi algväärtus kas konstandina C#algväärtus ehk kümnendarvuna, sisendsõnana EW, väljundsõnana AW, mälusõnana MW või andmesõnana DW.
- R – sisendimpulsi tõusva frondiga lõpetatakse loendamine (loendustulem viiakse nulli ning väljundi Q läheb olekusse “0”).
- DUAL – väljundist saab lugeda loendustulemit kahendarvuna.
- DEZ – väljundist saab lugeda loendustulemit BCD-koodis.
- Q – väljund näitab loenduri olekut. Kui toimub loendamine või loendustulem on suurem kui null, siis olek on “1”. Vastupidi, kui ei toimu loendamist ja loendustulem on null, siis loenduri olek on “0”.

Kontaktaseskeem ja loogikaskeem

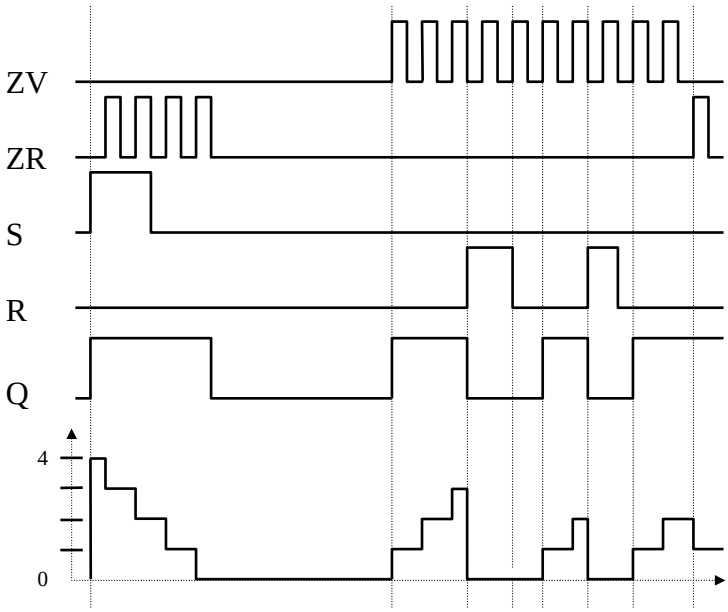


Käsulist

```
Loendustulemi suurendamine
   U   E    1.0
   ZV  Z1
Loendustulemi vähendamine
   U   E    1.1
   ZR  Z1
Algloendustulemi omistamine
   U   E    1.2
   L   MW 5
   S   Z1
Digitaalinfo lugemine väljunditest
   L   Z1
   T   AW 4
   LC  Z1
   T   AW 6
Binaarinfo lugemine väljunditest
   U   Z1
   =   A    5.0
```

Joonis 3.26. Loendur

Loenduri tööpõhimõtet selgitab joonisel 3.27 toodud ajadiagramm, kus sisendisse ZW loetava mälusõna MW 5 väärtus täisarvuna on 4.



Joonis 3.27. Loenduri ajadiagramm

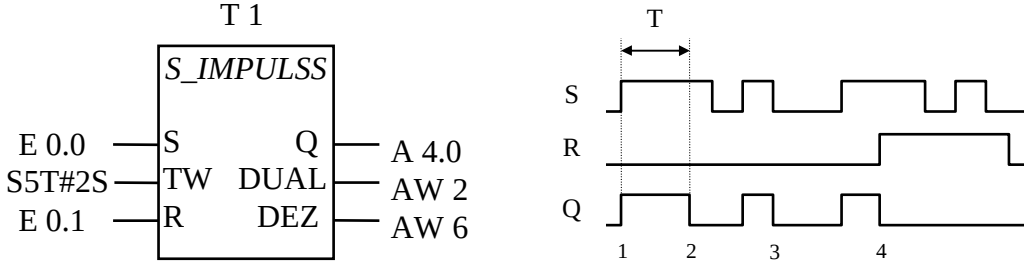
3.2.15 Viivituskäsud

Juhtimisülesannete lahendamisel tuleb väga sageli ühe või teise seadme sisse-, välja- ja ümberlülitamiseks kasutada viivitusi. Kontrollerites on ajafunktsioonid ehk taimerid integreeritud protsessoriplokki. Kasutamisel antakse programmiga ette ajaintervallid, taimerid ja käivitustingimused. Programmeerimisel saab kasutada viit erinevat ajafunktsiooni: impulss-, pikendatud impulss-, viivitusega sisselülitus-, salvestavat viivitusega sisselülitus-, viivitusega väljalülitusfunktsiooni. Kõik mainitud ajafunktsioonid ehk taimerid omavad kolme sisendit ja kolme väljundit, millede otstarve on järgmine:

- *S* – sisendimpulsi tõusva või langeva (sõltuvalt taimeri tüübist) frondiga algab ajaintervalli loendamine.
- *TW* – sisendisse antakse ajaintervalli väärtus kas konstandina *S5T#aeg* vahemikus *S5T#0ms…S5T#2h46m30s*, sisendsõnana EW, väljundsõnana AW, mälusõnana MW või andmesõnana DW.
- *R* – sisendimpulsi tõusva frondiga lõpetatakse ajaintervalli loendamine ning väljund Q läheb olekusse “0”.
- *Q* – väljund näitab taimeri olekut ehk vastava ajafunktsiooni tulemust.
- *DUAL* – väljundist saab lugeda ajaintervalli jooksvat väärtust kahendarvuna.
- *DEZ* – väljundist saab lugeda ajaintervalli jooksvat väärtust BCD-koodis.

Loetletud ajafunktsioonide realiseerimist STEP 7 keeles ja nende erisusi selgitab alljärgnev.

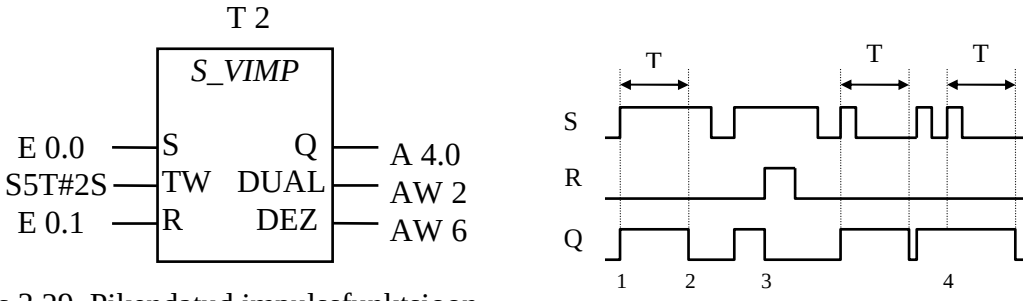
Impulssfunktsioon (joonis 3.28). Ajaintervalli *T* loendamine algab S sisendisse antava impulsi tõusva frondiga ja väljund Q läheb koheselt olekusse “1” (1). Kui sisendimpulss S on etteantud ajaintervallist pikem, loetakse ta lõpuni ja väljund Q läheb seejärel olekusse “0” (2). Kui sisendimpulss S on lühem (3) kui etteantud ajaintervall või sisendsignaal R (4) omab tõusvat fronti, katkestatakse ajaintervalli lugemine ja väljund läheb olekusse “0”. Seda ajafunktsiooni tähistatakse sümboliga *SP* (STEP 5 keeles SI).



Joonis 3.28. Impulssfunktsioon

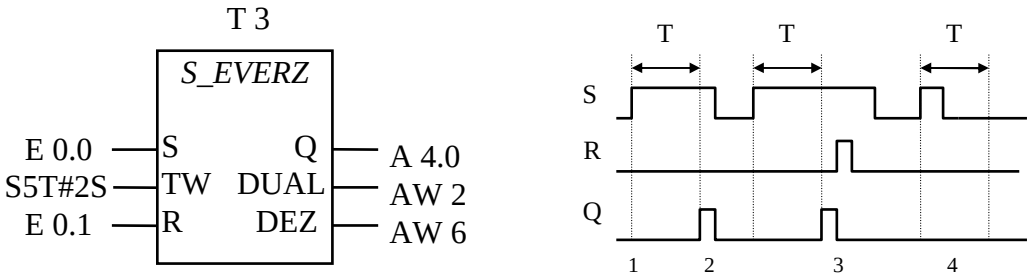
Pikendatud impulssfunktsiooni (joonis 3.29) korral hakatakse aega T lugema S-sisendisse antava impulsi tõusva frondi puhul ja väljund Q läheb olekusse “1” (1). Võrreldes lihtsa impulssfunktsiooniga ei sõltu ajaintervalli lugemine ja väljundsignaali oleku “1” kestus sisendisse S antava signaali kestusest, kuid sõltub R sisendisse antava signaali olekust. Väljundi Q signaal läheb olekusse “0”, kui

etteantud ajaintervalli loendamine on lõppenud (2) või kui R-sisendis on tuvastatud tõusva frondiga (3) signaal. Mitme jadamisi etteantud ajaintervalli jooksul S-sisendisse saabunud impulsi korral hakatakse aega lugema viimase impulsi tõusvast frondist (4). Seda ajafunktsiooni tähistatakse sümboliga *SE* (STEP 5 keeles SV).



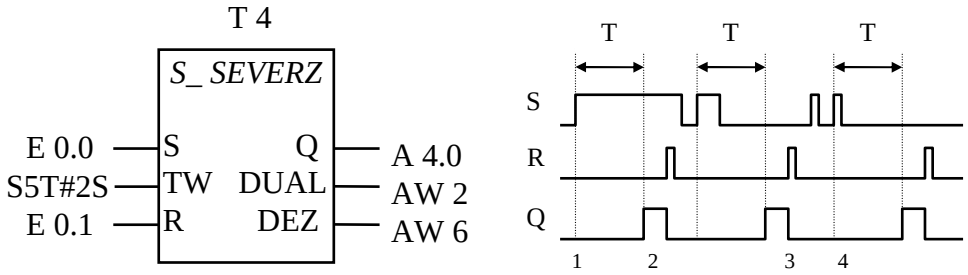
Joonis 3.29. Pikendatud impulssfunktsioon

Viivitusega sisselülitusfunktsiooni (joonis 3.30) kasutamisel hakatakse ajaintervalli *T* loendama S-sisendimpulsi tõusva frondi korral (1). Väljund Q läheb ajaintervalli möödudes olekusse “1”, kui sisendimpulsi kestus on etteantud ajaintervallist (2) suurem. Kui S-sisendsignaali olek “1” kestab etteantud ajast vähem, siis väljund Q ei lähe olekusse “1” (4). Väljund Q viiakse olekusse “0” S-sisendsignaali viimisega olekusse “0” pärast etteantud ajaintervalli möödumist (2) või R-sisendsignaali tõusva frondiga (3). Seda ajafunktsiooni tähistatakse sümboliga *SD* (STEP 5 keeles SE).



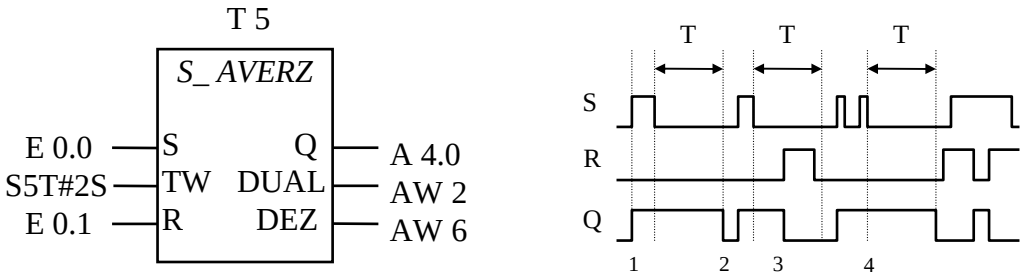
Joonis 3.30. Viivitusega sisselülitusfunktsioon

Salvestava viivitusega sisselülitusfunktsiooni (joonis 3.31) kasutamisel hakatakse ajaintervalli T loendama S-sisendimpulsi tõusva frondi korral (1) ja väljund Q läheb ajaintervalli möödudes olekusse “1” (2). Erinevus võrreldes eelmisega seisneb selles, et väljundit Q saab viia olekust “1” olekusse “0” ainult R-sisendimpulsi tõusva frondiga (3). Mitme jadamisi etteantud ajaintervalli jooksul S-sisendisse saabuva impulsi puhul hakatakse aega lugema viimase impulsi tõusvast frondist (4). Seda ajafunktsiooni tähistatakse sümboliga *SS* (STEP 5 keeles SS).



Joonis 3.31. Salvestav viivitusega sisselülitusfunktsioon

Viivitusega väljalülitusfunktsiooni (joonis 3.32) saab kasutada mingi seadme viivitusega väljalülitamiseks. S sisendimpulsi tõusva frondi korral läheb väljund Q koheselt olekusse “1” (1). Väljund Q läheb olekusse “0”, kui S-sisendsignaali langevast frondist on möödunud etteantud aeg (2) või kui R-sisendis on tuvastatud tõusva frondi (3) signaal. Ajaintervalli *T* jooksul mitme jadamisi S-sisendisse saabunud impulsi korral alustatakse viivituse lugemist viimase impulsi langevast frondist (4). Seda ajafunktsiooni tähistatakse sümboliga *SF* (STEP 5 keeles SA).



Joonis 3.32. Viivitusega väljalülitusfunktsioon

Ajafunktsioonide käsulistina programmeerimine on tunduvalt keerukam kui kontaktaseskeemi või loogikaskeemina. Joonisel 3.28 toodud loogika- ja kontaktaseskeemile vastav käsulist on esitatud joonisel 3.33 näidatud tabeli vasakus tulbas. Sümbol **SP** tähendab, et kasutatakse impulssfunktsiooni, teiste funktsioonide kasutamisel kirjutatakse selle sümboli asemele vastavalt **SE**, **SD**, **SS** või **SF**. Tähis *T1* määrab kasutatava ajafunktsiooni järjekorranumbri, joonistel 3.28-3.32 on tähised *T1*-*T5*. Seega joonisel 3.29 esitatava skeemi käsulistina esituse puhul tuleks **SP** asemel kirjutada **SE** ja *T1* asemele *T2* (joonis 3.33 parempoolne tulp) jne. Rasvase kirjaga tähised kehtivad STEP 7 korral ning võivad STEP keele teistes versioonides olla teistsugused.

Käsulist

U E 0.0 L S5T#2s **SP** *T1*	Taimeri käivitus	U E 0.0 L S5T#2s **SE** *T2*
U E 0.1 R *T1*	Taimeri oleku ennistamine (*reset*)	U E 0.1 R *T2*
U *T1* = A 4.0	Tulemi väljastamine binaarkujul	U *T2* = A 4.0
L *T1* T AW 2	Tulemi väljastamine kahendarvuna	L *T2* T AW 2
LC *T1* T AW 6	Tulemi väljastamine kümnendarvuna	LC *T2* T AW 6

Joonis 3.33. Joonised 3.28 ja 3.29 käsulistina

3.2.16 Aritmeetika-, teisendus-, nihke- ja loogikatehete käsud

Neid käske käsitletakse siinkohal põgusalt:

- ***Aritmeetikakäsud*** võimaldava matemaatilisi põhitehteid sõna- ja topeltsõnapikkuste täisarvude ning ujukomaarvudega. Näiteks liitmine *+I (plus integer), +D (plus double integer), +R (plus real),* lahutamine *-I, -D, -R,* korrutamine (**I, *D, *R*) ja jagamine (*/I, /D, /R*). Aritmeetikakäskude kasutamine sarnaneb võrdluskäskudega (joonis 3.34); esmalt laetakse akusse 1 (AKU1) esimene suurus, siis teine suurus mille tulemusel esimene suurus nihkub akusse 2 (AKU2); teostatakse tehe AKU1 ja AKU2 vahel ja tulemus siiratakse ehk kirjutatakse kas väljundisse, mällu või andmesõnasse jne. Peale selle võimaldab STEP 7 keel kasutada ka mõningaid trigonomeetria- ja erifunktsioone, nagu *SIN, COS, TAN, ASIN, ACOS, ATAN, EXP, LN, SQR* ja *SQRT*.

```
L     MD10   Mälu topeltsõna MD10 (ujukoma arvu kujul) laetakse AKUsse 1
SIN          Leitakse AKUs 1 paikneva ujukoma arvust siinus ja salvestatakse
             see AKUsse 1
T     MD14   AKU1 sisu kirjutatakse mälu topelt sõnasse 14
```

Joonis 3.34. Trigonomeetriafunktsiooni SIN näide

- ***Teisenduskäsud*** võimaldavad eri pikkustega (sõna, topeltsõna) arvude teisendusi ühest arvusüsteemist teise. Näiteks 2/10nd süsteemist täisarvuks *BTI (binar-code-decimal to integer)* ja *BTD (binar-code-decimal to double integer),* täisarvust 2/10nd süsteemi *ITB* ja *DTB,* täisarvust ujukoma *DTR (double integer to real)*

(joonis 3.35), reaalarvust täisarvuks koos ümardamisega kas väiksemaks arvuks *RND-* või suuremaks arvuks *RND+* jne.

```
L     MD20   Mälu topeltsõna MD20 (täisarvu kujul) laetakse AKUsse 1
DTR          Teisendatakse AKUs 1 paiknev topeltsõna pikkune täisarv
             ujukoma arvuks ja salvestatakse see AKUsse 1
T     MD14   AKU1 sisu kirjutatakse mälu topelt sõnasse 14
```

Joonis 3.35. Teisenduskäsu DTR näide

- ***Nihkekäsud*** võimaldavad nihet paremale *SRW (shift right word*) (joonis 3.36), *SRD (shift right double word)* ja vasakule *SLW, SLD* ning ringnihketehet paremale *RRD (rotate right double word)* ja vasakule *RLD* vastavalt sõnade ja topeltsõnadega.

```
L       EW10   Sisendsõna EW10 laetakse AKUsse 1
SRW 5          AKU1 sees paiknevad bitid nihutatakse 5 koha võrra paremale
T       AW10   AKU1 sisu kirjutatakse väljundsõnasse AW10
```

Joonis 3.36. Nihkekäsu SRW näide

- ***Loogikatehete käsud*** võimaldavad loogikatehteid nagu VÕI, NING ja VÄLISTAV-VÕI, kuid erinevus seisneb selles, et tehe teostatakse sõnade või topeltsõnade sama järku bittide vahel. VÕI-tehe *OD (or double word)* ja *OW (or word)*, NING-tehe *AD* ja *AW*, VÄLISTAV-VÕI tehe *XOD* (*exclusive or double word)* ja *XOW (exclusive or word)* (joonis 3.37).

```
L       EW10   Sisendsõna EW10 laetakse AKUsse 1
L       EW12   Sisendsõna EW12 laetakse AKUsse 1, mille tulenusel
               EW10 nihkub AKUsse 2
XOW            Teostatakse VÄLISTAV-VÕI tehe AKU1 ja AKU2 sama
               järku bittide vahel ning tulemus laetakse AKUsse 1
T       AW10   AKU1 sisu kirjutatakse väljundsõnasse AW10
```

Joonis 3.37. Loogikatehte XOW näide

Nende käskude täpsemaks tundmaõppimiseks ja rakendamiseks on vaja kasutada käsiraamatut.

3.2.17 Programmi struktuurikäsud

Programmeerimiskeele STEP 7 struktuurikäskude all mõistetakse plokisiseseid siirdekäske ja plokkidevahelisi plokisiirdekäske ehk plokkide päringukäske.

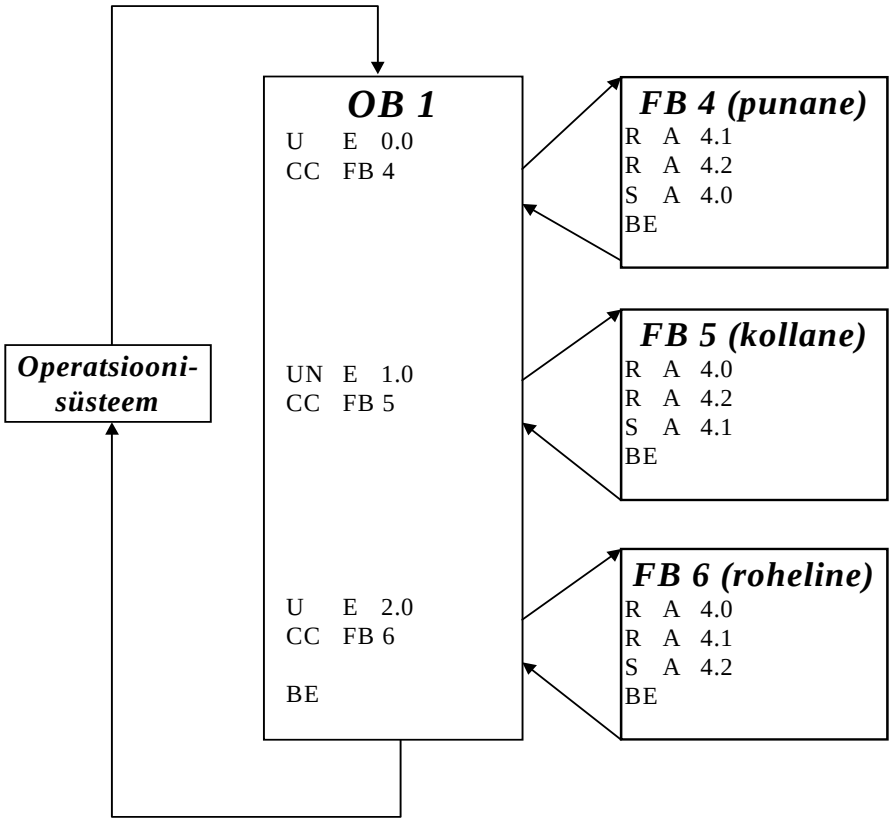
Keeruka protsessi juhtimisülesannet on lihtsam programmeerida, kui see jaotada osaprotsessideks ehk *plokkideks*. Alati on vaja lahendada küsimus, kuidas määrata osaprotsessid, et need moodustaksid terviku. Võtkem näiteks valgusfooride juhtimine. Nagu teada, on valgusfooril 3 tuld (punane, kollane, roheline). Teada on ka see, et valgusfoori tuled ei sütti korraga, s. t. iga tule süttimiseks on vastav tingimus. Seega koosneb valgusfoori juhtimine mitmest väiksemast alamprotsessist. Iga alamprotsessi käivitab omakorda mingi tingimus. Sellist programmeerimist, kus alamprotsesside kirjeldamiseks kasutatakse eraldi plokke, nimetatakse *jaotatud* programmeerimiseks.

STEP7 ja STEP 5 keeles kasutatakse ülemisel tasandil juhtplokina *OB*-tüüpi plokke, s. t. esimesena pöördub süsteem nende poole ja alles seejärel vastavalt teiste programmis (juhtplokis) määratletud plokkide poole. Üks tähtsamaid juhtplokke on OB1, mis on mõeldud tsükliliseks programmjuhtimiseks. Süsteem pöördub OB1 poole tsükliliselt, iga teatud ajavahemiku tagant. Plokkidevahelisteks siireteks ja neis sisalduvate programmide täitmiseks on plokkidevahelised ja plokisisesed siirdekäsud (tabel 3.7). Plokkidevaheliste siirete teostamiseks kirjutatakse käsu järele ploki nimi kuhu tuleb siirduda. Plokisiseste siirete teostamiseks kirjutatakse käsu järgi vastav märgendi nimetus, mis määrab ära millisele märgendiga tähistatud käsureale siire toimub.

Tabel 3.7 Siirdekäsud

STEP 7		STEP 5
Plokkidevahelised siirdekäsud	Kirjeldus	ingl.(saksa) keeles
BE	ploki lõpp	BE
BEU	ploki tingimuseta lõpp	BEU (BEA)
BEC	ploki tingimuslik lõpp	BEC (BEB)
CALL plokk	plokki siire	JU n (SPA n)
CC plokk	tingimuslik plokki siire	JC n (SPB n)
UC plokk	tingimusteta plokki siire	JU n (SPA n)
Plokisisesed siirdekäsud	Kirjeldus	Käsk
JU märgend	tingimuseta siire	JU märgend (SPA)
JC märgend	tingimuslik siire	JC märgend (SPB)
JZ märgend	siire nulli korral	JZ märgend (SPZ)
JM märgend	siire miinusmärgi korral	JM märgend (SPM)
JN märgend	siire mitte nulli korral	JN märgend (SPN)
JP märgend	siire plussmärgi korral	JP märgend (SPP)
LOOP märgend	tsükliline tingimuslik siire	-

Jaotatud programmeerimist aitab mõista valgusfooride näide (joonis 3.38). Et süsteem pöörduks tsükliliselt programmi algusesse tagasi, on vaja kasutada ülemise nivoo plokina juhtplokki OB 1.

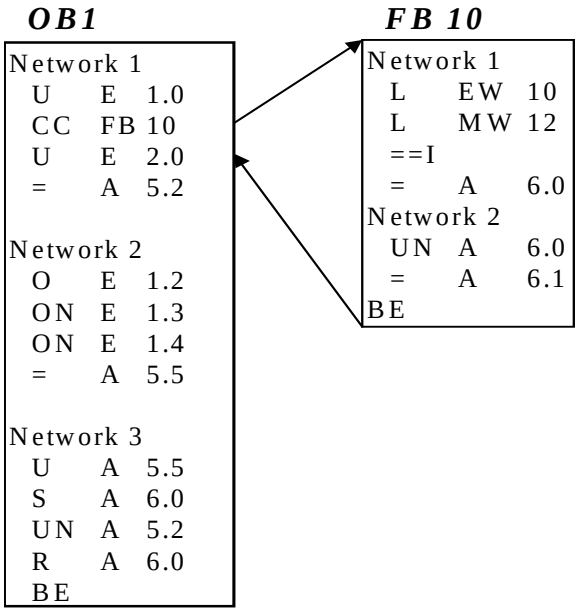


Joonis 3.38. Valgusfooride programmi jaotatud programmeerimine plokkidena

Programm töötab järgmiselt. Süsteem pöördub tsükliliselt juhtploki OB1 poole. Kui sisendis E 0.0 on signaali olek “1”, toimub siire funktsioonplokki FB 4. Seega kui sisendis E 0.0 on signaali olek “1”, viiakse väljund A 4.1 ja A 4.2 olekusse “0” ning väljund A 4.0 olekusse “1”. Jõudes käsuni BE, pöördutakse tagasi juhtplokki OB1 järgmise käsu juurde jne.

Märkus
Kõik plokid jaotatakse omakorda väiksemateks üksusteks, mida nimetatakse skeemideks või segmentideks (*network*, *segment*). Iga segment võib sisaldada mingit programmi osa. Näiteks käsulistina programmeerimise puhul võib iga plokk sisaldada kuni 999 segmenti ja iga segment omakorda kuni 2000 käsurida. Nii nagu süsteem pöördub tsükliliselt ehk korduvalt, iga teatud aja järel, juhtploki või funktsioonploki poole, niisamuti töödeldakse igas tsüklis jadamisi programmi segmente (segment 1, seejärel segment 2 jne.). Joonisel 3.39 on toodud kolme segmendiga juhtplokk OB1 ja kahe segmendiga funktsioonplokk FB 10. Funktsioonploki (ja temas sisalduvaid segmente) töödeldakse, kui sisend E 1.0 on olekus “1”.

Programmi struktuurikäskude (ka. **MCR** ehk *Master Control Relay* käsu) kohta leiab täpsustavat informatsiooni käsiraamatutest.



Joonis 3.39. Programmiplokkide jaotus segmentideks

3.2.18 Andmeploki talitluskäsud

Andmeplokke käsitletakse siinkohal lühidalt. Nende kohta saab leida täpsemat teavet käsiraamatutest. STEP 7 keeles kasutatakse juhtploki, funktsioonploki või funktsiooni sidumiseks sõltumatuid (*DI*) või sõltuvaid ehk lokaalseid (*DB*) andmeplokke (käsk ***OPN*** *andmeploki nimi*). STEP 5 keeles kasutatakse andmeploki sidumiseks funktsioonplokkidega käsku (***C*** *andmeploki nimi*) *(saksa keeles A).*

Andmeploki pikkuse (*DBLG*, *DILG*) ja numbriga (*DBNO*, *DINO*) opereerimiseks kasutatakse laadimis- ja siirdamiskäske *L* ja *T* (joonis 3.40.)

OB1

OPN	DB10	Avatakse andmeplokk DB10
L	DBLG	Laetakse andmeploki pikkus akusse 1
L	+10	Laetakse täisarv 10 akusse 1, mille tulemusel andmeploki pikkus nihkub akusse 2
<=I		Võrreldakse neid kahte arvu; kui andmeploki pikkus on väiksem kümnest,
S	M 10.0	viiakse lipp M10.0 olekusse “1”
U	M 10.0	Kui lipp M10.0 on olekus “1”,
L	EW 0	laetakse sisendsõna 0
T	DBW5	ja kirjutatakse ta andmeploki sõna 5 kohale
CALL	FB10, DB20	Selle käsu tulemusel siirdutakse funktsioonplokki, mis kasutab andmete säiltamiseks ja lugemiseks sõltumatut andmeplokiga

Joonis 3.40. Näide andmeploki kasutamisest

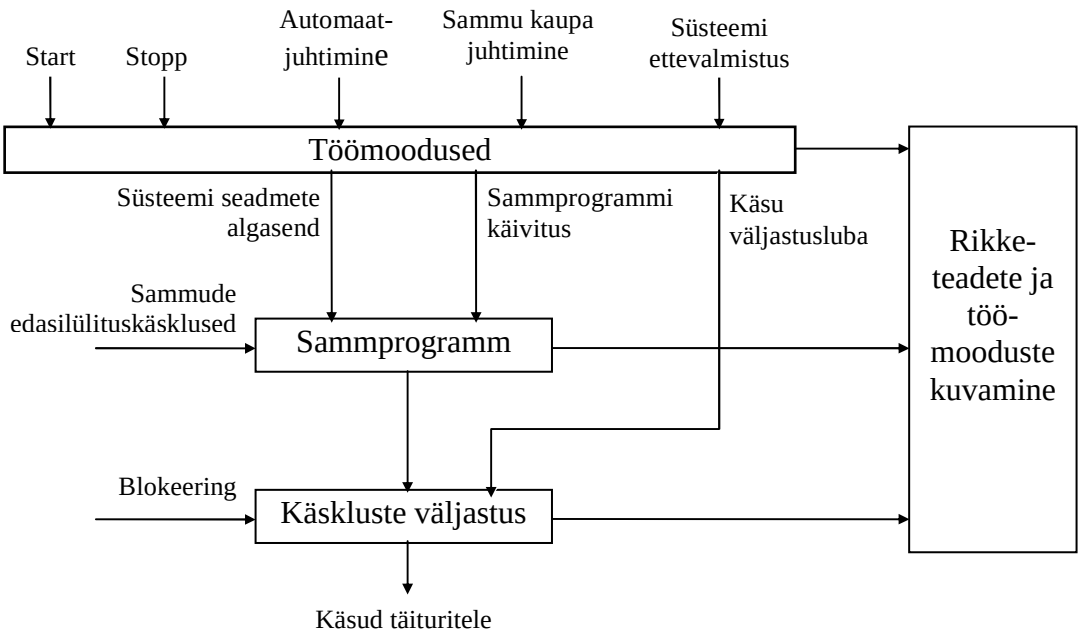
3.2.19 Sammprogrammjuhtimine

Sammprogrammjuhtimiseks nimetatakse juhtimist, kus tervikprotsessi osaprotsesse täidetakse kindlas järjekorras samm-sammult.

Sammprogrammjuhtimise eelisteks on

- projekteerimise ja programmeerimise lihtsus ning aja kokkuhoid,
- lihtne programmi muudetavus,
- juhtimisvigade ning -häirete lihtne ja kiire avastamine.

Sammprogrammjuhtimine koosneb eri töömooduste valiku, sammprogrammi ja juhtkäskude väljastamisplokist ning töömooduste indikatsiooni ja rikke kuvarist (joonis 3.41).



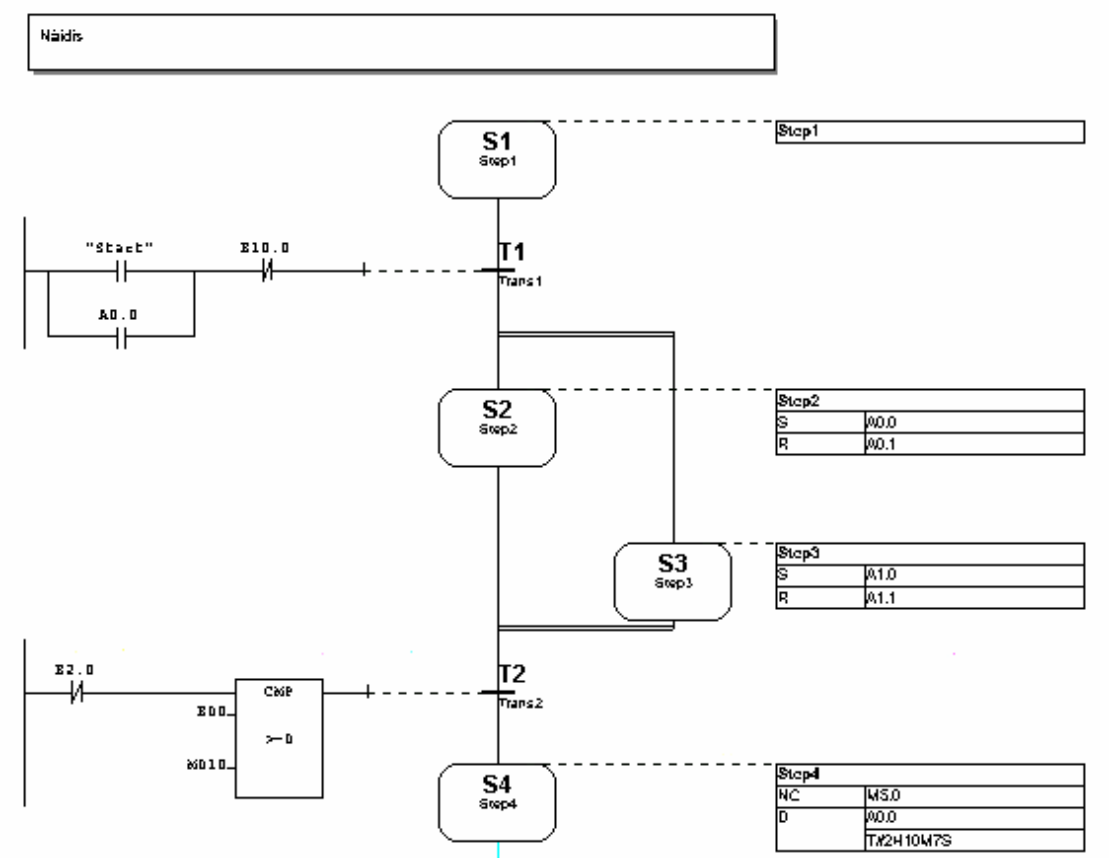
Joonis 3.41. Sammprogrammjuhtimine

Töömoodused valitakse ümberlüliti või juhtnupuga, millega käivitatakse sammprogramm või väljastatakse täiturseadmetele vastav käsk. Kuvarile ilmub aktiivne töömoodus. Eristatakse kolme tähtsaimat töömoodust:

1. Automaatjuhtimine - kogu sammprogramm töötab ilma kasutaja sekkumiseta.
2. Sammuhaaval juhtimine - programmi täidetakse kasutaja poolt samm-sammult.
3. Käsijuhtimine - võimaldab eri täiturite juhtimist sõltumatult ülejäänud programmist.

Sammprogrammi täidetakse sammuhaaval s. t. üksikud sammud ehk osaprotsessid täidetakse jadamisi sõltuvalt edasilülitustingimustest. Programmi täitmisel sõltub käskluste väljastamine alati töömooduse valikust, startlülitist ja blokeeringu lülitist tulevatest signaalidest. Kuvar näitab programmi töös tekkivaid vigu, eri töömoodustes täituritele väljastatavaid käsklusi ja antud ajahetkel aktiivset ehk jooksvat sammu.

Sammprogramm kujutab algoritmi plokkskeemi taolist ketti (joonis 3.42), mis jaotatakse sammudeks ehk osaoperatsioonideks ja tingimusteks (edasilülitustingimusteks) mis on vastava sammu täitmise eelduseks.



Joonis 3.42. Sammprogramm

Programmisammud täidetakse üksteise järel, kusjuures korraga täidetakse ainult ühele tingimusele vastav samm; järgmist sammu ei tohi täita seni kuni eelnev samm pole täidetud. Sammu täitmise tingimuseks võib olla piirlüliti olek, ajafunktsioon, talitlusviisi valik jne. Seega võib edasilülitustingimus sõltuda protsessist ja ajast. Iga sammu täitmise tulemusena antakse juhtkäsk täituri(te)le. Sammprogramme kasutatakse eriti juhtimisprotsessides, kus operatsioonid toimuvad üksteise järel, mitte rööbiti.

Sammprogrammi koostamiseks on mitmeid võimalusi, nt. *graafiline* programmeerimine (STEP 7 keeles programmeerimisviis GRAPH 7) (joonis 3.42). Programmeerimisviisi GRAPH 7 puudumisel võib kasutada teisi olemasolevaid programmeerimisviise nagu kontaktaseskeem, loogikaskeem või käsulist ning luua

nende baasil sammprogramm, kasutades sammude loomiseks mäluelemente. Neid võimalusi siin lähemalt ei vaadelda.

3.2.20 Analoogsignaalid ja pidevjuhtimine

Eelnevates punktides 3.2.1…3.2.15 käsitleti diskreetsete ehk binaarsignaalide töötlemist. Käesolevas punktis vaadeldakse pidevsignaalide töötlemist.

SIEMENS'i kontrollerite SIMATIC S7 puhul on sõltuvalt protsessorist keskplokki (protsessoriplokki) integreeritud peale digitaalsisendite ja -väljundite ka analoogsisendid ja -väljundid. Kui on tegemist keskplokiga, millel puuduvad analoogsisendid ja -väljundid, saab sellele lisada vastava mooduli. Info lugemine ja kirjutamine väljunditesse toimub vastavate käskude ja aadresside abil. Olgu teada, et kõikidesse analoogsisenditesse tulevad analoogsignaalid muundatakse sõltumata kasutaja toimingutest (analoog-digitaal-muunduriga) digitaalsele kujule ning analoogväljunditesse minev signaal muundatakse digitaalselt kujult (digitaal-analoog-muunduriga) analoogkujule. Programmeerija opereerib suuruste digitaalväärtustega. Nii analoogsisendi kui ka -väljundi jaoks on digitaalsuuruse esitus ühesugune, omades mälus kindlat piirkonda. Analoogsuurust esitatakse binaarsel kujul kahendarvuna (kahendkomplemendina). Analoogsuuruse esitus analoogmoodulites on toodud tabelis 3.8 [14].

Tabel 3.8 Analoogsuuruse esitus moodulites

Eraldus-võime	Analoogsuurus															
Bittide arv	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Biti kaal	S	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

Sümbol S (*sign*) tähistab analoogsuuruse märgi kohta, mis paikneb alati 15-ndas bitis, s. t. kahendarvu vanimas vasakpoolses bitis. Kui selle biti olek on "0", on analoogsuuruse märk "+". Olekul "1" on märk "–". Kui analoogsuuruse esituseks (sõltuvalt analoogmoodulitest) kasutatakse vähem kui 15 bitti, s. t. kui analoogmooduli eraldusvõime on alla 15 biti, sisestatakse märk AKU vasakpoolsesse vanema baidi vanimasse bitti. Noorema baidi vastavate bittide positsioonid täidetakse automaatselt olekuga "0". Tabel 3.9 näitab analoogsuuruste digitaalset esitust eri eraldusvõimega analoogmoodulites ning missugustele bittidele (tähistatud sümboliga **x**) omistatakse olek "0", kui analoogsuuruse kirjeldamiseks kasutatakse vähem kui 15 bitist esitust [14]. Eraldusvõime sõltub valitud analoogmoodulist (analoog-digitaal-muunduri kohtade arvust) ja selle parameetritest ning määrab analoogsuuruse täpsuse (tabel 3.9). Mida väiksem eraldusvõime, seda ebatäpsem on reaalse analoogväärtuse esitus. See tähendab, et väljundmooduli väljundpinge või -vool muutuvad suuremate astmete kaupa. Vastupidi, mida suurem on eraldusvõime, seda suurem on täpsus ja seda sujuvamalt muutub näiteks väljundsuurus (pinge, vool).

Tabel 3.9 Analoogmooduli täpsus sõltuvalt eraldusvõimest

Eraldusvõime bittides (+S)	Väärtus ehk ühik (täpsus)		Analoogsuuruse digitaalkuju	
	Kümnendarv	Kuueteistkümnend-arv	Vanem bait	Noorem bait
15	1	1_H	S0000000	00000001
14	2	2_H	S0000000	0000001x
13	4	4_H	S0000000	000001xx
12	8	8_H	S0000000	00001xxx
11	16	10_H	S0000000	0001xxxx
10	32	20_H	S0000000	001xxxxx
9	64	40_H	S0000000	01xxxxxx
8	128	80_H	S0000000	1xxxxxxx

Andurite ning täiturite ühendusviisid ja eri moodulite mõõtepiirkonnad on toodud kataloogides. Analoogsisend- ja -väljundmoodulite tööpõhimõte seisneb selles, et analoog-digitaalmuundurist saadakse digitaalsuurus, mis kantakse mällu ja/või kontrolleri andmesiinile. Kui analoogsisendmoodulil on enam kui üks sisend, siis loetakse ja muundatakse sisendsuurusi üksteise järel jadamisi. Seega loetakse analoogsisendite signaale tsükliliselt ja tsükli kestuse määrab sisendite arv. Analoogiliselt sisendmoodulile toimib ka väljundmoodul, s. t. digitaalsuurused laetakse tsükliliselt vastavasse väljundisse ja muundatakse analoogsignaaliks.

Kasutaja saab analoogsuuruse digitaalsel kujul analoogsisendmoodulist kätte ning saab saata digitaalsuuruse analoogväljundmoodulisse laadimis- (***L***) ja siirdamiskäskudega (***T***). Käsu operandi tunnuseks on vastavalt sisendi korral ***PEW*** ja väljundi korral ***PAW***. Operandi tunnusele järgneb aadress, mis määrab, millise sisend- või väljundsignaali jaoks operatsiooni ehk käsku täidetakse. Seega nt. teades, et analoogsisendi aadress on 128, loetakse tema signaaliväärtust sisendist järgmiselt: ***L PEW 128***. Selleks et kirjutada või saata signaali väärtus digitaalsel kujul väljundisse aadressiga 130, kirjutatakse järgmine käsurida: ***T PAW 130***.

Teades, kuidas lugeda sisendmoodulis analoogväärtust digitaalsel kujul ja kuidas saata analoogväärtust väljundmoodulisse digitaalkujul, on vaja lisaks teada ka seda, kuidas on ette antud juhitava väärtuse muutumise ülem- ja alampiir. Selleks on vaja teada missuguse voolu-, pinge- või temperatuuripiirkonna mõõtmiseks või signaalinivoo väljastamiseks on vastav sisend- ja väljundmoodul ette nähtud. Tuleks lisada, et eri piirkondade ja parameetrite määratlemiseks saab kasutada vastavaid mõõtemooduleid, STEP 7 keele käske ja süsteemseid või standardfunktsioone ning funktsioonplokke. Andmed moodulite ja mõõtepiirkondade kohta on toodud käsiraamatutes. Iga mooduli mõõtepiirkond jaotatakse 5 osaks, milleks on ületäitumis-, ülejuhtimis-, nimi-, alajuhtimis- ja alatäitumispiirkond. Tabelis 3.10 on toodud nt. sisendmooduli 1…5 V ja 4…20 mA mõõtepiirkondade esitus [14].

Tabel 3.10 Analoogväärtuse esitus digitaalkujul sisendmoodulil

Mõõtepiirkond 1…5 V	Mõõtepiirkond 4…20 mA	Ühikud		Piirkond
		Kümnendarv	Kuueteist-kümnendarv	
>5.7036	>22.810	32767	$7FFF_H$	Ületäitumine
5.7036 : 5.001	22.810 : 20.0005	32511 : 27649	$7EFF_H$: $6C01_H$	Ülejuhtimine
5.000 4.000 : 1.000	20.000 16.000 : 4.000	27648 20736 : 0	$6C00_H$ 5100_H : 0_H	Nimipiirkond
0.9999 : 0.2963	3.9995 : 1.1852	-1 : -4864	$FFFF_H$: $ED00_H$	Alajuhtimine
<0.2963	<1.1852	-32768	8000_H	Alatäitumine

Olgu lisatud, et üle- või alatäitumise korral on sisendmoodulis alati analoogsuuruse digitaalväärtus vastavalt $7FFF_H$ või 8000_H. Analoogväljundmoodulis on üle- või alatäitumise korral analoogsuuruse väärtus 0, et kaitsta ülepinge eest sinna ühendatud täitureid. Täpsemaks selgituseks on tabelis 3.11 [14] toodud väljundsuuruse esitus analoogväljundmoodulis.

Tabel 3.11 Väljundsuuruse esitus analoogväljundmoodulis

Mõõtepiirkond ±10 V	Ühikud		Piirkond
	Kümnendarv	Kuueteist-kümnendarv	
0	>32511	$>7EFF_H$	Ületäitumine
11.7589 : 10.0004	32511 : 27649	$7EFF_H$: $6C01_H$	Ülejuhtimine
10.0000 : 0 : -10.0000	27648 : 0 : -27648	$6C00_H$: 0_H : 9400_H	Nimipiirkond
-10.0004 : -11.7589	-27649 : -32512	$93FF_H$: 8100_H	Alajuhtimine
0	<-32512	$<8100_H$	Alatäitumine

Käsiraamatutes esitatud mõõtepiirkondade tabelid (nt. tabelid 3.10 ja 3.11) võimaldavad leida, millisele analoogväärtustele vastab teatud kümnend- või kuueteistkümnendarv. Teades seoseid analoog- ja digitaalsuuruste vahel, on lihtne programmiliselt määratleda juhtimistingimusi. Paragrahvis 5.4 on toodud näide analoogsuuruste kasutamise kohta.

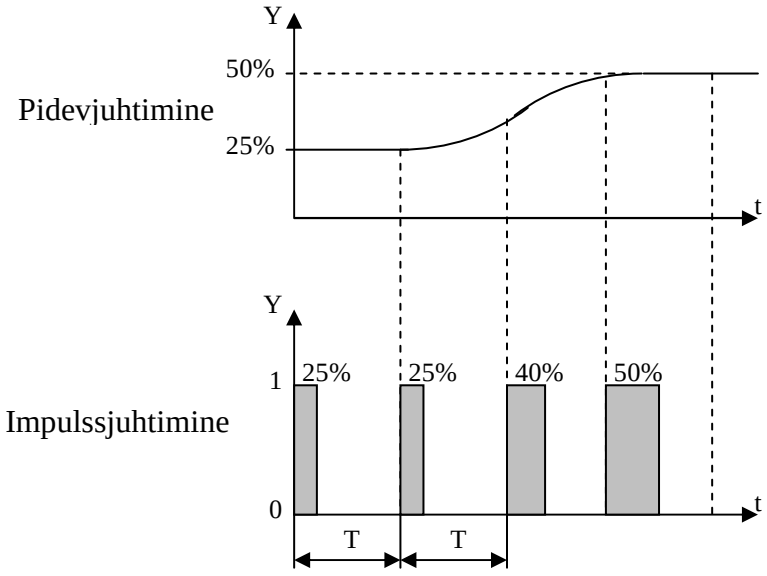
3.3 Reguleerimistarkvara

3.3.1 Reguleerimise alused

Järgnevalt on toodud ülevaade kontrollerites SIMATIC S7/M7/C7 kasutatavate juhtimisviiside ja -struktuuride kohta. Juhtimisviisi valik sõltub täiturist ja seega ka protsessist. Enamik regulaatoreid võimaldavad kolme allpool kirjeldatud reguleerimisviisi:

- Pidevjuhtimine, mille korral (joonis 3.43) mõõdetakse ja võrreldakse tegelikku ja etteande suurust katkematult. Pidevjuhtimisel saab etteandesuuruse iga väärtust etteantud piirkonnas vastu võtta ehk lugeda. Pidevjuhtimisel on väljundsuuruseks analoogsuurus. Sellist regulaatorit kasutatakse täiturite (nt. pingega reguleeritav ventiili) juhtimiseks, mis vajavad katkematut juhtsignaali (Y)..

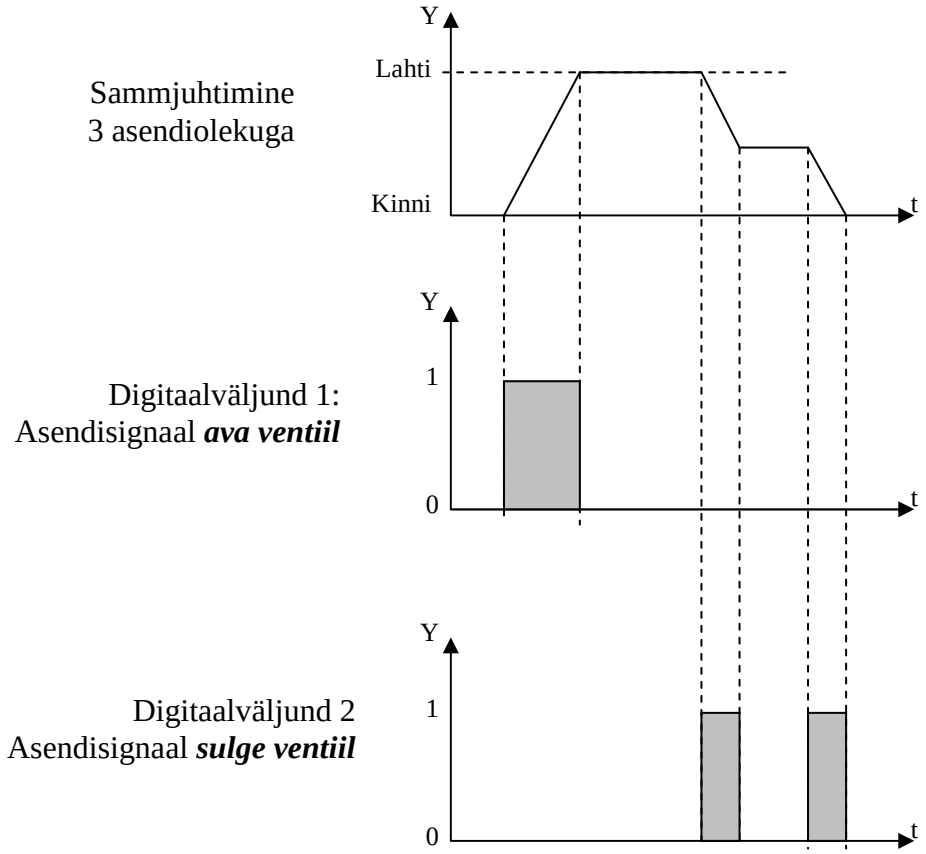
- Impulssjuhtimise korral (joonis 3.43) muudetakse väljundi pidevsignaal impulsside jadaks. Juhttoime tagatakse impulsside ja nende vaheliste pauside suhtega. Enamasti nimetatakse impulssjuhtimist *pulsilaiusmodulatsiooniks*. Impulsside sagedus hoitakse konstantsena, muudetakse vaid impulsi laiust perioodis. Parim näide impulssjuhtimise kohta oleks elektrilise kütteelemendi temperatuuri reguleerimine, kus igale temperatuuri väärtusele vastab kindel impulsi ja pausi pikkuse suhe.



Joonis 3.43. Pidev- ja impulssjuhtimine

- Sammjuhtimine võimaldab aktiveerida 3 lülitusolekut, näiteks liikumine *paremale - seis - vasakule*. Sammregulaatorit kasutatakse nende täiturmehhanismide puhul, mis juhtsignaali puudumisel hoiavad oma olekut, näiteks ventiilide, klappide ja siibrite korral. Ventiili juhtimise näide sammregulaatori abil on toodud joonisel

3.44. Mootor annab vaid siis impulsi, kui etteandesuurus erineb tegelikust suurusest.



Joonis 3.44. Ventiili sammjuhtimise näide

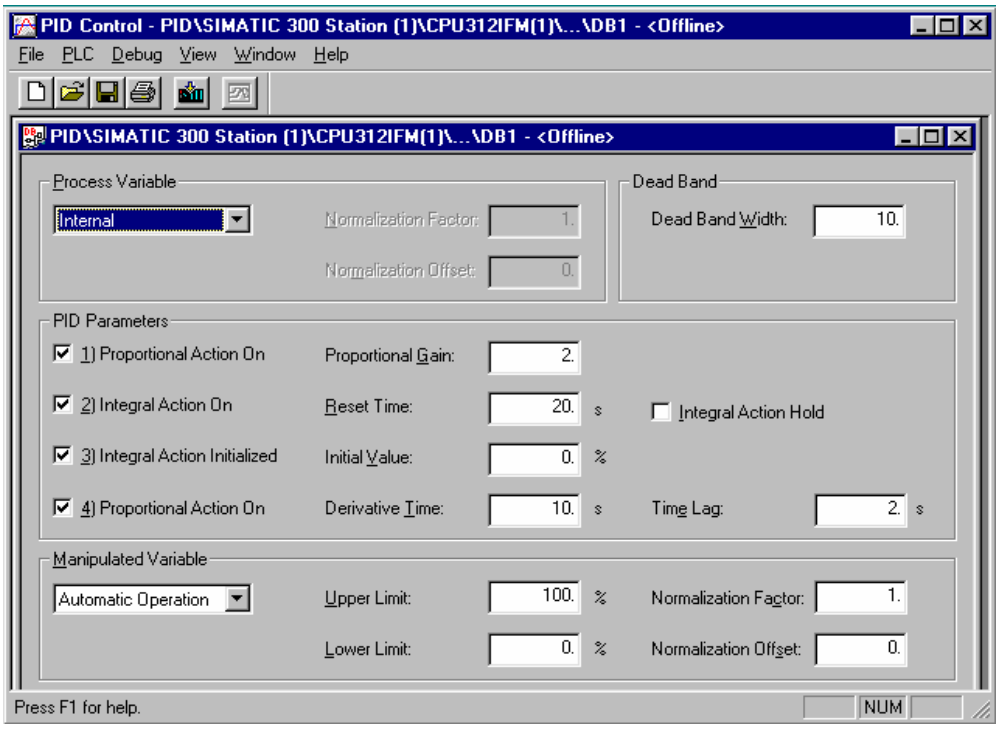
Kontrollerid SIMATIC võimaldavad eelmainitud juhtimisviiside korral kasutada järgmisi juhtimisahelaid ehk –struktuure:

- stabiliseeriv juhtimine - stabiilse, aegajalt muudetava etteandesuurusega (seadesuurusega);
- järgivjuhtimine - tegelik suurus järgib täpselt ja kiiretoimeliselt jooksvalt muudetavat etteandesuurust;
- kaskaadjuhtimine - koosneb jadaregulaatorist, mis edastab ettandesuuruse järgivregulaatorisse, mis reageerib veale ja väljastab toimesuuruse (tegeliku suuruse);
- proportsionaaljuhtimine - hulk protsessi suurusi hoitakse kindlas vahekorras ehk proportsioonis;
- segajuhtimine - reguleeritakse hulka protsessi suurusi, mida hoitakse kindlas proportsioonis omavahel;
- jaotatud juhtimine - etteandesuurus jaotatakse vastavalt mingile funktsioonile mitme täituri vahel (nt. 0…30 % ventiilile 1 ja 30...100 % ventiilile 2)

3.3.2 Reguleerimistarkvara SIMATIC S7/M7/C7 kontrolleritele

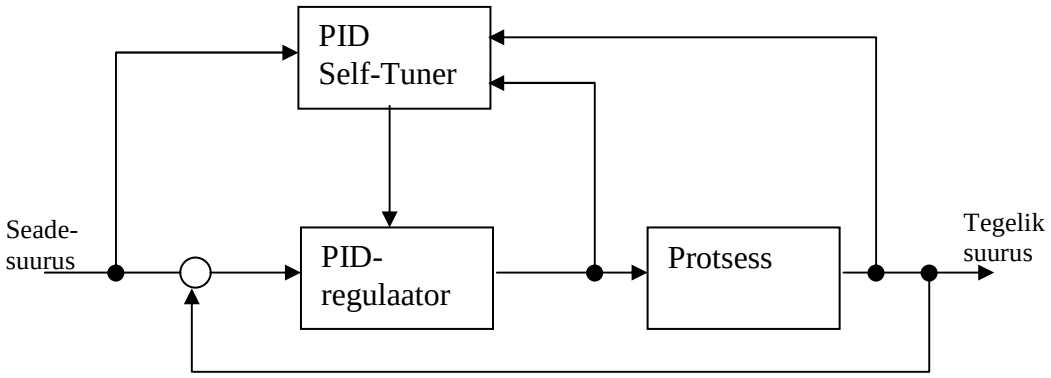
Reguleerimise ja automaatjuhtimise ülesannete lahendamiseks programmeeritavatel kontrolleritel (nt. SIMATIC S7/M7/C7) saab kasutada mitmeid reguleerimisviise. Selle jaoks on firmal Siemens välja töötatud mitu tarkvarapaketti.

- ***PID Control in STEP 7*** sisaldab lihtsaid reguleerimisalgoritme, standardfunktsioonplokke ning PID-parameetrite määramiseks *kasutajaliidest PID Control Parameter Assignment* (joonis 3.45). Kasutatakse enamasti lihtsate temperatuurireguleerimisülesannete lahendamiseks. PID-juhtimine on integreeritud kontrollerite S7-200 jaoks tarkvarapaketti *STEP 7-MicroWin*; samuti on PID-juhtimine kontrollerite S7-300/400 jaoks integreeritud tarkvarapaketti *SIMATIC Manager* (*STEP7*) ja *CFC* ning alates protsessoritest CPU 314 IFM riistvarsse süsteemsete funktsioonplokkidena. Kontrollerite S7-200 iga protsessor CPU võimaldab reguleerida kuni 8 üksteisest sõltumatut ahelat. S7-300/400 korral sõltub reguleeritavate ahelate arv juhtseadme mälu mahust. *PID Control* pakett sisaldab järgmisi funktsioonplokke: pidevjuhtimiseks FB41 “CONT_C”, sammjuhtimiseks FB42 “CONT_S” ja pulsilaiusmodulatsiooniks FB43 “PULSGEN”. Kautajaliides “PID Control Parameter Assignment” võimaldab kõigile eelmainitud funktsioonplokkidele ja kontrolleri protsessorites (alates CPU314IFM) paiknevatele süsteemsetele funktsioonplokkidele vastavad juhtimisparameetrid ette anda. Funktsioonplokke saab ka omavahel kombineerida (nt. FB41 ja FB43) ja luua segajuhtimissüsteeme.



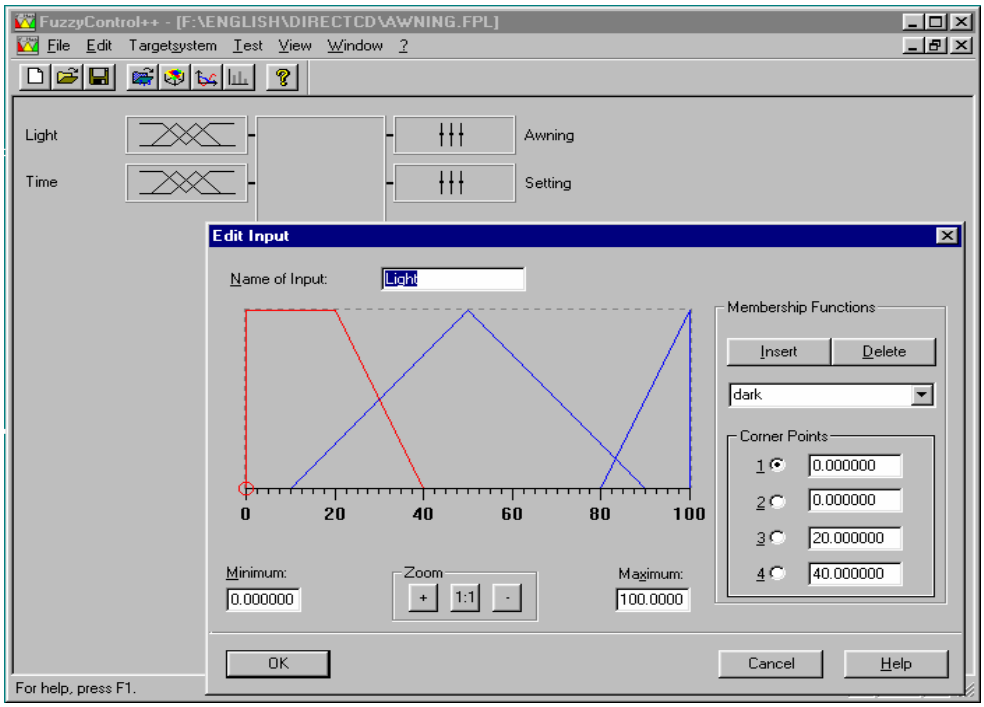
Joonis 3.45. PID-parameetrite määramine STEP7 abil

- ***Standard PID Control*** sisaldab *eelmääratlustega* (sks. *vorkonfektionierte*) reguleerimisstruktuure lihtsate ja keskmise raskusastmega reguleerimisülesannete lahendamiseks nagu temperatuuri, rõhu, voolu jne. reguleerimine. Standard PID Control tarkvarapaketti kasutatakse kontrolleritel S7-300/400 baseeruvate reguleerimisülesannete lahendamiseks toiduaine-, keemia-, farmaatsia-, klaasi-, keraamika- ja masinatööstuses ning kliimatehnika valdkonnas. Standard PID Control võimaldab regulaatori graafilist koostamist ja parameetrite määratlemist.
- ***Modular PID Control*** sisaldab modulaarseid tarkvaraplokke raskete ja mahukate reguleerimisülesannete lahendamiseks, kus nõutakse suurt mälumahtu ja lühikesi lülitusaegu kuid see ei välista tema kasutamist ka väikeste protsesside juhtimiseks. Modular PID Control võimaldab koostada reguleerimisstruktuure SIMATIC-sarja kõikidele kontrolleritele nagu S7, M7 ja C7. Modular PID Control võimaldab regulaatori graafilist koostamist ja parameetrite määratlemist.
- ***PID Self-Tuner*** sisaldab funktsioonplokki PID-regulaatorite *online*-eneseoptimeerimiseks. See funktsioonplokk võimaldab optimeerida PID-regulaatorite parameetreid etteantud jooksva protsessi järgi (joonis 3.46). *PID Self-Tuner* on mõeldud temperatuuri-, vooluhulga- ja nivooregulaatorite optimeerimiseks.



Joonis 3.46. PID-regulaatori optimeerimise põhimõte

- ***Fuzzy Control*** on Windows 95 keskkonnas töötav tarkvarapakett (joonis 3.47), mis kujutab parametreerimistööriista, et mugavamalt sätestada protsessi sisend- ja väljundparameetreid ja sätestatut (koostatud programmi) protsessi juhtimiseks tööle rakendada. Peale selle sisaldab ta standardseid funktsioone, mis lihtsustavad programmeerimist ja sätestamist. Spetsiaalset lisariistvara antud tarkvarale ei ole vaja. Seda paketti kasutatakse keeruliste protsesside juhtimiseks, kus on raske või võimatu protsessi matemaatiliselt kirjeldada; saab juhtida ka neid protsesse, mida varem sai juhtida ainult käsitsi. Protsessi matemaatilisi seoseid pole vaja tunda; piisab teadmistest, kuidas protsess peaks toimima, kui kasutada KUI-SIIS-sõnade kombinatsiooni.



Joonis 3.47. Fuzzy Control ++ tööaken

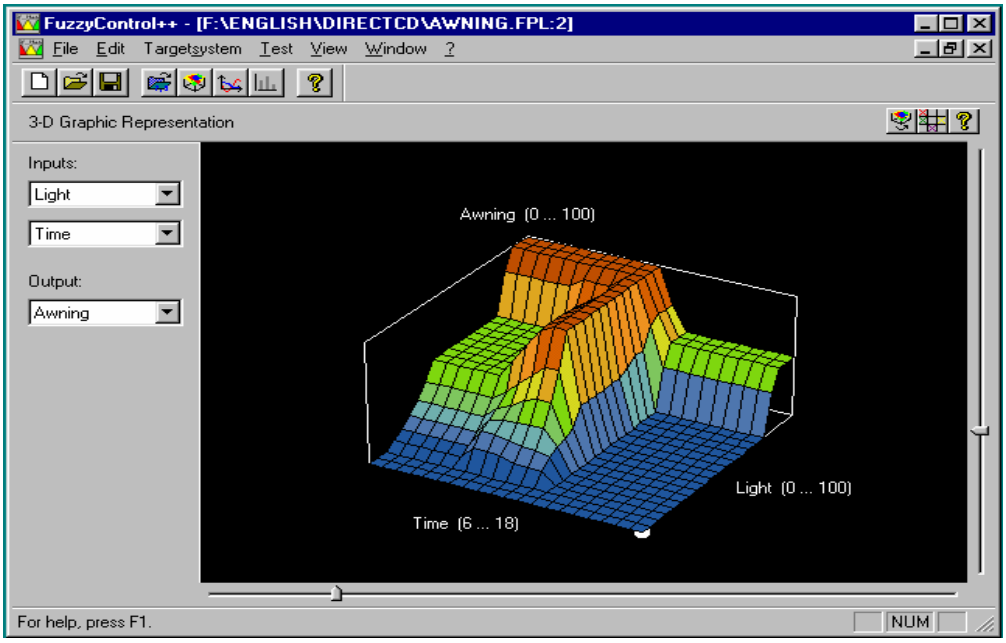
Kasutusaladeks on tehnilist laadi rakendused kõrge juhtimistehnilise osaga. Keskkonnakaitsealal on palju mittelineaarseid protsesse, mida sageli veel käsitsi juhitakse. Just need kuuluvad *Fuzzy Control* paketiga automatiseerimisele, nt.

- vingugaasi eraldamine puhastites,
- veemajandus jne.

Fuzzy Control on mõeldud ka juhtimisülesannete optimeerimiseks. Näiteks saab võimalikult võrdselt koormata seadmete eri osi; mille tulemusel mahtuvuste kasutamine paraneb, protsessi seisakud ja vead vähenevad, voolutipud alanevad.

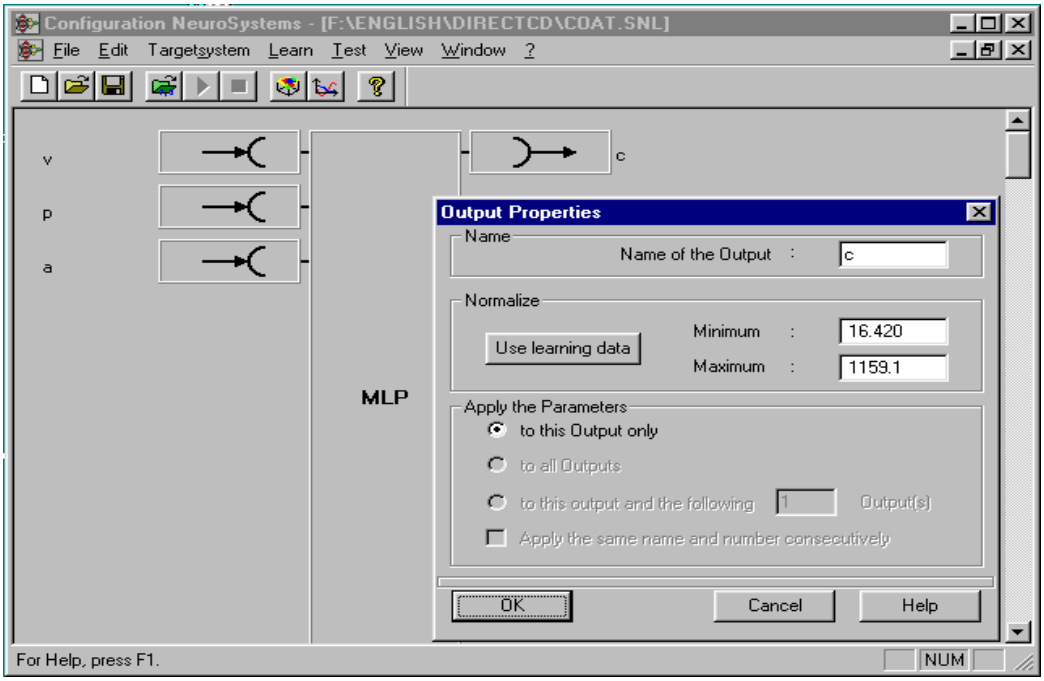
Tarkvarapakett *Fuzzy Control* teeb hägusjuhtimissüsteemi põhimõtted kasutatavaks automatiseerimissüsteemidele SIMATIC S7-300 (alates CPU 313) ja SIMATIC S7-400 ja samuti ka SIMATIC C7. *Fuzzy Control* kujutab ka toetust tavapärasele PID-regulaatorile nagu näiteks "*Standard PID Control*" või "*Modular PID Control*" eriti mittelineaarsuste korral.

Kui kasutaja on defineerinud protsessi sisend- ja väljundparameetrid, neid siduva funktsiooni ja loonud reeglite (juhiste, põhimõtete) tabeli, saab ta simuleerida süsteemi tööd kahe- ja kolmemõõtmelistel graafikutel (joonis 3.48).



Joonis 3.48. Kolmemõõtmeline esitus tarkvarapaketis Fuzzy Control++

- ***NeuroSystems*** - on Windows 95 keskkonnas töötav tarkvarapakett (joonis 3.49), mida kasutatakse neuronvõrkude arendamiseks ja konfigureerimiseks SIMATIC S7 seadmete ja protsessivisualiseerimissüsteemi WinCC juures. NeurSystems tarkvara saab kasutada süsteemide juhtimiseks ja juhtimismudelite koostamiseks, optimeerimiseks ja juhtimismustrite identifitseerimiseks. Samuti võimaldab ta optimeerida MLP (multi layer perceptrons), RBF (radial basis function networks) ja NFN (neuro-fuzzy network) andmesidevõrke.

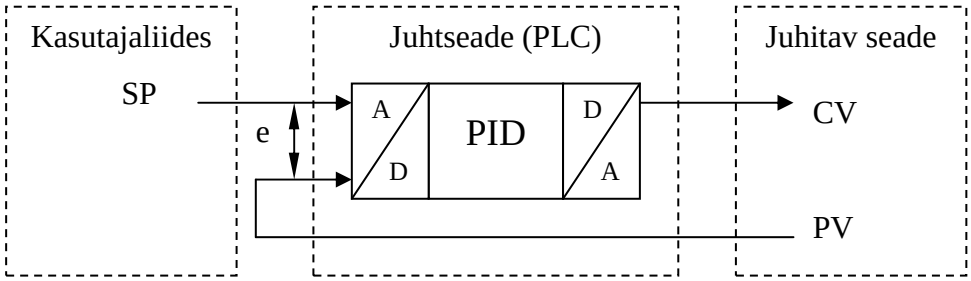


Joonis 3.49. NeuronSystems tööaken

3.3.3 PID- ja hägusjuhtimine

PID-juhtimist saab kontrolleril realiseerida süsteemsete funktsioonplokkidega; nende puudumisel võib kirjutada vastava reguleerimisprogrammi. Joonisel 3.50 on toodud PID-juhtimise plokkskeem tehnoloogilises protsessis, kus

SP on etteandesuurus (*setpoint*),
CV - juhtsuurus (*control variable*) ja
PV - protsessi suurus (*process variable*)



Joonis 3.50. PID-juhtimise põhimõtteskeem

PID-regulaatori toime koosneb proportsionaalse, integreeriva ja diferentseeriva toime summast

$$CV(t) = K_P \cdot e + K_I \int_0^t edt + CV_{initial} + K_D \cdot \frac{de}{dt},$$

kus

e - juhtimisviga ehk etteande- ja protsessisuuruse erinevus (vahe)

K_P - proportsionaalse toime võimendustegur

K_I - integreeriva toime võimendustegur

K_D - diferentseeriva toime võimendustegur

Numbrilisel kujul leitakse integraal summa funktsioonina, diferentsiaal aga vahefunktsioonina teatud ajahetkel *t*.

$$PID\_CV_N = K_P \cdot e_N + K_I \cdot \sum_1^N e_i + CV_{initial} + K_D \cdot (e_N - e_{N-1}),\ i = 1 \ldots N$$

Väljund = P-lüli + I-lüli + D-lüli

kus

$CV_{initial}$ - juhtseadme väljundi algväärtus
PID_CV_N - juhtseadme tulem diskreetsel ajahetkel *N*
e_N - juhtseadme vea väärtus diskreetsel ajahetkel *N*
e_{N-1} - juhtimisvea väärtus diskreetsel ajahetkel *N* - 1

Proportsionaalne lüli arvutab juhttoime sõltuvalt võimendustegurist ning etteande ja protsessisuuruse vahelisest veast (edaspidi lihtsalt **viga**). Integreeriv lüli arvutab juhttoime proportsionaalselt ajas summeeritud vea järgi. Diferentseeriva lüli korral arvutatakse juhttoime proportsionaalselt ajas muutuva vea järgi. P, I ja D-lülid on matemaatiliselt kirjeldatavad kontrolleri programmis järgmisel kujul:

P $P\_CV_N = K_P*(SP_N-PV_N)$

I $I\_CV_N = K_P*T_S/T_I*(SP_N-PV_N) + I\_CV_{N-1}$

D $D\_CV_N = K_P*T_D/T_S*((SP_N-PV_N)-(SP_{N-1}-PV_{N-1}))$, kui $SP_N= SP_{N-1}$, siis

$D\_CV_N = K_P*T_D/T_S*(PV_{N-1}-PV_N)$

kus

P_CV_N - proportsionaalse lüli juhttoime diskreetsel ajahetkel N
I_CV_N - integreeriva lüli juhttoime diskreetsel ajahetkel N
I_CV_{N-1} - integreeriva lüli juhttoime diskreetsel ajahetkel N-1
D_CV_N - diferentseeriva lüli juhttoime diskreetsel ajahetkel N
SP_N - etteandesuurus diskreetsel ajahetkel N
PV_N - protsessisuurus diskreetsel ajahetkel N
SP_{N-1} - etteandesuurus diskreetsel ajahetkel N-1
PV_{N-1} - protsessisuurus diskreetsel ajahetkel N-1
K_P - võimendustegur
T_S - kvantimis- ehk programmitsükli aeg
T_I - integreerimisaeg
T_D - diferentseerimisaeg

Seega kujutab PID-juhtimise kohta käiv valem endast eeltoodud osalahendite summat:

$PID\_CV_N = P\_CV_N + I\_CV_N + D\_CV_N$.

Klassikaline automaatjuhtimine rajaneb peamiselt PID-reguleerimisele. Keerukad mittelineaarsed ja liitprotsessid vajavad lisaks reguleerimisstruktuuri, milleks on ***hägusjuhtimine*** ehk FUZZY-Control. Järgnevalt võrdleme kahte tegelikkuses esinevat protsessi, mille abil selgub täpselt millal kasutada PID-juhtimist ja millal lisaks PID-juhtimisele hägusjuhtimist.

Klassikaline PID-juhtimine on kasutusel põlemisprotsessi juhtimisel temperatuuri reguleerimiseks. Eesmärgiks on reguleerida ühes anumas temperatuuri vastavalt seadesuurusele (etteandesuurusele). Selleks on vaja reguleerida gaasi ja õhu segu suhet, et toimuks optimaalne põlemine. See eeldab seega proportsionaaljuhtimise integreerimist kaskaadstruktuuri.

PID-juhtimine koos hägusjuhtimisega võimaldab aga teatud (nt. ohtlikes) situatsioonides seadesuuruse korrektuuri, mida tavaline klassikaline PID regulaator ei võimalda. Selle ilmestamiseks on toodud järgmine näide.

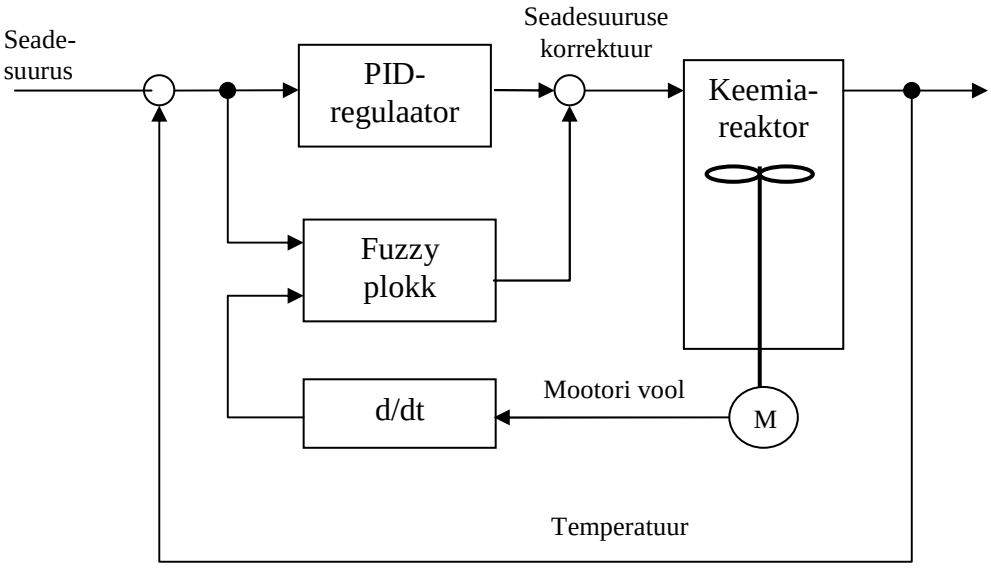
Ühes keemiareaktoris tuleb hoida temperatuuri vastavalt seadesuurusele (joonis 3.51), kusjuures tuleb arvestada sellega, et eksotermilise reaktsiooni tõttu toimub väga suure energiahulga eraldumine. Tavaline PID-regulaator sellega toime ei tule. Miks? Parameetrite, etteandesuuruse ja tegeliku suuruse väga väikese erinevuse korral muudetakse toimesuurust arvutuslikult vastavates piirides. Kui aga toimub eksotermiline reaktsioon ja temperatuuri tõus on väga järsk, siis selline protsessi kulg mõjub juhtparameetreid muutmata nagu häire regulaatorile, kuna ületatakse toimesuurus mitmekordselt. Kogenud operaator oskab sellist situatsiooni ennetada ning lülitab soojenduse peaaegu välja enne kui toimub vastava temperatuuri ületus (pisut üle 70 °C) ja rõhu järsk tõus.

KUI-SIIS-sõltuvus on hägusloogika programmeerimise alus.

Nüüd tuleb veel hägusad ehk ebatäpsed sõnad nagu “natuke üle 70 kraadi”, “tugevalt tõusev” ja “peaaegu sulgema” juurdekuuluvusfunktsioonina kirjeldada, ja juba võib PID-regulaatorit kriitilistes protsessifaasides koos hägusjuhtimisega kasutada.

Eelised nagu riskivähesus ja odavus põhjendavad hägusjuhtimise kasutamise koos PID juhtimisega, kuna

- PID-juhtimise kõiki kogemusi saab ära kasutada.
- töökindlusele saab just protsessi ajal PID-regulaatoriga eelnevalt testida,
- hägusjuhtimise järgi moodustatud manuaalseid korrektuure saab PID-regulaatori väljundisse juurde lülitada,
- igal ajal saab kontrollida, milliseid väärtusi omab protsessis PID-regulaator ja millist väärtust hägusjuhtimine.



Joonis 3.51. PID-juhtimine ja seadesuuruse korrektuur hägusjuhtimise abil

4 Protsessijuhtimisprogrammi koostamine

Protsessijuhtimisprogrammi ettevalmistamine ja koostamine, moodustavad ainult väikese osa kogu automaatikasüsteemi projektist. Alljärgnevas tabelis 4.1 [13] on Näidatud, missugused toimingud kuuluvad automaatikasüsteemide projekteerimise valdkonda ning millise osa neist moodustab programmi ettevalmistamine ja koostamine.

Tabel 4.1 Projekti valmimise etapid

<table>
<tr><td colspan="3">Ülesande kirjeldus ja jaotus</td><td rowspan="8">Dokumentatsioon</td></tr>
<tr><td colspan="3">Juhtimiseks vajaliku valik</td></tr>
<tr><td>Tarkvara valik</td><td>Riistvara valik</td><td>Elektrikilbi või -kapi valik</td></tr>
<tr><td colspan="3">Juhtimiseks vajaliku väljatöötlus</td></tr>
<tr><td>Programmeerimine</td><td>Paigutus ja ühendused</td><td>Ehituse iseärasused</td></tr>
<tr><td colspan="3">Riist- ja tarkvara kasutuselevõtt</td></tr>
<tr><td colspan="3">Juhtsüsteemi üleandmine tellijale</td></tr>
</table>

Esimeses etapis tehakse kogu süsteemi kirjeldus koos ülesande või ülesannete kirjeldusega ning jaotatakse töö teatud sõlmedeks, mis esitatakse hinnapakkumiseks elektri- ja muid komponente tootvatele firmadele. Üleande püstitamisel ei tohi unustada tehnilisi, keskkonnaalaseid ja muid tingimusi, millest sõltub edasine töö. Teises etapis valitakse antud protsessi juhtimiseks vajalik riistvara, elektrikilbid jne., mille põhjal koostatakse kolmandas etapis elektri- vm. projekt. Vastava riistvara valik tingib ka sobiva tarkvara valiku. Samaaegselt elektriprojekti valmimisega programmeeritakse juhtsüsteeme. Peale elektri- jm. projektide valmimist alustatakse kokkupaneku- ja paigaldustöödega. Kui kogu projekteeritav süsteem on valmis, alustatakse juhtseadmete häälestamisega, mida saab mõnikord teha ka paralleelselt paigaldustöödega. Juhtsüsteemi testimise ja häälestamise järel, olles igati veendunud, et see töötab vastavalt ülesandes püstitatud nõuetele, toimub projekti üleandmine tellijale.

4.1 Programmeerimiseks ettevalmistamine

Protsessijuhtimisprogrammi koostamiseks tuleb eelnevalt kindlaks määrata ja kirjeldada protsessi, mida kavatsetakse juhtida. Protsessi kirjeldamisel tuleb

- jaotada protsess väiksemateks iseseisvateks operatsioonideks ning
- kirjeldada neid operatsioone üksikasjalikult.

Protsessi kirjeldus peab lähtuma tehnoloogiaskeemist ja selle detailsest kirjeldusest. Selle põhjal koostatakse graafskeemid, ajadiagrammid jms., mis on abiks juhtimisalgoritmi koostamisel.

Teise etappi kuulub kahtlemata ka riistvara, tarkvara jm. valik (arv, tüüp jne.) (tabel 4.1). Nende valik sõltub juhitava protsessi keerukusastmest ja liigist. Näiteks riistvaraliselt sisend- ja väljundseadmete arv sõltub sellest,

- kui palju andureid ja täitureid on kasutusel
- keerukaid juhtimisülesandeid peab protsessor lahendama
- jne.

Reeglina kuuluvad kõige lihtsamatest kuni kõige raskemate protsesside juhtseadmete juurde

- toiteplokk,
- protsessor koos programmimäluga,
- sisend/väljundliides (plokk).

Keerukate protsesside lahendamisel lisanduvad neile kolmele vastavad spetsiaalliidesed, mille valikuks kasutatakse vastava firma seadmete katalooge. Tehnoloogiaskeemi kirjelduse põhjal määratakse ka kindlaks, kus on tegemist analoog- või digitaalsignaalidega, missugused ajalised sõltuvused ja seosed on andurite ja täiturite vahel jne. Riistvara valikuga paralleelselt tuleb jälgida, et juhtseadmetega kaasaskäiv tarkvara võimaldaks vastavat protsessi juhtida. Samuti tuleb arvestada keskkonnaolusid (niiskust, temperatuuri, tuleohtlikkust jne.), mis määravad riistvara, sh. elektrikilbi või -kapi valiku.

Kolmanda etapina toimub juhtimiseks vajaliku väljatöötlus, mis hõlmab nii programmeerimist, kilbi ja kapi jooniseid kui ka ehitusiseärasuste määratlemist. Programmeerimise lihtsustamiseks ja protsessist parema ülevaate saamiseks on otstarbekas jaotada ta üksikuteks operatsioonideks või funktsionaalseteks plokkideks, siduda need omavahel vastavate tingimustega ning esitada vastavad seosed graafiliselt. See tähendab algoritmi plokkskeemi, protsessi üldstruktuuri, ajadiagrammi, olekutabeli, tsüklogrammi vms. esituse koostamist. Protsessi jaotamine osadeks ja esitamine graafiliselt aitab sellest paremini aru saada ning hõlbustab hilisemat programmeerimist. Seega määratlevad loetletud kirjeldused sisuliselt programmi struktuuri, kus on näidatud operatsioonide täitmise järjekord ja nende täitmise tingimused.

Algoritmi või üldstruktuuri koostamisel tuleb lähtuda protsessi tehnoloogiaskeemist ja analüüsida, kas on võimalik seda lihtsustada. Protsessi loogikaskeemi lihtsustamiseks saab kasutada mitmesuguseid meetodeid nagu Karnaugh’ kaart, Quine-McCluskey meetod jne. Kuid eelnev ei kehti juhtautomaatide puhul, mis sisaldavad peale lihtloogika elementide ka mälufunktsioone. Seega ei saa juhtprogrammi lihtsustamiseks, kui see sisaldab ka mälu-, loendus-, viivitus- jm. elemente, kasutada eelmainitud lihtsustusmeetodeid, vaid tuleb lihtsalt jälgida, et kõik ühe ja sama väljundi juhttingimused oleksid määratletud väljundlülituse juures. Teiste sõnadega ei ole mõtet kasutada väljundi juhtskeemi eri tingimuste korral mitu korda eri programmilõikudes, kuna halveneb ülevaade programmist.

Järgneva sammuna toimub ohutustingimuste määratlemine protsessi juhtimiseks kusjuures eelnevalt peab olema täpselt teada operatsioonide omavahelised sõltuvused (elektrilised, mehaanilised, loogilised). Otstarbekas on koostada sisend/väljund diagrammid ja operatsioonide võimalikult täpsed kirjeldused. Tuleb koostada ka ahelad, mis võimaldaksid protsessi hädaolukorras käsitsi juhtida. Määratleda tuleb ka see, kus hädastopplüliti paikneb, missugused seadmed temaga välja lülitatakse; samuti peab programmis kajastuma hädastoppolukord. Programmi ettevalmistavasse etappi kuulub ka operaatorjuhtimise ja selle paneelide kirjeldus.

4.2 Programmeerimine

Kui skeem on koostatud ja lihtsustatud, seadmed valitud ning juhtimiseks vajalik välja töötatud, tuleb asuda kontrolleri programmeerimise juurde. Et kontrollerit programmeerida, on vaja vastavat tarkvara. Tarkvara valik on määratud vastava riistvara kasutusega. Kasutades protsessi juhtimiseks mingi kindla firma riistvara, tuleb kasutada ka sama firma tarkvara. Vastasel juhul ei ole võimalik riistvara programmeerida ja protsessi juhtida, kuna iga konkreetse firma seadmetel on eripärad, mis ei pruugi sarnaneda mõne teise firma eripäradega. Kui seadmete ja tarkvara valik on tehtud, tuleb valida kasutajale kõige sobivam programmi esitusviis. Enamkasutatavad programmeerimise esitusviisid on kindlaks määratud standardiga IEC 61131-3 (vt. § 2.1). Programmeerija peab seega valima pakutud esitusviisidest kõige sobivama ja alustama programmeerimist.
Programmi saab kirjutada nii arvuti kõvakettalele (ketasmällu) kui ka otse kontrolleri muutmällu. Soovitatav on programm eelnevalt arvuti kettaseadmetele kirjutada, mille järel tuleks programmis kontrollida täiturite sisse- ja väljalülitamistingimusi. Selleks tuleks küsida endalt täiturseadme kohta järgnevalt, millal

- tohib sisse lülitada,
- ei tohi sisse lülitada,
- peab sisse lülitama,
- peab jääma sisselülitatuks,
- tohib välja lülitada,
- ei tohi välja lülitada,
- peab välja lülitama.

Kui on veendutud programmi õigsuses, s. t. selles, et ta täidab programmeerija ja protsessi poolt püstitatud tingimusi, võib minna järgmise etapi juurde, milleks on programmi *testimine*.

4.3 Programmi testimine ja protsessi diagnostika

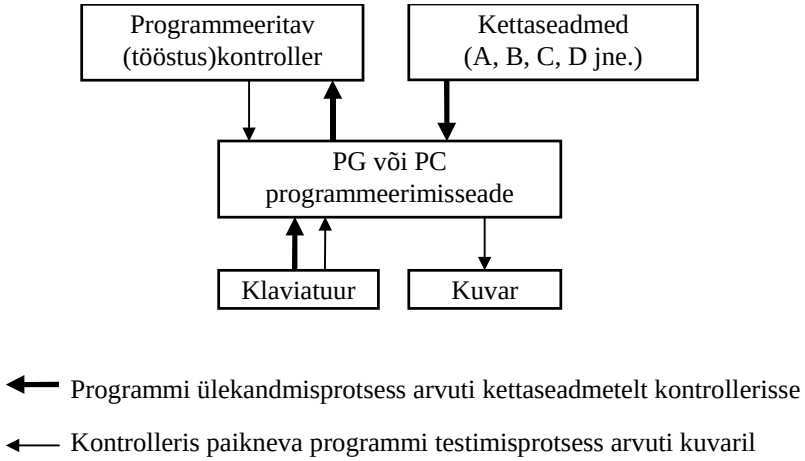
Riist- ja tarkvara kasutuselevõtt (tabel 3.1) koosneb paljudest toimingutest, nagu nt. protsessi töö kontroll, diagnostika ja programmi testimine. Riist- ja tarkvara kasutuselevõtule järgneb töö üleandmine tellijale, mida siin ei käsitleta.

Protsessi kontrolltoimingutega tehakse kindlaks, kas elektriseadme, seadmekomplekti või paigaldise ohutus, talitlusvõime ja füüsikalised omadused (parameetrid, tunnusjooned jms.) on nõuetekohased. Seega võib protsessi kontrollimine (katsetamine, testimine) sisaldada *proovimist*, *mõõtmist* või *teimi*. Proovimine seisneb seadme või selle osa talitlemise kontrollis mõõtmisi kasutamata. Mõõtmine on mõõteriista(de) abil sooritatav kontrollitoiming. Teim seisneb teatava, enamasti seadme mingi tunnussuuruse teimiväärtuse rakendamises seadmele teatavaks ajaks või muul teataval viisil; tarbe korral võidakse teimi ajal mõõta seadme korrasolekut iseloomustavaid suurusi.

Protsessi töö kontroll hõlmab endas samuti koostatud programmi testimist ja diagnostikat reaalses tööprotsessis. Vastav tarkvara on iga konkreetse seadme jaoks tavaliselt olemas. Ka selle kohta saab täpsemat infot vastava firma seadmete kataloogist.

Protsessi töö kontrolli all mõistetakse programmeeritavate kontrollerite juures programmi ja protsessi õigsuse kontrolli ehk testimist s. t. kindlakstegemist, kas programmi poolt juhitav protsess vastab protsessi kirjelduses püstitatud tingimustele ja eelnevalt koostatud algoritmile. Et selles veenduda, tuleb arvutil koostatud programm laadida kontrollerisse ja käivitada vastav testprogramm või kasutada testpaketti mis võimaldab testida (S7-PLCSIM) programmi kontrolleri olemasoluta. Tarkvarapakett S7-PLCSIM simuleerib kontrolleri puudumisel kontrolleris toimuvat arvuti ekraanil. Programmi seda laadi eelnev testimine tõstab programmi kvaliteeti ja vähendab projekti esmakäikuandmise kulusid, sest seadmete protsessi jaoks häälestamise aeg väheneb tunduvalt.

Et testida valminud programmi kontrolleril, tuleb kõigepealt kopeerida ehk siirata ta programmaatori mälust, milleks võib olla personaalarvuti kõvaketas, juhtseadme ehk kontrolleri mällu ning viia seejärel kontroller talitlusolukorda. Seejärel tuleb programmpaketis vastavast menüüst startida testimine, mille abil saab testida programmi tööd kontrolleris (joonis 4.1).



Joonis 4.1. Programmi ülekandmine ja testimine

Kuna programm võib koosneda paljudest plokkidest (juht-, andme-, funktsioonplokid ja funktsioonid), siis tuleb jälgida seda, et testimise lõppfaasis, milles kontrollitakse kogu programmi tööd, oleks kõik plokid kontrolleri mällu laetud.

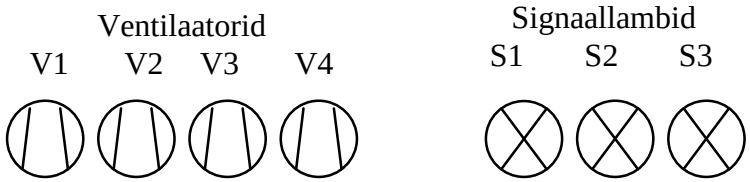
Vigade esinemisel programmis või kontrolleri moodulites on diagnostika võimalik kui programmeerimistarkvara seda võimaldab. Sellise diagnostika kasutamise korral on küllaltki lihtne leida kontrolleris protsessori töö katkemist põhjustanud viga. Lähemalt vaadeldakse diagnostikat punktis 6.3.4.

5 Juhtimisprogrammide näiteid

5.1 Ventilatsiooni signalisatsioon

Protsessijuhtimise algoritmi loomist alustatakse tehnoloogilise skeemi koostamisega (joonis 5.1) ja selle kirjeldamisega. Vaadelgem alljärgnevat näidet [13].

Ettevõtte sööklat ventileeritakse nelja ventilaatoriga. Sõltuvalt õhu temperatuurist, niiskusest või töös olevate pliitide arvust reguleeritakse ventilatsiooniõhu hulka. Elektriseadmete hooldusspetsialistil on vaja tagada söökla ventilaatorite korrashoid. Seega huvitab teda töös olevate ventilaatorite arv, milleks kasutatakse signaallampe.



Joonis 5.1. Tehnoloogiaskeem

Pärast tehnoloogiaskeemi koostamist tuleb juhitav protsess operatsioonide kaupa üksikasjalikult kirja panna, et hiljem kasutada seda programmeerimisel ja programmi õigsuse kontrollimisel. Käesolevas näites on protsessi kirjeldus järgmine:

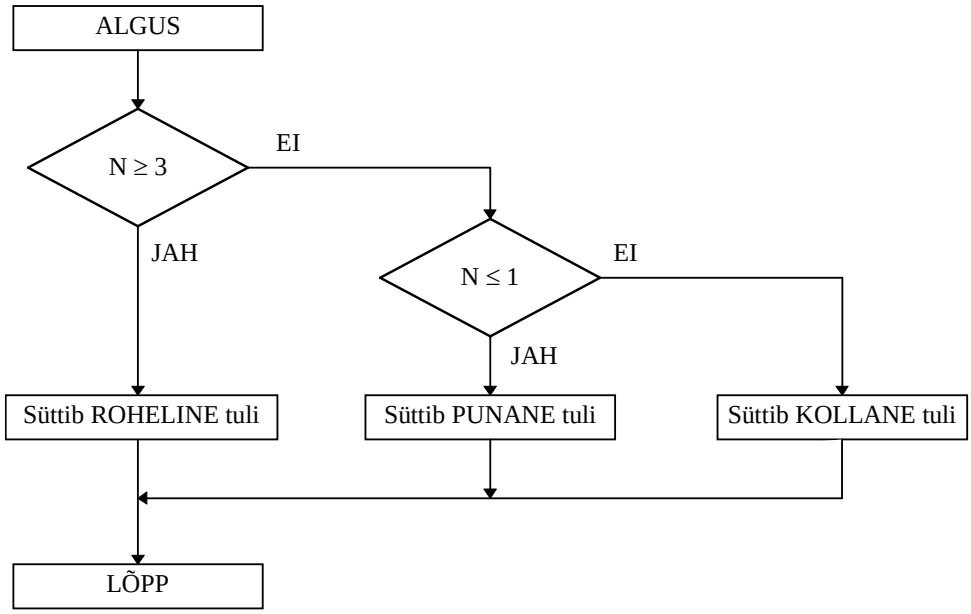
1. Signaallamp 1 (punane) peab olema sisse lülitatud, kui ei tööta ükski ventilaator või töötab ainult üks ventilaator neljast.
2. Signaallamp 2 (kollane) peab olema sisse lülitatud, kui töötab ainult kaks ventilaatorit neljast.
3. Signaallamp 3 (roheline) peab olema sisse lülitatud, kui töötavad kõik ventilaatorid või kolm ventilaatorit neljast.

Pärast tehnoloogiaskeemi ja protsessikirjelduse koostamist tuleb määratleda ja kirjeldada juhtkontrolleri sisend- ja väljundoperande (tabel 5.1)

Tabel 5.1 Juhtkontrolleri sisend- ja väljundoperandid

Operand	Tähis	Kirjeldus
E 0.0	V 1	Ventilaator 1
E 0.1	V 2	Ventilaator 2
E 0.2	V 3	Ventilaator 3
E 0.3	V 4	Ventilaator 4
A 4.0	S 1	Signaallamp 1 (punane)
A 4.1	S 2	Signaallamp 2 (kollane)
A 4.2	S 3	Signaallamp 3 (roheline)

Järgmise sammuna on vaja protsessi tehnoloogiaskeemi ja kirjelduse järgi luua programmeerimist hõlbustav struktuuriskeem või algoritmi plokkskeem. Ülesande esitamisel algoritmina tuleb täpselt arvestada, mis tingimustel operatsioonid toimuvad. Antud näite kohta on joonisel 5.2 toodud algoritmi plokkskeem, kus *N* tähistab mingil hetkel töötavate ventilaatorite arvu.



Joonis 5.2. Signaallampide juhtimise algoritmi plokkskeem

Protsessi juhtalgoritmi plokkskeemi põhjal tuleb koostada loogikaskeem või valem, mida võimaluse korral lihtsustatakse. Antud ülesande lahendamisel on tegemist kombinatsioonloogikaga ning loogikavalemid näevad välja järgmised:

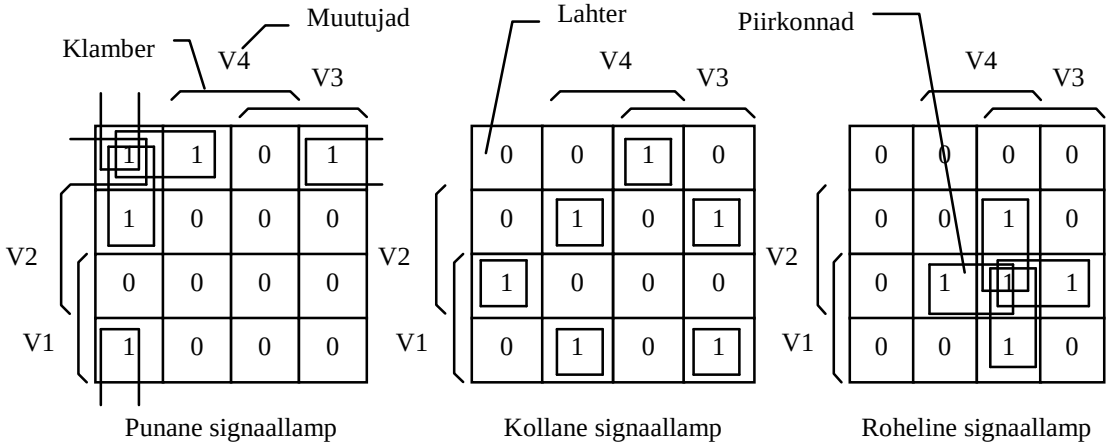
$$S1 = \overline{V1}\cdot\overline{V2}\cdot\overline{V3} + \overline{V1}\cdot\overline{V2}\cdot\overline{V4} + \overline{V1}\cdot\overline{V3}\cdot\overline{V4} + \overline{V2}\cdot\overline{V3}\cdot\overline{V4} + \overline{V1}\cdot\overline{V2}\cdot\overline{V3}\cdot\overline{V4},$$

$$S3 = V1\cdot V2\cdot V3 + V1\cdot V2\cdot V4 + V1\cdot V3\cdot V4 + V2\cdot V3\cdot V4 + V1\cdot V2\cdot V3\cdot V4,$$

$$S2 = \overline{S1}\cdot\overline{S3},$$

kus S1 tähistab punast, S3 rohelist ja S2 kollast signaallampi ning V1…V4 ventilaatoreid.

Kombinatsioonloogikalülituse lihtsustamiseks võib kasutada näiteks Karnaugh' kaarti (joonis 5.3) [7]. Karnaugh' kaardi lahtrite (ruutude) arv sõltub funktsiooni sisendmuutujate arvust *n* ning vastab muutujate kombinatsioonide arvule 2^n. Sisendmuutujate arvuks siin on ventilaatorite arv. Muutujad ja funktsiooni väärtused paigutatakse tabelisse nii, et saaks esitada kõiki muutujate kombinatsioone. See eeldab muutujate erilist paigutust. Klambriga hõivatud alas on muutujal otsene, väljaspool klambrit aga inverteeritud väärtus. Karnaugh' kaardi iseloomulikuks omaduseks on see, et funktsiooni väärtused erinevad kõrvuti asuvates lahtrites vaid ühe muutuja poolest, s. t. naaberlahtrisse minekul muudab oma väärtust vaid üks sisendmuutuja. Naabriteks loetakse ka kaardi äärmised vasak- ja parempoolsed lahtrid omavahel ning ülemised ja alumised lahtrid omavahel. Naaberlahtreid, mis erinevad vaid ühe muutuja poolest, kasutatakse loogikafunktsiooni minimeerimiseks. Lahtritesse, mis vastavad funktsiooni tingimusele, kirjutatakse olek "1". Naaberlahtritest mis omavad sarnast suurust, moodustatakse piirkonnad suurusega 2^n, kus n = 0, 1, 2, 3, …, m. Neid piirkondi peab olema nii vähe kui võimalik ja nad peavad olema nii suured kui võimalik. Iga sellise piirkonna jaoks saab kirjutada loogilise korrutamise ja enama kui ühe piirkonna puhul seotakse need piirkonnad loogilise liitmisega. Seega loogikafunktsiooni avaldis kirjutataks *disjunktiivsel normaalkujul*, kus igale piirkonnale vastab elementaarkonjunktsioon muutujatest, mis terve kontuuri jaoks on inverteerimata või inverteeritud.



Joonis 5.3. Karnaugh' kaardiga lihtsustamine

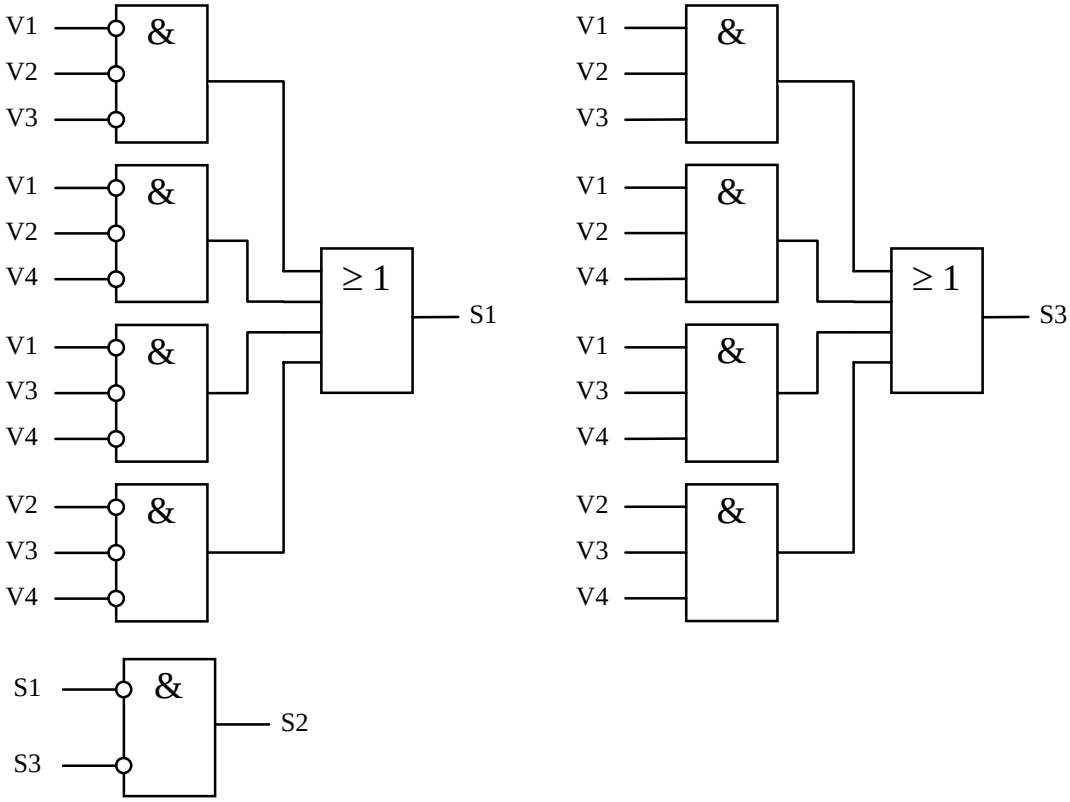
Karnaugh' kaartidelt on näha, et lihtsustada saab punase ja rohelise signaallambi juhtimislülitust. Kõigepealt tuleb kirjutada ruutudesse, mis vastavad funktsiooni tingimusele olek "1". Kuna punase tule süttimise tingimus oli, et töötab üks või ei tööta ühtegi ventilaatorit, tuleb kirjutada olek "1" neisse ruutudesse, mis ei lange kas ühegi klambri piirkonda või langeb ainult ühe klambri piirkonda. Rohelise tule puhul oli tingimus, et lamp põleb, kui töötab 3 ventilaatorit või kõik neli. Seega ruudud, mis langevad korraga kas kolme või nelja klambri (muutuja) piirkonda peavad sisaldama olekut "1". Piirkondade moodustamisel ja loogikafunktsiooni kirjutamisel tuleb jälgida, et piirkonna jaoks väljakirjutatud loogilises korrutises peavad olema nende klambrite ehk sisendmuutujate tähised, mille piirkonda või mille piirkonnast välja nad täielikult langevad. Seega on ülesande lihtsustamisel saadud loogikafunktsioonid järgmised:

$$S1 = \overline{V1} \cdot \overline{V2} \cdot \overline{V3} + \overline{V1} \cdot \overline{V2} \cdot \overline{V4} + \overline{V1} \cdot \overline{V3} \cdot \overline{V4} + \overline{V2} \cdot \overline{V3} \cdot \overline{V4} = \overline{V1} \cdot \overline{V2} \cdot (\overline{V3} + \overline{V4}) + \overline{V3} \cdot \overline{V4} \cdot (\overline{V1} + \overline{V2})$$

$$S3 = V1 \cdot V2 \cdot V3 + V1 \cdot V2 \cdot V4 + V1 \cdot V3 \cdot V4 + V2 \cdot V3 \cdot V4 = V1 \cdot V2 \cdot (V3 + V4) + V3 \cdot V4 \cdot (V1 + V2)$$

$$S2 = \overline{S1} \cdot \overline{S3}$$

Saadud loogikavalemite põhjal võime koostada loogikaskeemi (joonis 5.4), mis saab aluseks hiljem loodavale juhtimisprogrammile. Loogilise liitmise puhul, nagu teada, kasutatakse loogikaelementi VÕI ja loogilise korrutamise korral elementi NING.



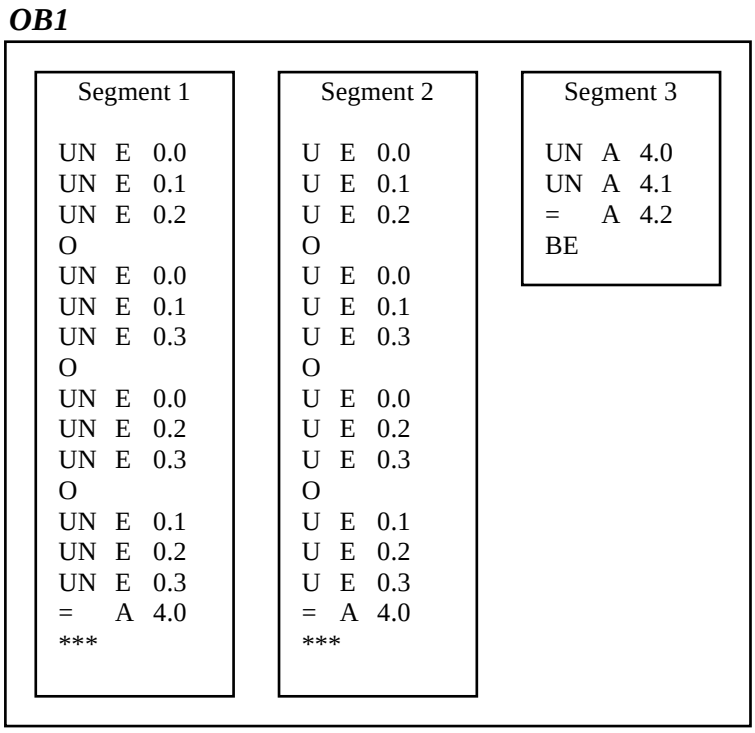
Joon 5.4. Juhtimisskeemi koostamine

Programmeerimise ettevalmistusse kuulub ka seadmete valik (p. 6.3.1). Kuna antud juhul on tegemist ainult nelja sisendsuuruse ja kolme väljundsuurusega, siis on juhtimiseks vaja kontrollerit, mille sisend/väljundliidesel on vähemalt 4 sisendit ja 3 väljundit (enamasti sisaldab juba protsessoriplokk sisend/väljundliidest). Kuna tegemist on väga lihtsa ja väikesemahulise protsessi ja programmiga, võib vabalt kasutada väikese või keskmise võimsusega seadet ning valida selleks SIMATIC S7-300 või 400. Valime S7-300, mis on ettenähtud väikese ja keskmise võimsusega ülesannete lahendamiseks.

Kontrolleri S7-300 puhul kasutatakse programmeerimispaketti *SIMATIC Manager*. Programmi esitusviisina valime käsulisti. Kuna programm on väga lühike, pole vaja seda parema ülevaate saamiseks alamplokkideks jaotada, vaid kirjutada tervikuna juhtplokki OB1. Kuna kõik plokid koosnevad segmentidest ja antud juhul koosneb

programm kolmest signaallambi juhtimisskeemist, siis võiks selle jaotada kolmeks segmendiks (joonis 5.5).

Signaallampide puhul pole vaja kasutada lisaahelatena hädastopp ja käivituslülitit, sest nad peavad ventilaatorite töö kohta pidevalt infot edastama. Seega pole tarvidust kasutada programmis vastavaid ahelaid.



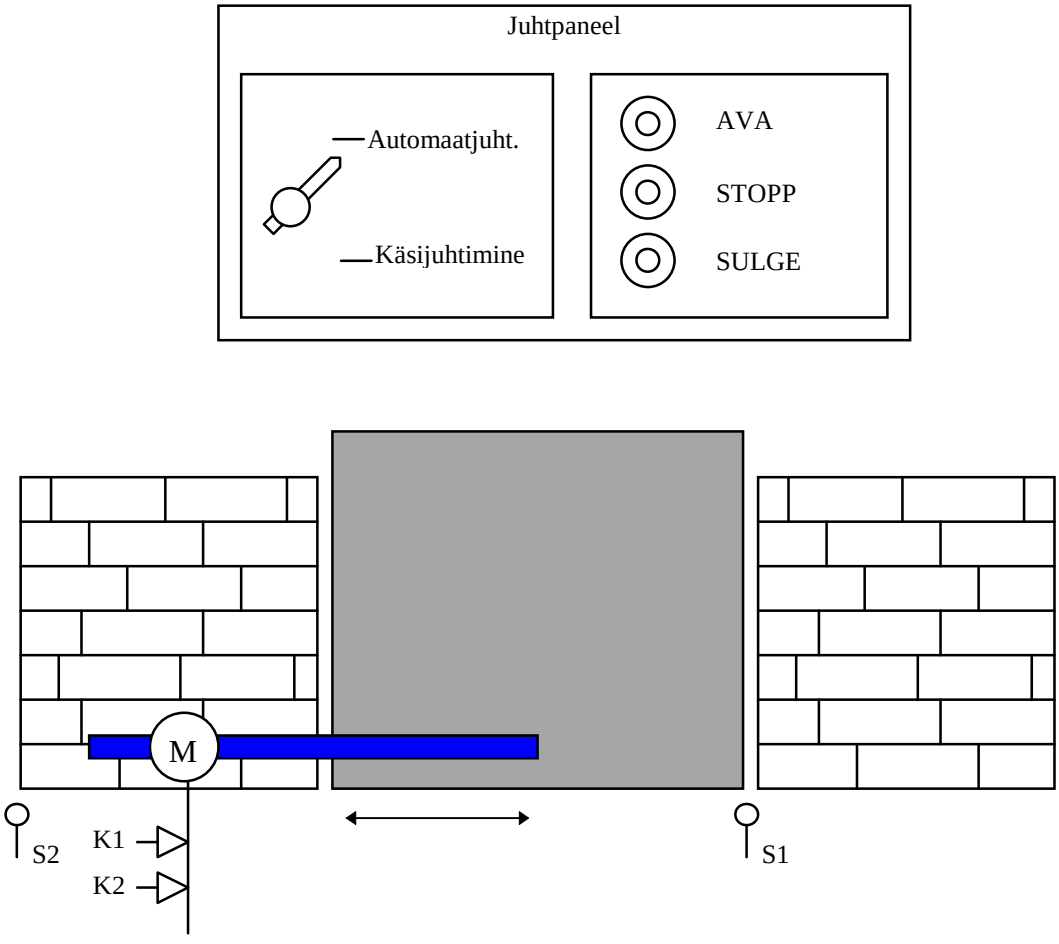
Joonis 5.5. Ventilatsiooni signalisatsioon käsulistina

Antud programmi testimiseks laetakse esmalt juhtplokk OB1 kontrolleri mällu ning seejärel käivitatakse testprogramm.

5.2 Tehase värava juhtimine

Mööblitehases toimub pidev tooraine sissevedu ja valmistoodangu väljavedu. Et tehase territooriumile sissesõit ning sealt väljasõit saaks toimuda, peab väravavaht jälgima, mida ja kuhu veetakse ning vastavalt vajadusele kas avama või sulgema värava. Seda tehakse siiani käsitsi sõltumata sellest, milline ilm parajasti õues on. Et lihtsustada värava avamist ja sulgemist, peeti otstarbekaks see automatiseerida nii, et väravavaht saaks väravat eemalt juhtida, olles ise soojas ruumis, kust monitoridelt saab kaamerate abil värava juures toimuvat jälgida.

Ettevalmistus programmeerimiseks algab analoogiliselt eelmise näitega. Kõigepealt koostatakse juhitava protsessi tehnoloogiaskeem. Tehnoloogiaskeemi koostamisel tuleb kindlasti arvestada liikuvate osade piir- ja vaheasendeid, milles toimub peatumine. See tähendab, et vastavatesse kohtadesse tuleb paigutada andurid. Tehnoloogiaskeemil peavad olema näidatud kõik andurid ja täiturid (joonis 5.6).



Joonis 5.6. Tehnoloogiaskeem

Tehnoloogiaskeemi põhjal tuleb jälle koostada protsessi kirjeldus. Tehase väravat peab olema võimalik elektrimootoriga avada ja sulgeda. Elektrimootorit juhitakse kahe kontaktori K1 ja K2 abil. Kui rakendub kontaktor K1, pöörleb mootor paremale ja värav avaneb. Kontaktori K2 rakendumisel pöörleb mootor vasakule ja värav

sulgub. Mõlemad kontaktorid ei tohi korraga rakenduda. Värava piirasendi olekute kohta annavad infot piirlülitite S1 ja S2 avanevad kontaktid.

Juhtpuldil saab valida käsi- või automaatjuhtimise. Kui on valitud automaatjuhtimine, piisab lühiajalisest vajutamisest lülitile AVA (sulguv kontakt) või SULGE (sulguv kontakt) ning uks sulgub või avaneb, kui seda toimingut ei katkesta piirlüliti (avanevad kontaktid) või lüliti STOPP (avanev kontakt) rakendumine (vajutus). Käsijuhtimisel toimub avanemine või sulgemine nii kaua kui vajutatakse vastavat juhtnuppu või kui uks saavutab piirasendi. Programmeerimisel tuleb jällegi määrata protsessis kasutatavad kontrolleri sisend- ja väljundoperandid koos sümbolvastetega (tabel 5.2).

Tabel 5.2 Sümbolite tabel

Operand	Tähis	Kirjeldus
E 0.0	Automaat-käsijuhtimine	talitlusviisi valik
E 0.1	AVA	Käsklus-ava värav
E 0.2	SULGE	Käsklus-sulge värav
E 0.3	STOPP	Käsklus-värav stopp
E 1.0	S1	Piirlüliti-värav on kinni
E 1.1	S2	Piirlüliti-värav on lahti
A 4.0	K1	Värava avanemise kontaktor
A 4.1	K2	Värava sulgemise kontaktor

Protsessi täpsemaks selgitamiseks on järgnevalt soovitav koostada tsüklogramm või olekutabel. Antud juhul on valitud mõlema väljundi jaoks eraldi olekutabel (tabelid 5.3 ja 5.4). Vasakpoolsetes veergudes on toodud antud väljundi kirjeldamise jaoks olulised sisendite olekud, kusjuures sisendina võib kasutada ka mõnda teist väljundit. Tabelites kasutatav sümbol “**x**” tähendab seda, et antud sisendparameetri olek võib olla kas “0” või “1”

Tabel 5.3 Värava avanemise juhtimine

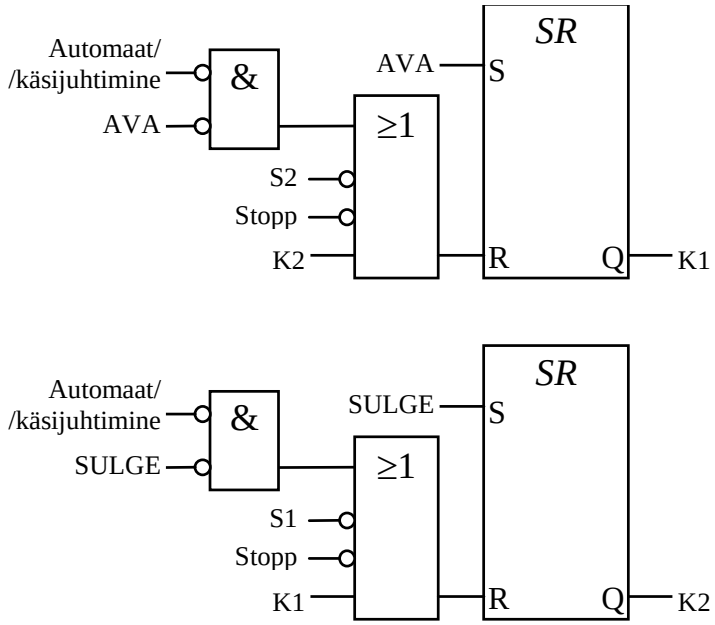
Sisendid					Väljund
Automaat/käsijuhtimine	AVA	STOPP	S2	K2	K1
x	1	1	1	0	1
0	0	x	x	x	0
x	x	0	x	x	0
x	x	x	0	x	0
x	x	x	x	1	0

Tabel 5.4 Värava sulgemise juhtimine

Sisendid					Väljund
Automaat/käsijuhtimine	SULGE	STOPP	S1	K1	K2
x	1	1	1	0	1
0	0	x	x	x	0
x	x	0	x	x	0
x	x	x	0	x	0
x	x	x	x	1	0

Värava sulgemise juhtimise olekutabeli 5.4 esimest rida saab kirjeldada järgmiselt. Sõltumata lüliti *Automaat/käsijuhtimine* olekust, kui lüliti *SULGE* ning lüliti *STOPP* ning lüliti *S1* on olekus “1” ning kontaktor *K1* on olekus “0”, siis kontaktor *K2* läheb olekusse “1” ja jääb sellesse olekusse kuni ei toimi tabeli mõni teine tingimus. Näiteks tabeli teine rida määrab, et sõltumatult lülitite *STOPP* ja *S1* ning kontaktori *K1* olekust, kui lüliti *Automaat/käsijuhtimine* on olekus 0 ja lüliti *SULGE* on olekus “0”, siis lülitatakse kontaktor K2 olekusse “0” ja ta jääb sellesse olekusse kuni ei toimi tabeli mõni teine tingimus.

Eelnevalt koostatud olekutabeli ja protsessi kirjelduse põhjal võib joonistada loogilise juhtskeemi (joonis 5.7), mis toimib vastavalt protsessi kirjeldusele.



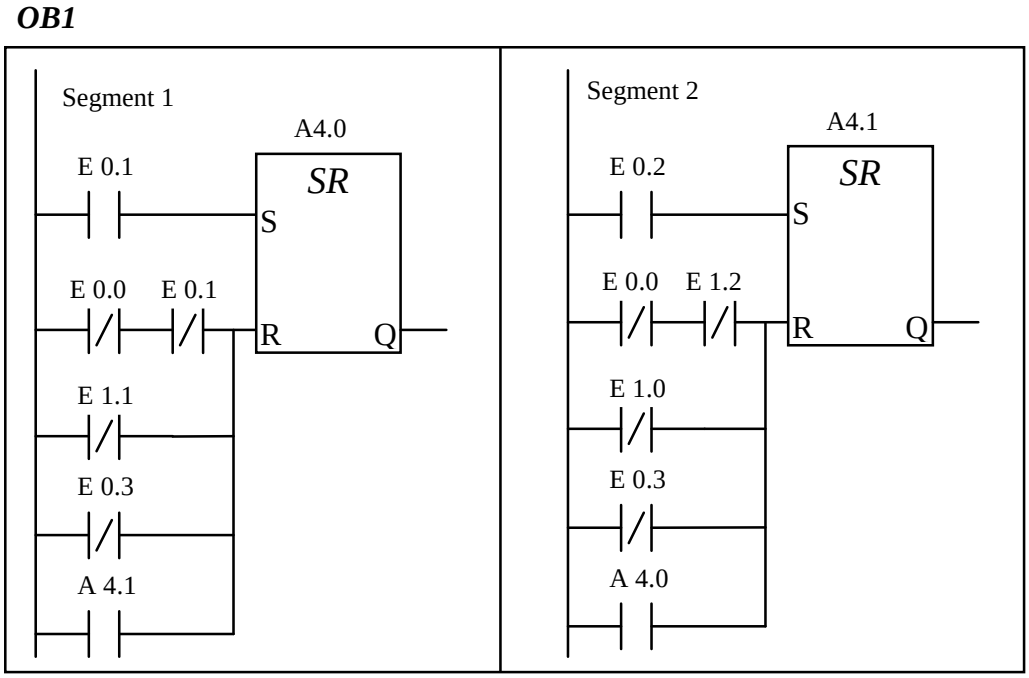
Joonis 5.7. Juhtimisskeem

Kuna antud protsessis on tegemist ainult viie sisendsuuruse ja kahe väljundsuurusega, on antud protsessi juhtimiseks meil vaja toiteplokki, protsessoriplokki ja vähemalt 5 sisendi ja 2 väljundiga sisend/väljundliidest.

Samuti nagu eelmise näite puhul on tegemist väga lihtsa ja väikesemahulise programmiga, seepärast võib juhtimiseks kasutada väikese või keskmise võimsusega seadet, näiteks SIMATIC S7-300. Kontrolleri S7-300 programmeerimiseks tuleb jälle kasutada programmeerimispaketti *SIMATIC Manager*. Järgmisena valitakse programmi esitusviis, milleks olgu nt. kontaktaseskeem. Kuna programm on väga lühike, pole seda vaja jaotada alamplokkidesse ning võime kirjutada ta tervikuna juhtplokki OB1 (joonis 5.8). Nagu teada, koosnevad kõik plokid segmentidest ning antud juhtploki segmendis 1 on värava avamise programmi lõik ja segmendis 2 värava sulgemise programmi lõik.

Tehase värava juhtimise korral on tegemist lihtsa protsessiga, mida kontrollib operaator, ning STOPP-lüliti (E 0.3) asendab hädastopplülitust. STOPP-lüliti

katkestab koheselt värava liikumise. Siiski ei tohi unustada, et hädastopplüliti peab eksisteerima mitte ainult programmiliselt, vaid ka riistvaraliselt.

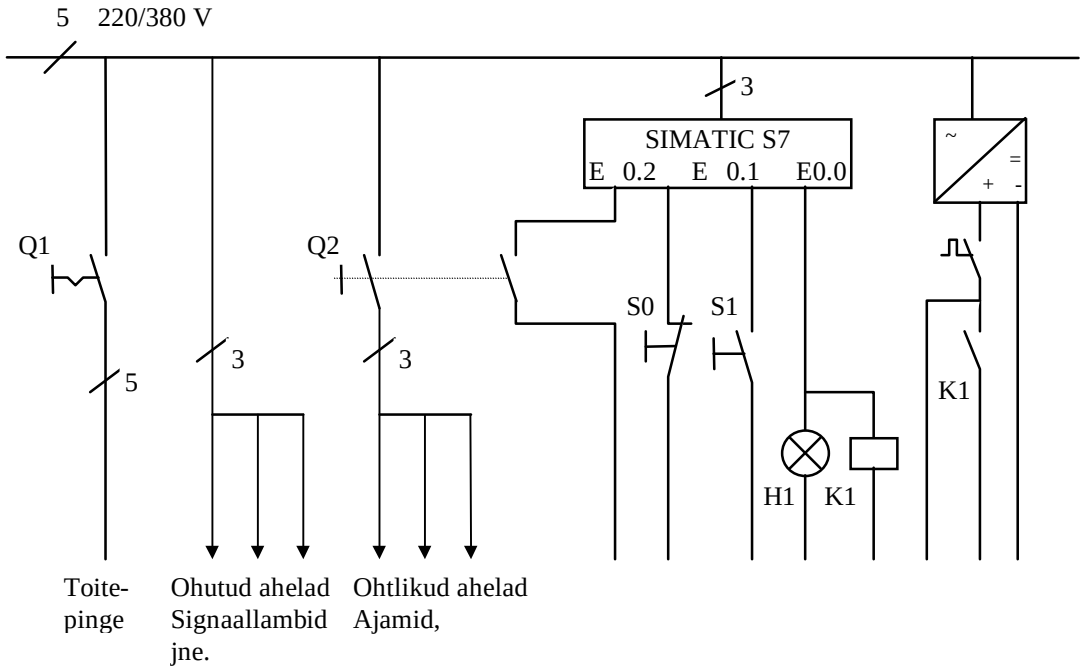


Joonis 5.8. Tehase väravate juhtimine kontaktskeemina

Antud programmi testimiseks laetakse esmalt juhtplokk OB1 kontrolleri mällu ning seejärel käivitatakse testprogramm.

5.3 Hädastopp- ja käivituslüliti

Hädastopplülitusseade peab ohuolukorras peatama ajami, tööpingi või tööpingiosa nii, et ei tekkiks ohtu inimestele ega seadmetele. Ohuolukorras tuleb peatada vaid need seadmed, mis on juba põhjustanud ohtlikku olukorra või mis võivad seda lähimal ajal põhjustada.



Joonis 5.9. Hädastopplülitusseade

Joonis 5.9 iseloomustab ühe juhtsüsteemi toiteahelat. Lüliti Q1 (pealüliti) abil lülitatakse sisse kogu juhtimissüsteemi toitepinge.

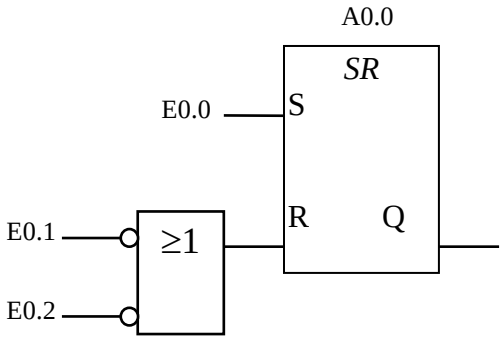
Hädastopplülitiga Q2 **<u>lülitatakse välja</u> kõik ajamid ja seadmed, mis on põhjustanud või võivad põhjustada ohtliku olukorra inimestele või teistele seadmetele**.

Hädastopplülitiga Q2 **<u>ei lülitata välja</u> seadmeid, mille väljalülitamine võib põhjustada ohtliku olukorra inimestele või teistele seadmetele**.

Hädastopplüliti oleku kohta annab juhtsüsteemile infot abikontakt, mis on ühendatud kontrolleri sisendisse E 0.2 . Seega tuleb tehnoloogilises programmis alati arvestada hädastopplüliti olekut. Hädastopplüliti Q2 võib paigutada ka toiteahelasse, kus paikneb pealüliti Q1, kuid sel juhul peab olema kindel, et ükski ajami ega seadme osa ei vaja toidet hädaolukorras. Tuleb arvestada ka seda, et siis ei saa juhtprogrammis hädastopplüliti olekut arvestada, sest ka juhtsüsteem lülitatakse välja.

Käivituslüliti

Pärast toitesüsteemi sisselülitamist, juhtseadme tsüklilisse talitlusse lülitamist ja pingekatkestust ei tohi kontrolleri väljunditega ühendatud täiturseadmed oma talitlust jätkata (väljundisse ei tohi juhtsignaali anda) enne kui selleks on antud luba, kuna selle läbi võivad saada kannatada inimesed, seadmed ning tootmine. Üks võimalus sellist juhtimist programmiliselt realiseerida on toodud joonisel 5.10.

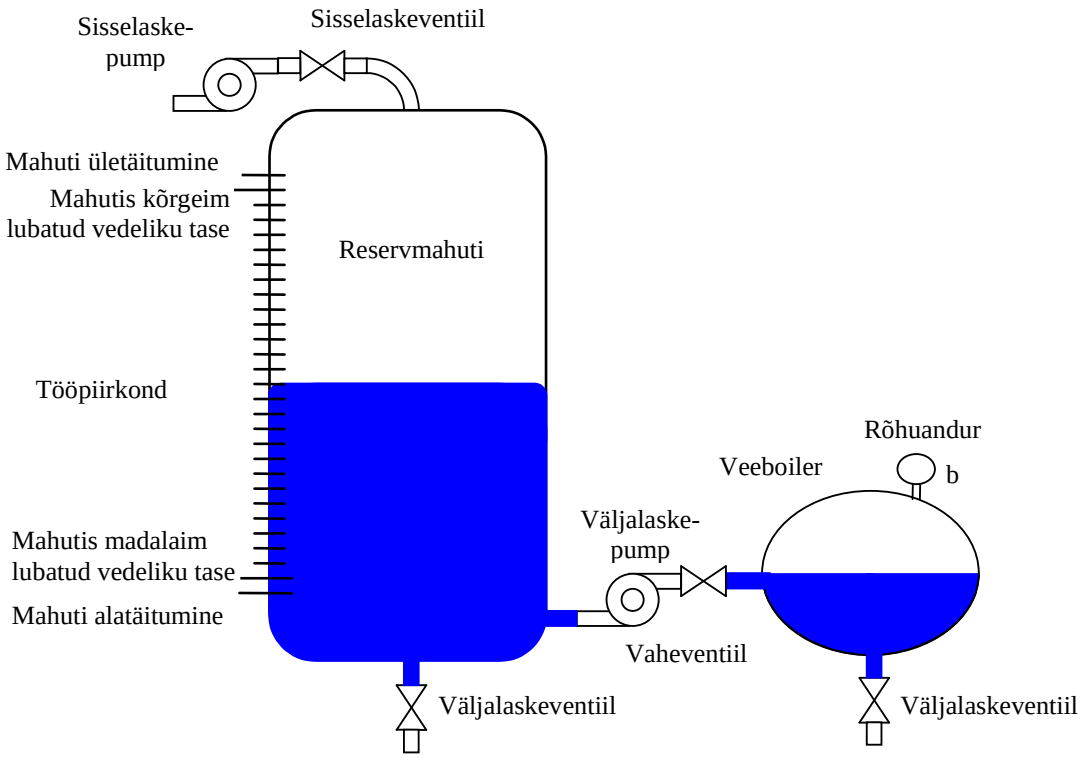


Joonis 5.10. Käivitusahela loogikaskeem

Väljund A 0.0 lülitatakse lülitiga S1 (E 0.0) sisse. Väljund A0.0 lülitakse välja, kui on vajutatud lülitile S0 (E 0.1) või hädastopplülitile Q2 (E 0.2). Samuti toimub väljundi A0.0 väljalülitamine andurivea või pingekatkestuse korral vahemälu kustutamise tõttu. Väljundsignaali A 0.0 väärtust saab mujal programmis kasutada sisse- ja väljalülitustingimusena. Kaitsekontakti K1 abil, mis on ühendatud väljundisse A 0.0, saab valikuliselt sisse ja välja lülitada ainult teatud täiturite vooluahelad (joonis 5.9).

5.4 Pidevjuhtimine

Pidevjuhtimise tüüpnäide on vedeliku nivoo mõõtmine, mis sisaldab nivoo piirväärtuste pidevat jälgimist. Kõigepealt tuleb protsessist mõõta mahuti vedeliku nivoo ja anda see voolu- või pingesuurusena juhtseadmesse. Juhtseadme sisemise töötlemise jaoks muudetakse väljast tulev analoogsignaal analoogsisendplokis digitaalsuuruseks. Digitaalsuurus loetakse STEP7-programmi abil juhtseadme keskplokki, kus programmiliselt antakse ette piirväärtused, mille vahel vedeliku nivoo võib kõikuda. Analoogväljundeid kasutatakse reguleerimistööde jaoks, näiteks kui tuleb reguleerida voolu ventiiliga mingil mootoril (tööseadmel). Analoogväljundplokk muundab sel juhul sisemised digitaalsed reguleerimissuurused mootorile (tööseadmele) vajalikeks analoogsignaalideks.



Joonis 5.11. Vedeliku nivoo reguleerimine

Joonisel 5.11 ongi esitatud ülesanne, milles tuleb kasutada analoogsisendmoodulit reservmahuti vedeliku nivoo mõõtmiseks. Ülesandeks on hoida vedeliku tase reservmahutis alati lubatud piirides. Selleks tuleb lülitada vastavaid pumpasid ja ventiile õigel ajal sisse või välja. Näiteks kui reservmahutis on vedelik saavutanud kasutaja poolt vähima lubatud või ettemääratud nivoo, tuleb koheselt lülitada väljalaskepump välja ja sulgeda pumba juures paiknev vaheventiil ning seejärel, juhul kui sisselaskepump ei tööta, lülitada see sisse ning eelnevalt avada sisselaskeventiil. Vastupidisel juhul, kui vedelik reservmahutis on saavutanud kasutaja poolt suurima lubatud või ettemääratud nivoo, tuleb koheselt lülitada välja sisselaskepumba mootor ja sulgeda sisselaskeventiil ning seejärel lülitada sisse väljalaskepump ja avada pumba juures paiknev vaheventiil. Järgneva sammuna alustatakse pidevprotsessi juhtimise

programmeerimist. Selleks valitakse nivooanduri analoogsignaali oletatav signaalivahemik. Oletagem, et vedelikunivoo mõõtmiseks kasutatakse andurit, mis väljastab analoogsignaali 1…5 V. Madalaima vedeliku lubatud nivoo korral (mahuti on tühi) väljastab andur pinge 1 V ja suurima lubatud nivoo korral (mahuti täis) pinge 5 V. Tabelis 2.9 on näha, et 1…5 V pingeväärtuste vahemikule vastab kümnendarvude vahemik 0…27648. Lisaks on teada, et analoogsisendi eraldusvõime on 12 bitti ja märgikoht. Seega, vaadates tabelit 3.9 ja 3.10 punktis 3.2.20, on antud analoogsisendi täpsus kümnendsüsteemis 8 (ehk 0.00116 V). Kasutaja määrab suurimale lubatud vedelikunivoole vastava anduri signaali 4,5 V ja madalamale lubatud vedelikunivoole vastava anduri signaali 1,5 V. Et kasutada programmis eeltoodud suurusi kümnendkujul, tuleb teisendada vastavad analoogväärtused kümnendkujule:

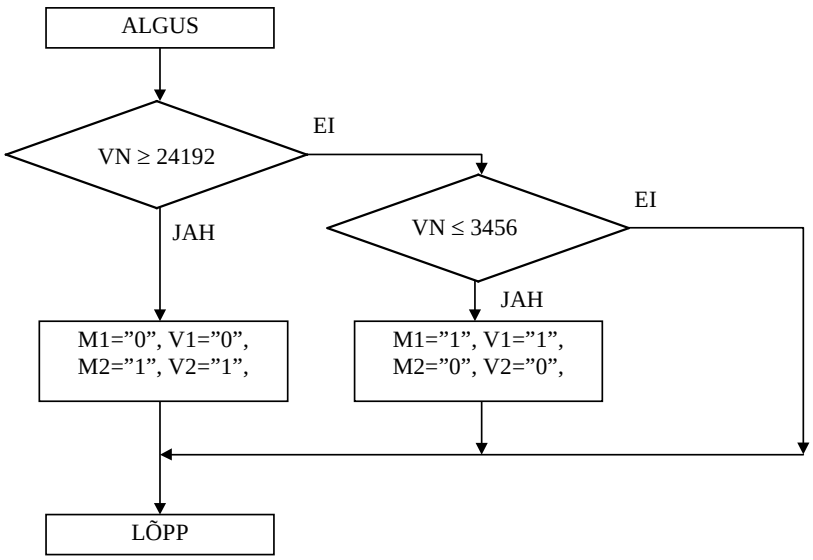
$$\frac{27648}{5-1}\cdot(1{,}5-1)=3456 \qquad \frac{27648}{5-1}\cdot(4{,}5-1)=24192$$

Kui on teada analoogväärtustele vastavad kümnendarvulised tulemid, tuleb järgnevalt kirjeldada sisselaskepumba, -ventiili, väljalaskepumba ja vaheventiili väljundite aadresse ning analoogmooduli sisendaadressi (kuhu on ühendatud analoogsignaali väljastav andur) (tabel 5.5).

Tabel 5.5 Sisenditele ja väljunditele vastavad operandid

Operand	Tähis	Kirjeldus
PEW 128	VN	Analoogsisendsuurus
A 0.0	M1	Sisselaskepump
A 0.1	V1	Sisselaskeventiil
A 0.2	M2	Väljalaskepump
A 0.3	V2	Vaheventiil

Järgmine samm on juhtalgoritmi koostamine, kus tuleb määrata, milliste analoogsuuruse piirväärtuste korral kuidas juhtsüsteem peab käituma (joonis 5.12).



Joonis 5.12. Algoritm analoogsuuruse piirväärtuste määramiseks

Tingimused programmi koostamiseks olid juba esitatud. Kuna programm on suhteliselt lühike, valitakse juhtplokiks ***OB1***, kuhu kirjutatakse juhtprogramm. Joonisel 5.13 toodud juhtprogramm iseloomustab küllaltki ilmekalt analoogsuurustega opereerimist ja juhtimiseks vajalike tingimuste kirjeldamist.

OB1

L PEW 128	Laetakse analoogsisendist aadressiga 128 analoogsuurus
T MW 100	digitaalsel kujul ja kantakse mällu aadressile 100
L MW 100	Laetakse mälust aadressil 100 paiknev väärtus ja
L +24192	laetakse sõnapikkune konstanttäisarv 24192 ning
>=I	võrreldakse; kui mälus paiknev sõna on suurem konstantsest
R A 0.0	täisarvust või võrdne, lülitatakse väljund aadressiga 0.0
	olekusse “0”,
R A 0.1	väljund 0.1 olekusse “0”,
S A 0.3	väljund 0.3 olekusse “1” ja
S A 0.2	väljund 0.2 olekusse “1”
	Võrdlus miinimumsuuruse suhtes:
L MW 100	kõigepealt laetakse mälust aadressil 100 paiknev väärtus ja
L +3456	laetakse sõnapikkune konstanttäisarv 3456 ning
<=I	võrreldakse; kui mälus paiknev sõna on väiksem konstantsest
S A 0.0	täisarvust või võrdne, lülitatakse väljund aadressiga 0.0
	olekusse “1”,
S A 0.1	väljund 0.1 olekusse “1”,
R A 0.3	väljund 0.3 olekusse “0” ja
R A 0.2	väljund 0.2 olekusse “0”.

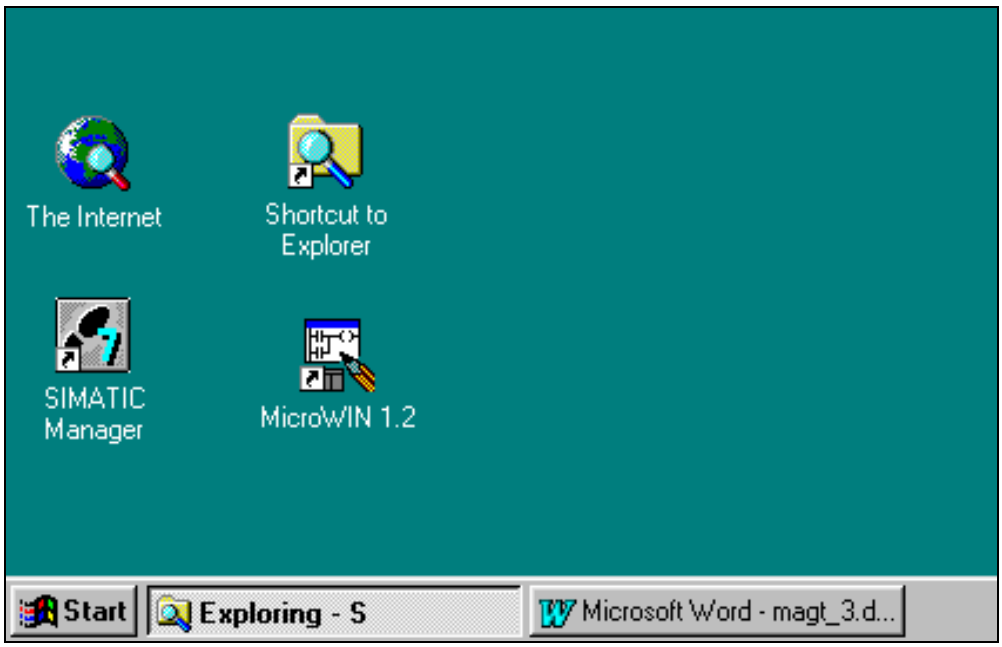
Joonis 5.13. Analoogsuuruse arvestamine juhtprogrammis

6 Töö STEP 7 keskkonnas

Nagu tänapäeva seadmetele kohane, on tarkvara lüliks inimese (programmeerija) ja riistvara vahel. Tarkvara võimaldab programmeerija poolt kirjutatud programmi viia n. ö. riistvarale arusaadavale kujule. Seega kujutab programmeeritavate kontrollerite tarkvara endast kasutajaliidest inimese suhtlemiseks masinaga.

Programmeeritavate kontrollerite tarkvara võib põhineda väga mitmesugustel operatsioonisüsteemidel, nagu näiteks *DOS*, *UNIX*, *WINDOWS 95*, *OS/2* jne. Tarkvara valmistatakse regiooniti vastavalt levinud operatsioonisüsteemidele. Euroopas võib praegu ilmselt kõige levinumaks operatsioonisüsteemideks lugeda *WINDOWS 95, 97, 98 ja NT*, vastavalt millele kohandataks ka uusimaid tööstuskontrollerite programmeerimispakette. Siiski tehakse veel ka DOS-keskkonnal baseeruvaid pakette.

Firma *SIEMENS* töötas välja kontrollerite SIMATIC S5 programmeerimiseks tarkvarapaketi, mille algversioonid baseerusid ainult *DOS* keskkonnal, hilisemad *juba* WINDOWS 3.xx keskkonnal. Praegu toimub eri jõudlusega SIMATIC-kontrollerite programmeerimine S7-200 puhul WINDOWS 3.xx keskkonnas paketiga *MicroWIN* ja S7-300, S7-400 puhul WINDOWS 95 keskkonnas tarkvarapaketiga *SIMATIC Manager* (joonis 6.1). Kompaktkontrolleri LOGO! programmeerimiseks kasutatakse Windows 3.xx keskkonnas töötavat tarkvarapaketti *LOGO! Soft* või Windows 95/98 keskkonnas töötavat tarkvara *LOGO!Soft Comfort*.



Joonis 6.1. Programmpakett *SIMATIC Manager* WINDOWS 95 keskkonnas

Kuna *MicroWIN* programm on välja töötatud suhteliselt väikese jõudlusega kontrollerile S7-200, siis on ka programmeerimispakett väga lihtne. Märksa

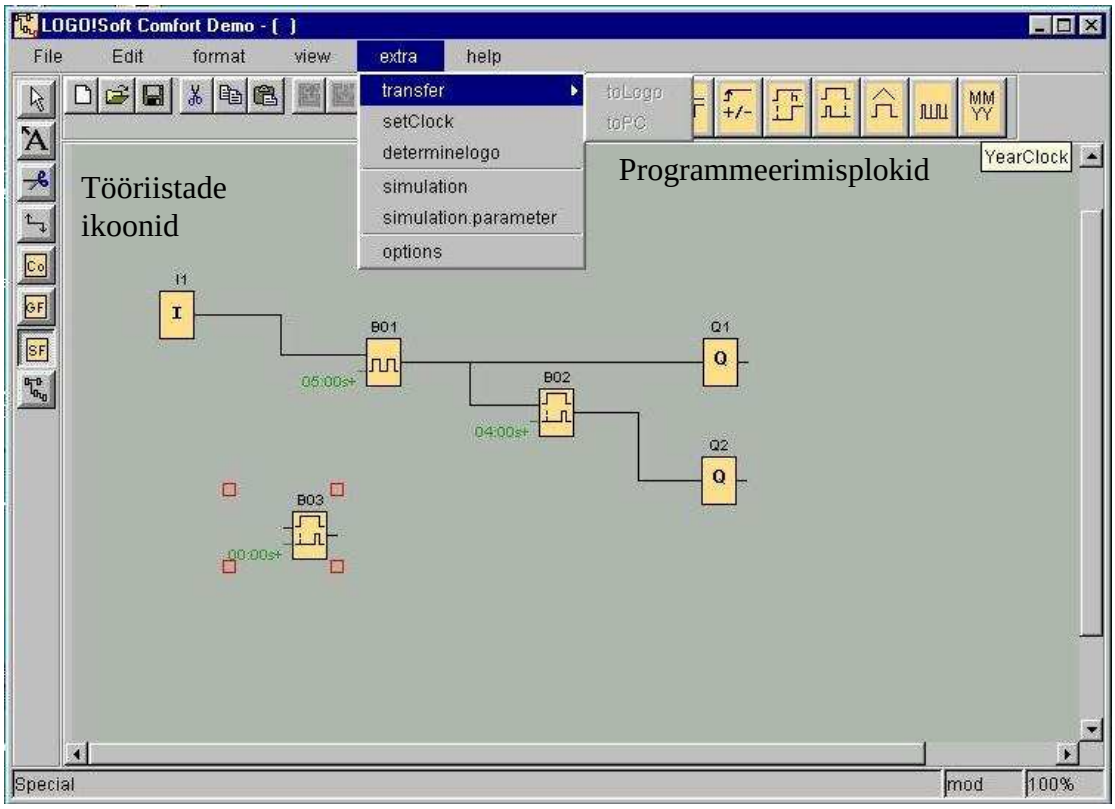
suuremate võimalustega on programmeerimine *SIMATIC Manageri* abil STEP 7 keskkonnas, mida edaspidi põhiliselt ka käsitletakse.

Töö automaatikaprojekti kallal jaotatakse SIMATIC Manager STEP 7 keskkonnas 7 osaks:

- projekti alustamine,
- riistvara konfigureerimine ja parameetrite määramine (sh. andmeside parameetrid).
- sümbolite tabeli koostamine,
- S7 kontrolleritele programmi koostamine,
- programmi laadimine kontrollerisse,
- juhtsüsteemi (programmi) testimine,
- juhtsüsteemi (programmi) diagnostika.

6.1 Tarkvara LOGO!Soft Comfort

LOGO!Soft Comfort erineb oma eelkäijast *LOGO!Soft* selle poolest, et ta võimaldab graafilisel kujul programmeerimist. Programmeerimisviisina kasutatakse loogikaskeemi. Tarkvaraga *LOGO!Soft* programmeerimine sarnanes *LOGO!* esipaneelil paikneva kasutajaliidese kaudu programmeerimisega, mis ei võimaldanud piisavat ülevaadet kogu programmist.



Joonis 6.2. Tarkvarapaketi LOGO!Soft Comfort tööaken

LOGO!Soft Comfort tööakna (joonis 6.2) vasakul pool veerus paiknevad tööriistade ikoonid, mille abil saab redigeerida joonist ja valida vastav funktsioonide kataloog nagu

- Co (kontaktid), mis sisaldab elemente sisendite ja väljundite tähistamiseks skeemil
- GF (põhifunktsioonid), mis sisaldab lihtloogikalülitusi NING, VÕI, EI, NING-EI, VÕI-EI
- SF (erifunktsioonid), mille abil saab valida erifunktsioone nagu rakendusviide, tagastusviide, triger, taktrelee, salvestav rakendusviide, asünkroonne taktrelee, nädala ja aasta kell, impulsiloendur, tunniloendur, impulsi tõusvale frondile reageeriv lülitus, impulssrelee jne. Erifunktsioone lisatakse LOGO!-le pidevalt juurde. Täpsemaid andmeid nende kohta leiab käsiraamatutest.

Programmi saab testida ka LOGO! olemasoluta menüüst *Extra➢Simulation*. Pärast simuleerimist, olles veendunud programmi õigsuses, saab programmi vastava kaabli

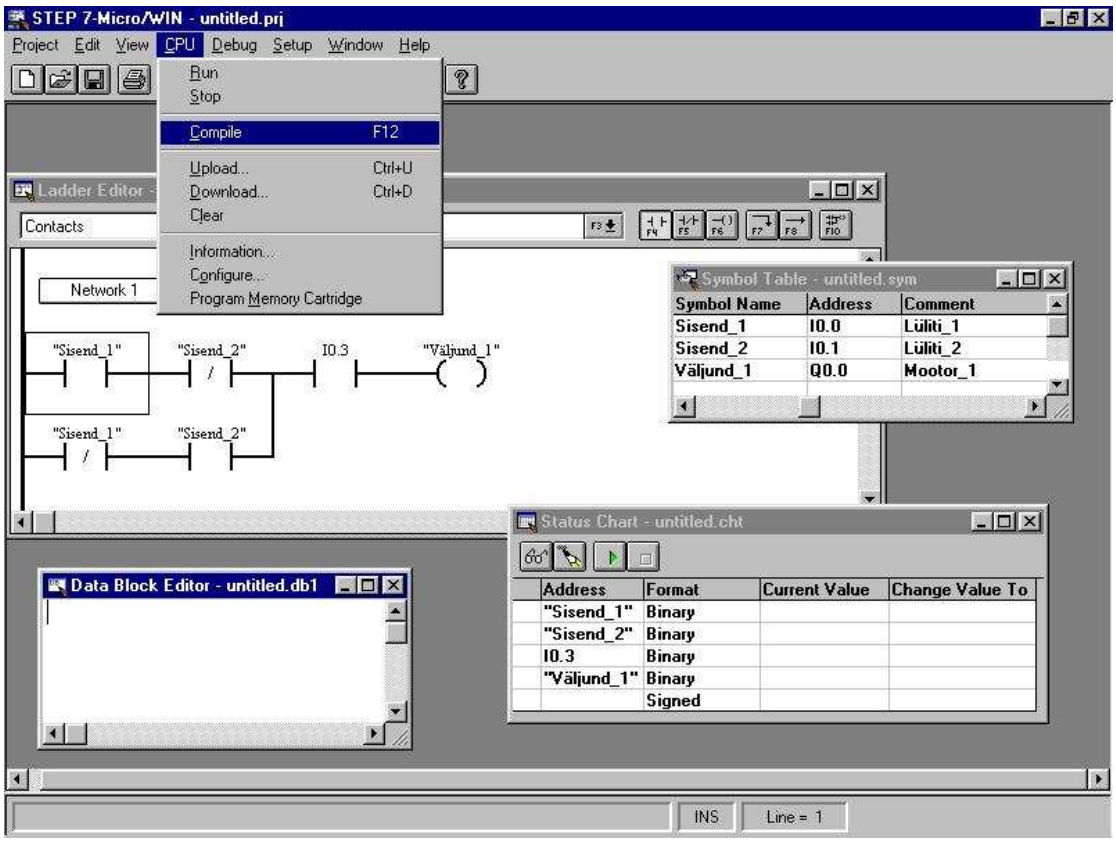
olemasolul laadida LOGO!-sse, valides menüüst *Extra➢Transfer➢toLogo*. Tarkvara võimaldab ka üsna lihtsalt kindlaks määrata, millist LOGO! antud programmi jaoks valida; selleks tuleb valida vaid menüüst *Extra➢determinelogo* ning vastav teade väljastatakse akna alla serva (joonis 6.2).

6.2 Tarkvara STEP7 MicroWIN

Uue projekti alustamisel *MicroWIN* tarkvaras tekib neli uut akent STEP7 MicroWIN aknasse (joonis 6.3):

- programmi aken (*Ladder Editor* või *STL Editor*) programmeerimiseks,
- sümbolite tabel (*Symbol Table*) aadressidele sümbolkujul nimetuste andmiseks,
- olekudiagramm (*Status Chart*) parameetrite olekute jälgimiseks ja muutmiseks,
- andmeploki redaktor (*Data Block Editor*) andmeplokkide koostamiseks.

MicroWIN tarkvara võimaldab kasutada kahte programmi esitusviisi, milleks on kontaktskeem ja käsulist, mida saab valida menüüst vastavalt *View➢STL* või *View➢Ladder*. S7-200 seeria kontrollerite programmeerimise käsud erinevad teataval määral S7-300/400 programmeerimiskäskudest (peatükk 3), kuid vastavad rahvusvahelisele standardile IEC 61131 ning on käsiraamatu abil kiiresti omandatavad.

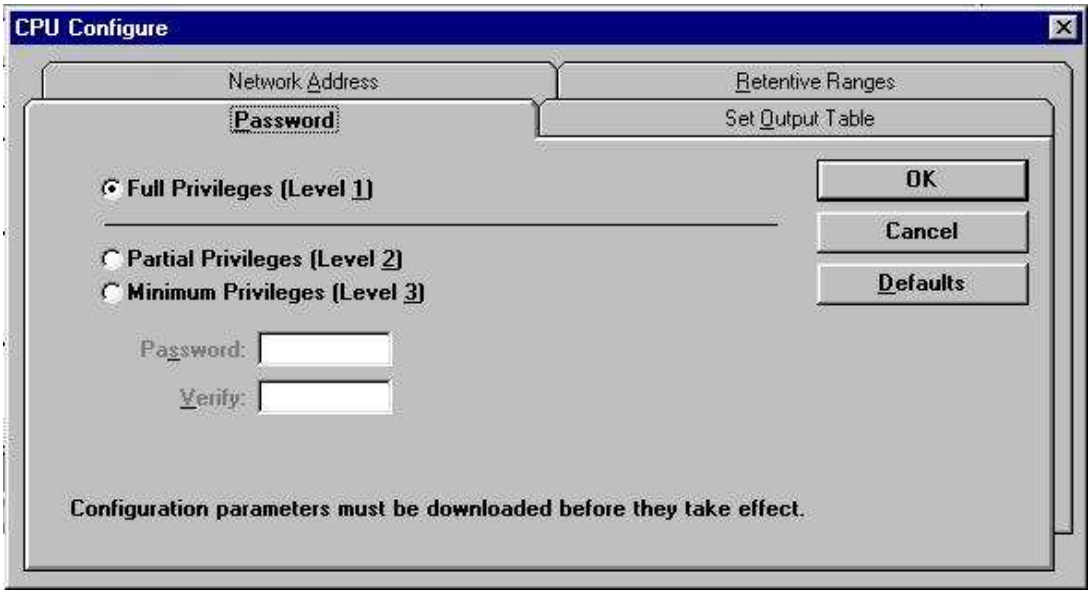


Joonis 6.3. Tarkvara MicroWIN tööaken

Pärast programmi koostamist ja enne kontrollerisse laadimist tuleb ta kompileeerida menüüst *CPU➢Compile*, mille tulemusel väljastatakse teade, et programmis on või ei ole vigu. Pärast kompileerimist võib programmi kontrollerisse laadida *CPU➢Download*, olles eelnevalt viinud kontrolleri talitlusse *STOP*. Tähelepanu tuleb

pöörata sellele, et kontrolleri küljes paiknev talitlusviisi lüliti oleks õiges asendis. Programmi töö kontrollimiseks pärast kontrollerisse laadimist tuleb valida menüüst talitlus *CPU➢Run*. Nüüd saab arvuti ekraanil ja kontrolleri valgusdioodindikaatorite abil jälgida ja testida programmi tööd vastavate sisendite ja väljundite eri olekute korral.

MicroWIN tarkvara võimaldab ka konfigureerida ja sätestada kontrollerile mitmesuguseid parameetreid nagu andmesidevõrgu aadressid, juurdepääsu privileegid programmile koos salasõnadega jne (joonis 6.4).

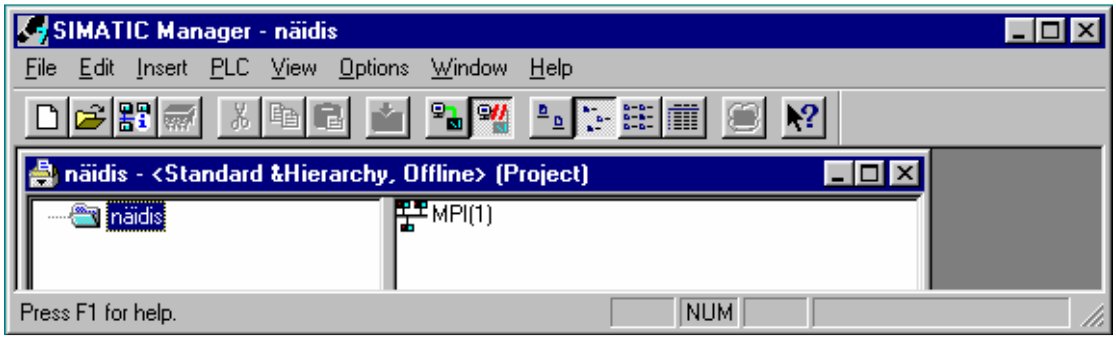


Joonis 6.4. Parameetrite määramine

6.3 SIMATIC Manager (STEP7 for SIMATIC S7-300/400)

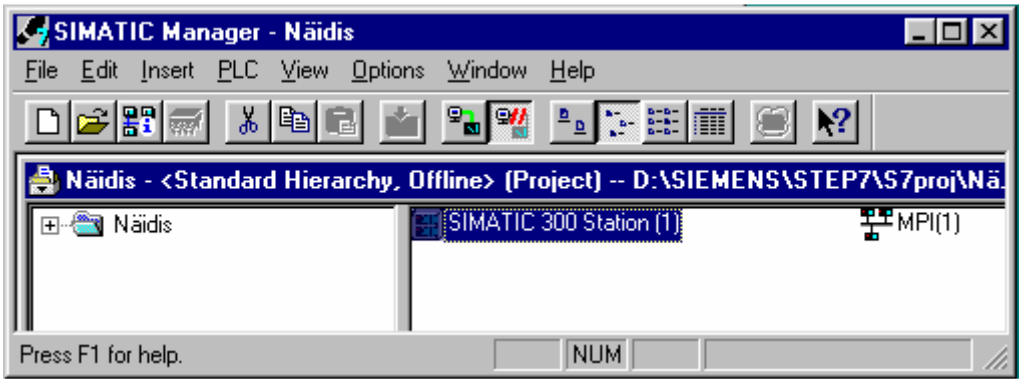
6.3.1 Projekti alustamine ja riistvara määratlemine

Projektiga alustamise eelduseks STEP 7 keskkonnas on vastav riistvara, tarkvara ning tehnoloogiaprotsessi kirjeldus ja skeem. Kui eeltoodud vahendid ja info on olemas, võib alustada süsteemi projekteerimist. Selleks tuleb käivitada programmpakett *SIMATIC Manager* ning valida menüüst *File➢New➢Project*. Seejärel tuleb kirjutada ilmunud aknasse sobiv programmi nimi ja vajutada klahvile *Save*. Kui see on tehtud, ilmub järgmine aken (joonis 6.5), mis on aluseks edaspidistele toimingutele.



Joonis 6.5. Projektiga alustamine

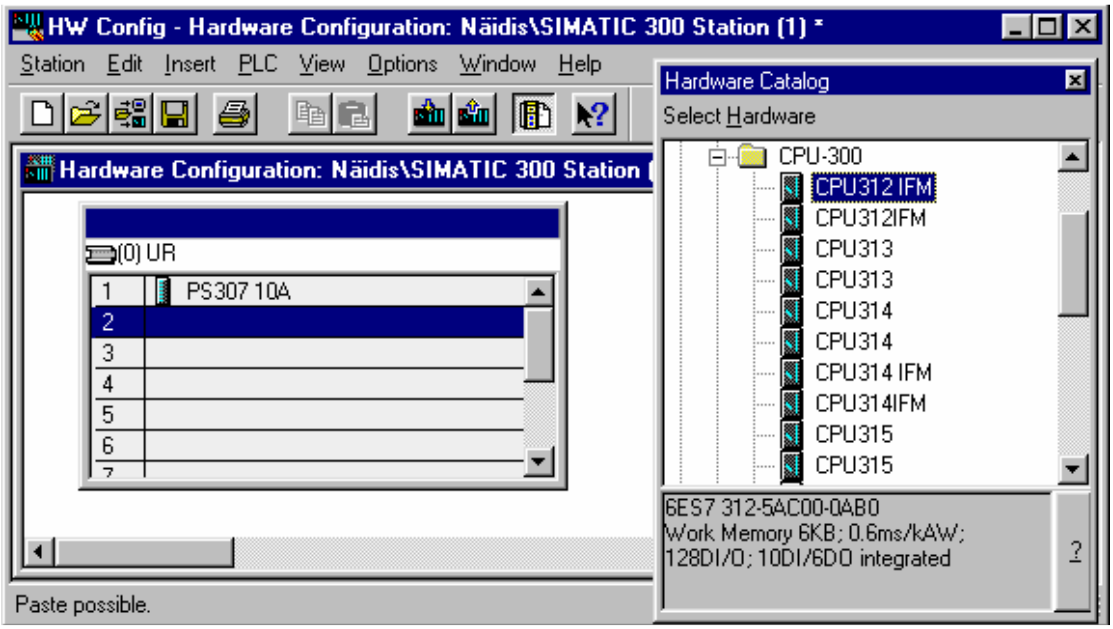
Pärast projekti alustamist, faili avamist ja talle nime andmist tuleb määratleda (konfigureerida) projektis kasutatav riistvara ja selle parameetrid. Riistvara määratlemine kujutab endast juhtsüsteemi ning tema töömoodulite valikut. Kõigepealt valitakse juhtsüsteem menüüst *Insert➢Station*, milleks antud juhul on *SIMATIC S7-300 Station*.



Joonis 6.6. Juhtsüsteem SIMATIC 300 Station (1) aktiivseks tegemine

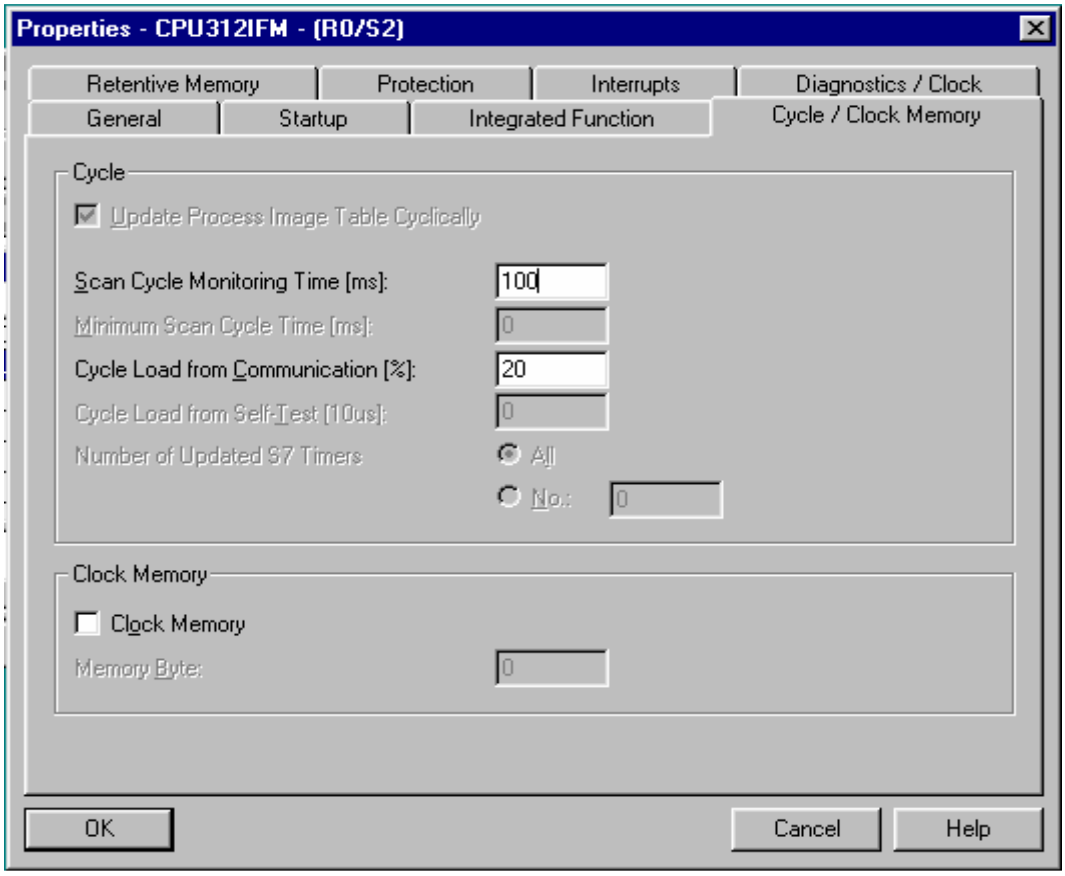
Pärast juhtsüsteemi valikut tuleb määratleda ka selle kõikidele moodulitele tööks vajalikud parameetrid (omadused, karakteristikud), s. h. moodulite adresseerimine.

Selleks aktiveeritakse projekti aknas (joonis 6.6) juhtsüsteem (SIMATIC S7-300 Station) ning avatakse menüüst *Edit➢Open Object*, mille tulemusel tekkib uus aken *Hardware Configuration*. Sellest aknast valitakse menüüst *Insert➢Hardware Components*, mille tulemusel ilmub riistvara kataloog. Kataloogi saab valida ka menüüst *View➢Catalog*. Kataloogist saab valida juhtsüsteemi jaoks vajalikke mooduleid, kandes neid hiire abil sealt riistvara konfiguratsiooni aknasse (joonis 6.7).



Joonis 6.7. Riistvara konfiguratsioon kataloogi abil

Ka lihtsaim süsteem (projekt) koosneb vähemalt koostesiinist (ingl *Rail*) ning sinna peale monteeritavast toiteplokist (nt. PS 307 10A) ja protsessoriplokist (nt. CPU 312 IFM). Siinjuures tuleb märkida, et osal protsessorplokkidest on olemas diskreetsete signaalide sisend- ja väljundliides. Kui protsessoril puudub sisend- ja väljundliides või on vaja töödelda pidevsignaale, tuleb ka lihtsate projektide puhul kasutada kas digitaalsisend- ja -väljundmooduleid või analoogsisend- ja -väljundmooduleid. Pärast juhtsüsteemi ja tema moodulite valikut alustatakse moodulite parametreerimist. Riistvara konfiguratsiooniaknas aktiveeritakse vastava moodul (nt. CPU 312 IFM). Valides menüüst *Edit➢Object Properties*, saab määrata iga konkreetse mooduli jaoks vajalikud juhtparameetrid (joonis 6.8). Näiteks saab muuta protsessori programmitsükli aega jne. Kui kõigi moodulite vajalikud parameetrid on määratud, toimub andmete salvestamine riistavara konfigureerimise aknas, valides menüüst *Station➢Save* või *Station➢Save and Compile* ning seejärel tulemuste õigsuse kontroll menüüst *Station➢Consistency Check*, et kontrollida, kas parameetrite määramisel pole tehtud vigu. Pärast seda saab laadida kogu riistavara täieliku konfiguratsiooni koos parameetritega kontrollerisse. Selleks kasutatakse menüüd *PLC➢Download*, kus saab moodulite parameetreid kas eraldi või korraga üle kanda. Eelnevalt on soovitav kustutada kontrolleri mälu, viies selle kõigepealt olekusse *STOP* (valides menüüst *PLC➢ Operating Mode* oleku *STOP*) ning kustutades seejärel mälu (avades objekti ehk kontrolleri *Station➢Open Online* ja valides menüüst *PLC➢Clear/Reset*).

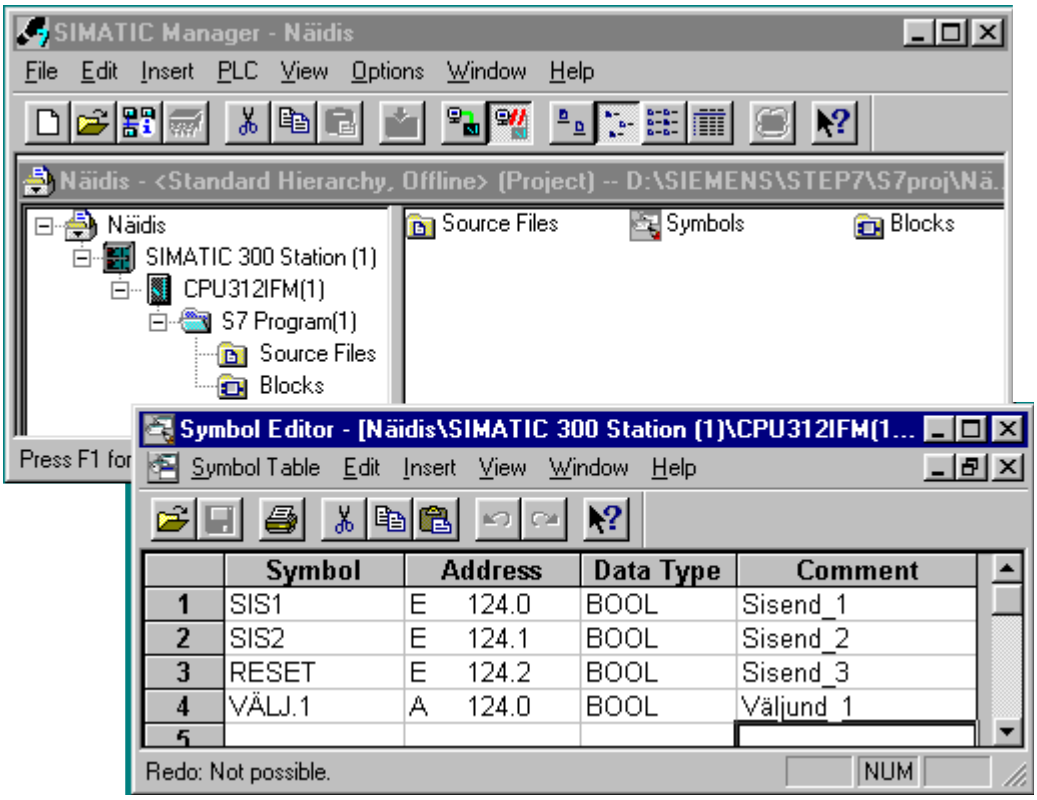


Joonis 6.8. Protsessori mooduli parameetrite määramine

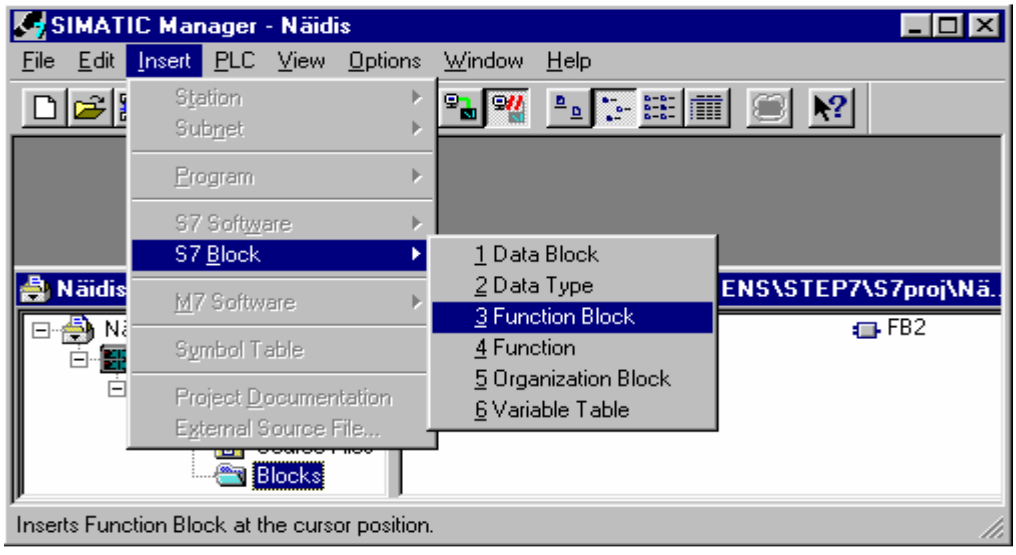
6.3.2 Sümbolite tabeli ja programmi koostamine

Sümbolite tabeli koostamisel määratakse programmis kasutatavad aadressid koos neile vastavate sümbolitega. See hõlbustab hilisemat programmeerimist, sest programmis saab näha aadresse sümbolite kujul. Sümbolite tabel luuakse automaatselt koos S7 programmi kataloogiga. Sümboltabelisse sisestatakse esmalt sümbolkujul sisendid ja väljundid, seejärel neile vastavad aadressid ning lõpuks kommentaarid (joonis 6.9). Sümboltabeli täitmiseks tuleb *SIMATIC Manageri* aknas avada kataloog *S7 Program* ning klõpsata ikoonile *Symbols*.

S7 programmi loomisel peab kasutaja teadma projekti struktuuri, mille põhjal ta loob vastavad programmiplokid. Plokke saab luua kataloogis *Blocks*, valides seejärel menüüst *Insert➢S7 Block* vastava ploki (joonis 6.10).



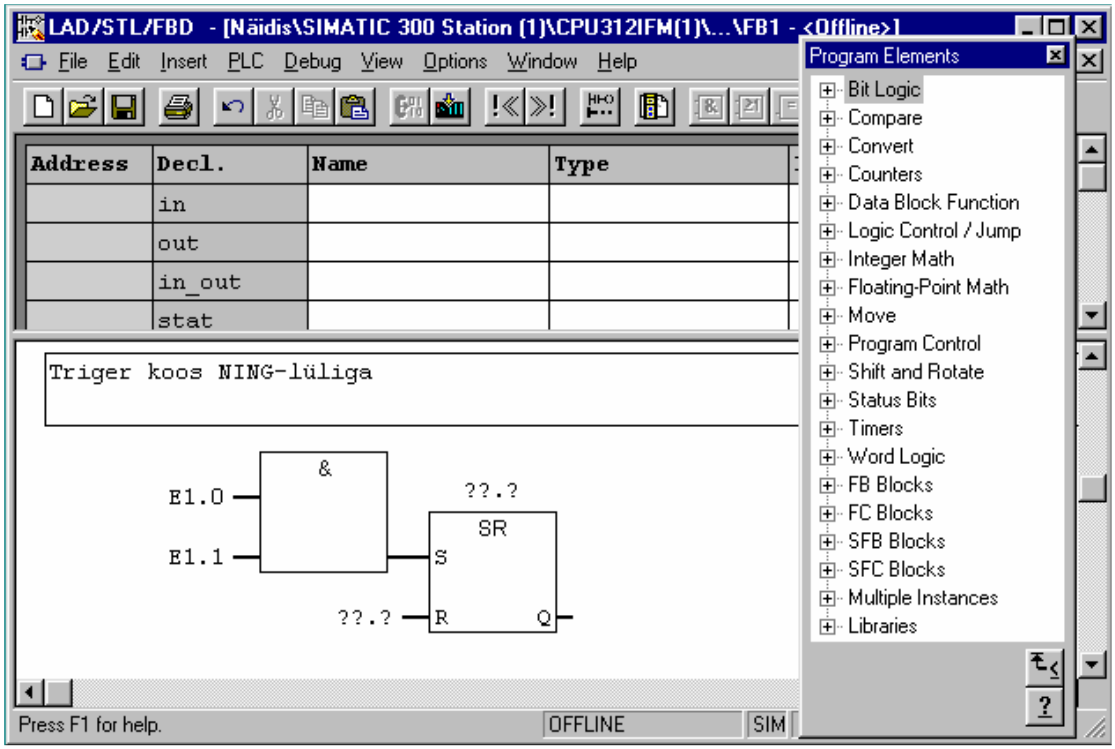
Joonis 6.9. Sümbolite tabel koos *SIMATIC Manager* tööaknaga



Joonis 6.10. Plokkide sisestamine programmi

Pärast vajaliku ploki loomist tuleb ploki programmeerimiseks sellele hiirega kaks korda klõpsata või aktiveerida ja valida menüüst *Edit➢Open Object*. Selle toimingu tulemusena tekib uus aken *LAD/STL/FBD*, kus plokk on avatud. Akna ülemine osa kujutab endast muutujate deklareerimistabelit ja alumine programmikäskude

sisestusvälja. Järgnev programmeerimine toimubki *LAD/STL/FBD* aknas (joonis 6.11).



Joonis 6.11. Programmeerimisaken

Programmeerimise alustamiseks valitakse menüüst *View* STEP7 programmeerimiskeel ehk esitusviis (LAD, STL, FBD jne.). Kui programmi esitusviis on määratud, võib alustada programmeerimist. Et sisestada kontaktaseskeemi (LAD), loogikaskeemi (FBD) või käsulistingu (STL) kujul vastavaid programmikäske, tuleb valida menüüst *Insert➢Program Elements* vastavad käsud (joonis 6.11). Uue segmendi sisestamisel kasutatakse samuti menüüd *Insert➢Network*. Kui programmiplokk on valmis, salvestatakse ta menüüst *File➢Save* kettaseadmele. Nii toimitakse iga plokiga.

6.3.3 Programmi kontrollerisse laadimine ja testimine

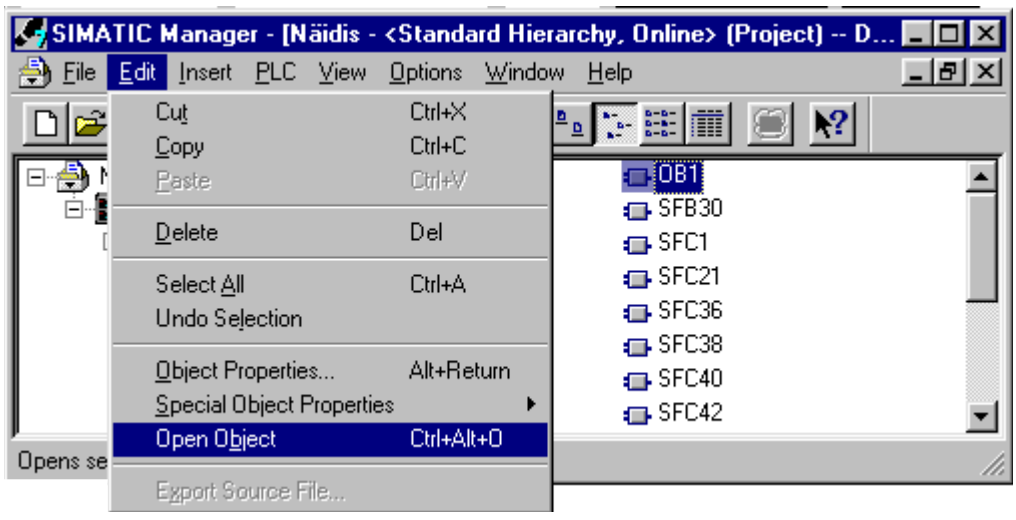
Programmi testimiseks on kaks võimalust:

- testimine kontrolleri abil,
- testimine tarkvarapaketi S7-PLCSIM abil.

Programmi testimine kontrolleri abil

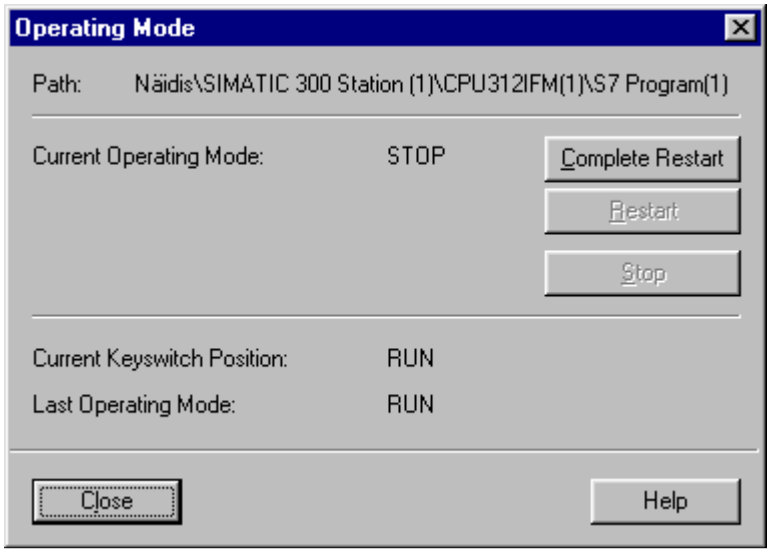
Konfigureeritud ja parameetritega määratletud programmi võib kasutaja kas tervikuna või plokkide kaupa laadida kontrolleri mällu. Enne uue programmi laadimist protsessoriplokki tuleb veenduda, et mälus pole vana programmi. Vanale programmile uue programmi pealekirjutamine võib põhjustada vigu uue programmi töös. Seetõttu tuleb vana programm eelnevalt kustutada. Peale selle tuleb jälgida, et kontrolleri ja programmaatori (PC) vahel oleks *online*-side; programm peab olema eelnevalt vigadeta kompileeritud ja protsessor (CPU) peab olema STOP talitluses.

Programmi laadimiseks tuleb *SIMATIC Manageri* aknas selekteerida ehk ära märkida plokid, mida soovitakse laadida kontrollerisse. Seejärel valitakse menüüst käsk *PLC➢Download*. Pärast programmi laadimist kontrollerisse tuleb valida menüüst *View➢Online*, et *online*-talitluses saaks programmi tööd jälgida ehk testida. Kui eelnevad operatsioonid on sooritatud, tuleb avada SIMATIC Manager aknas menüüst *Edit➢Open Object* (joonis 6.12) vastav programmiplokk. Plokk on avatud nüüd *online*-talitluses.



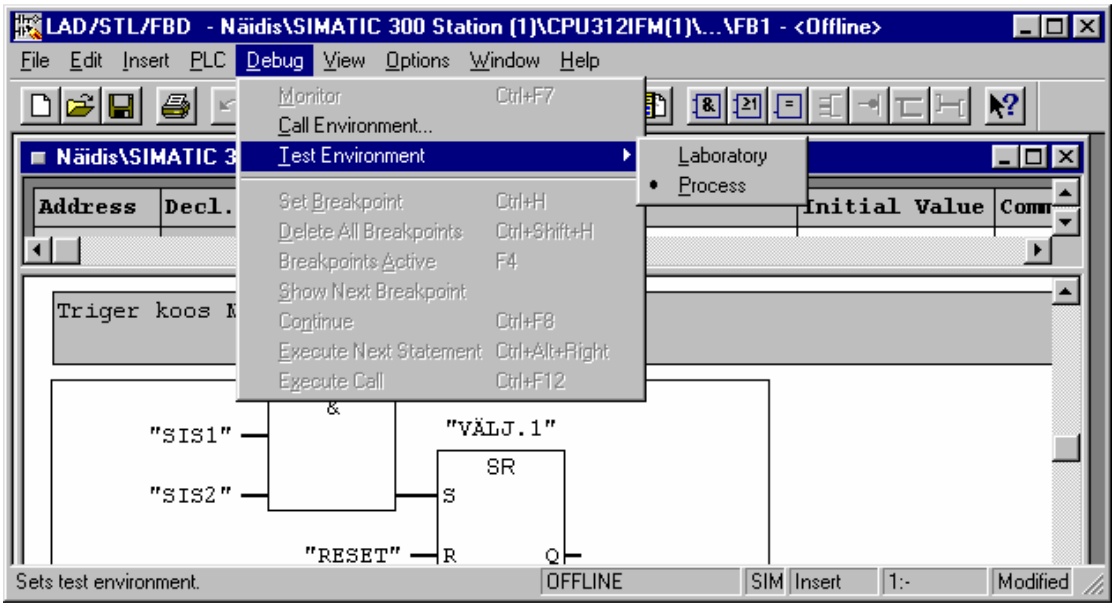
Joonis 6.12. Ploki OB1 avamine objektina

Kontroller viiakse RUN-talitlusse akna *LAD/STL/FBD* menüüst *PLC➢Operating Mode* (joonis 6.13). Selle tulemusel süttib protsessoriploki valgusdiood RUN.



Joonis 6.13. Kontrolleri talitlusviisi valik

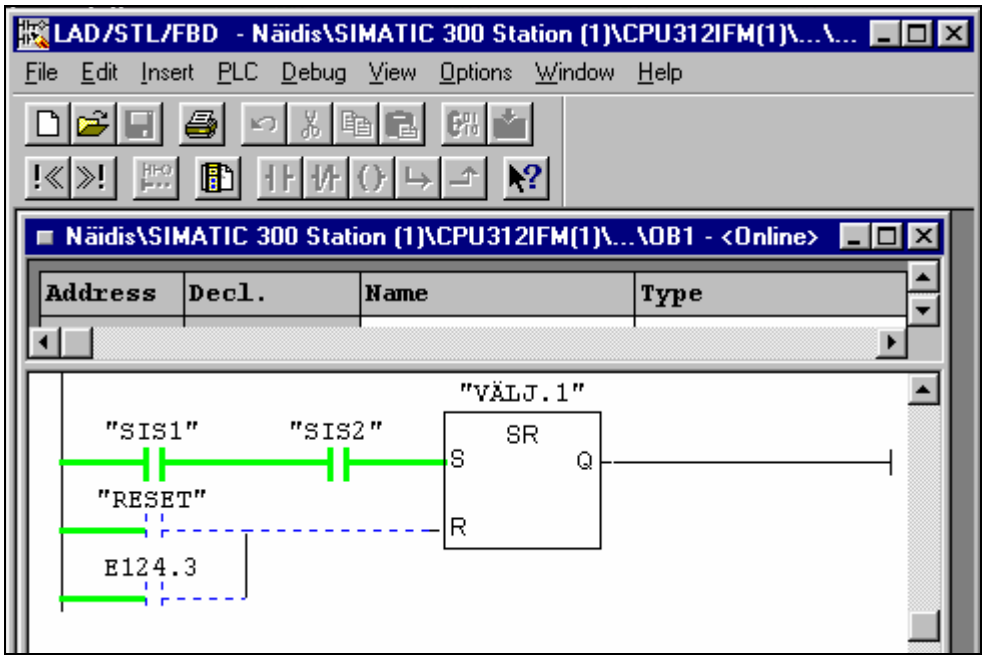
Protsessori RUN-talitluses defineeritakse menüüst *Debug➢Call Environment* testitava ploki päringu- ehk väljakutsumis- ja testimistingimused (*Trigger Condition*). Järgnevalt valitakse akna *LAD/STL/FBD* menüüst *Debug➢Test Environment➢Process/Laboratory* vastav testimisviis (joonis 6.14). Kui on valitud *Process*, siis programmplokis esitatud tsüklite ehk silmuste (*loop*) töötlust kuvatakse ekraanile üks kord, *Laboratory* korral kuvatakse programmiplokisisest silmust ekraanile pidevalt.



Joonis 6.14. Testimisviisi valik

Viimasena määratakse testimistalitluse esitusviis ekraanil, s. t. kuidas ekraanil on näha protsessi käik ehk selle olekute muutus. Selleks kasutatakse menüü *Options➢Customize* käsku. Pärast valikut ilmub aken, kus saab valida *LAD/FBD*

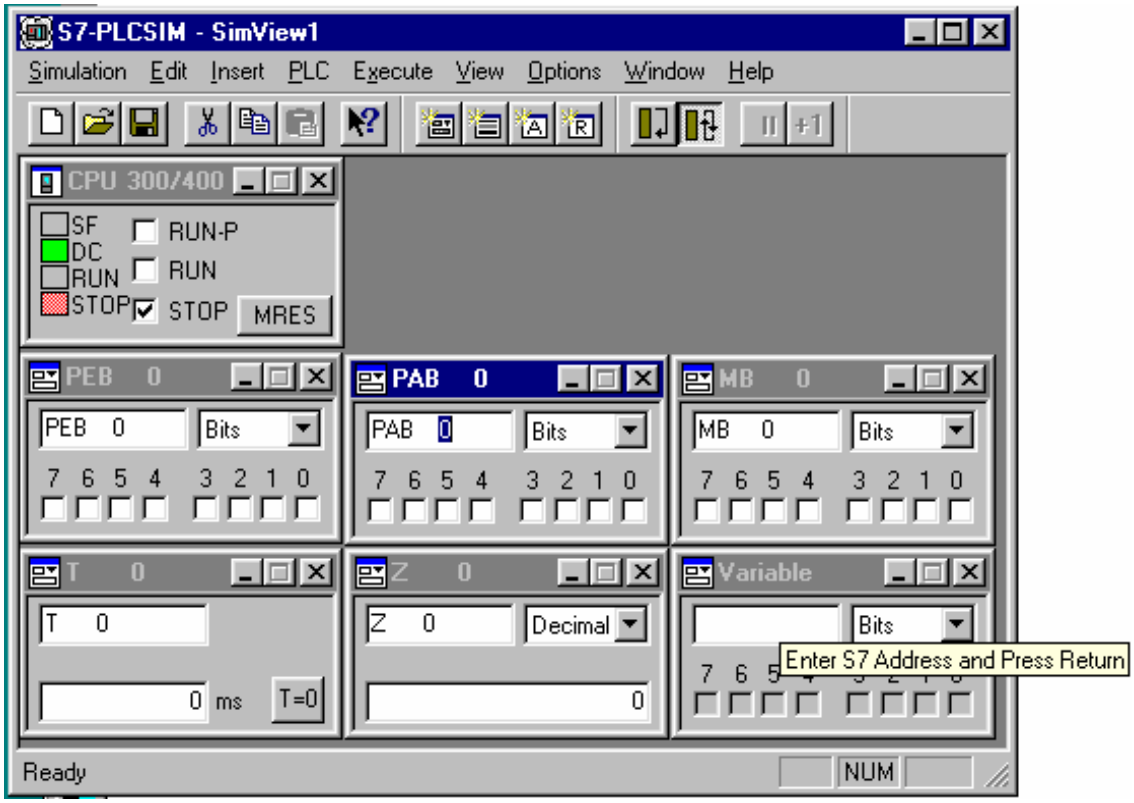
skeemi testimisel joone värvi ja paksust, *STL* esitusviisi puhul aga eri registrite olekubitte. Näiteks graafilise esituse juures (joonis 6.15) on vooluga ahel (olek “1”) teist värvi või teist tüüpi joonega kui need ahelad, kus voolu pole (olek “0”).



Joonis 6.15. Programmi testimine

Programmi testimine tarkvarapaketi S7-PLCSIM abil

Programmi saab testida ka kontrollerit jäljendava tarkvarapaketiga S7-PLCSIM. Selleks tuleb valida *SIMATIC Manager* aknas menüüst *Options➢Simulate Modules* ja seejärel laadida programm simuleeritud kontrollerisse *PLC➢Download*, mille peale tekib kuvarile uus aken S7-PLCSIM (joonis 6.16). Kasutades S7-PLCSIM menüüd saab ette anda, muuta ja jälgida simuleeritava kontrolleri talitlust ning sisendite, mälude, muutujate, viivituslülituste, loendurite ja registrite olekuid ja parameetreid. Selleks tuleb valida menüüst *Insert➢* vastav sisend, väljund jne. operand, mille peale ilmub ekraanile vastav aken, kuhu kirjutatakse jälgitava või muudetava operandi tunnus ja parameeter (aadress). Menüü *View➢* alt saab jälgida täiendavalt akude, olekusõnade, aadressiregistrite, pinuregistrite ning andme- ja loogikaplokkide olekuid. Samuti nagu kontrolleri puhul saab testimisviisi määrata menüüst *Execute➢Mode*.

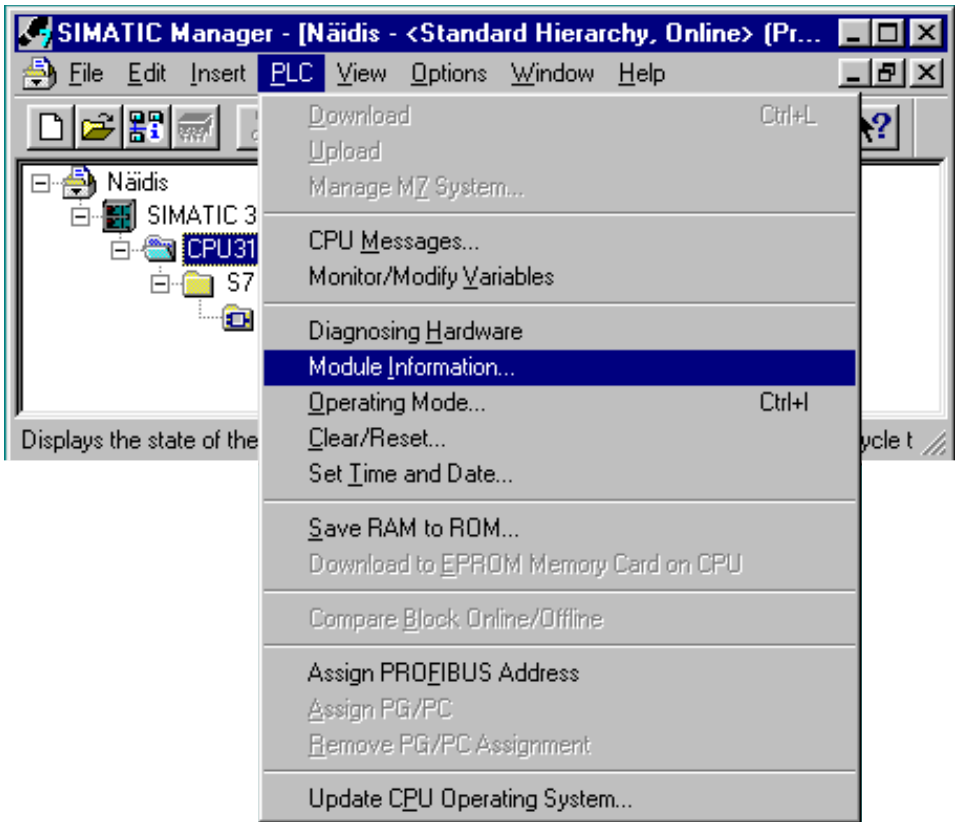


Joonis 6.16. Programmi testimine tarkvarapaketi S7-PLCSIM abi

Et kuvada programmi graafiliselt, tuleb valida *SIMATIC Manager* aknas menüüst *View➢Online*. Selle tulemusel avaneb projekti *Online* aken, kus klõpsates hiirega testitava programm- või funktsioonploki peale avatakse uus aken *LAD/STL/FBD*. Edasine toimub sarnaselt reaalse kontrolleri testimisega.

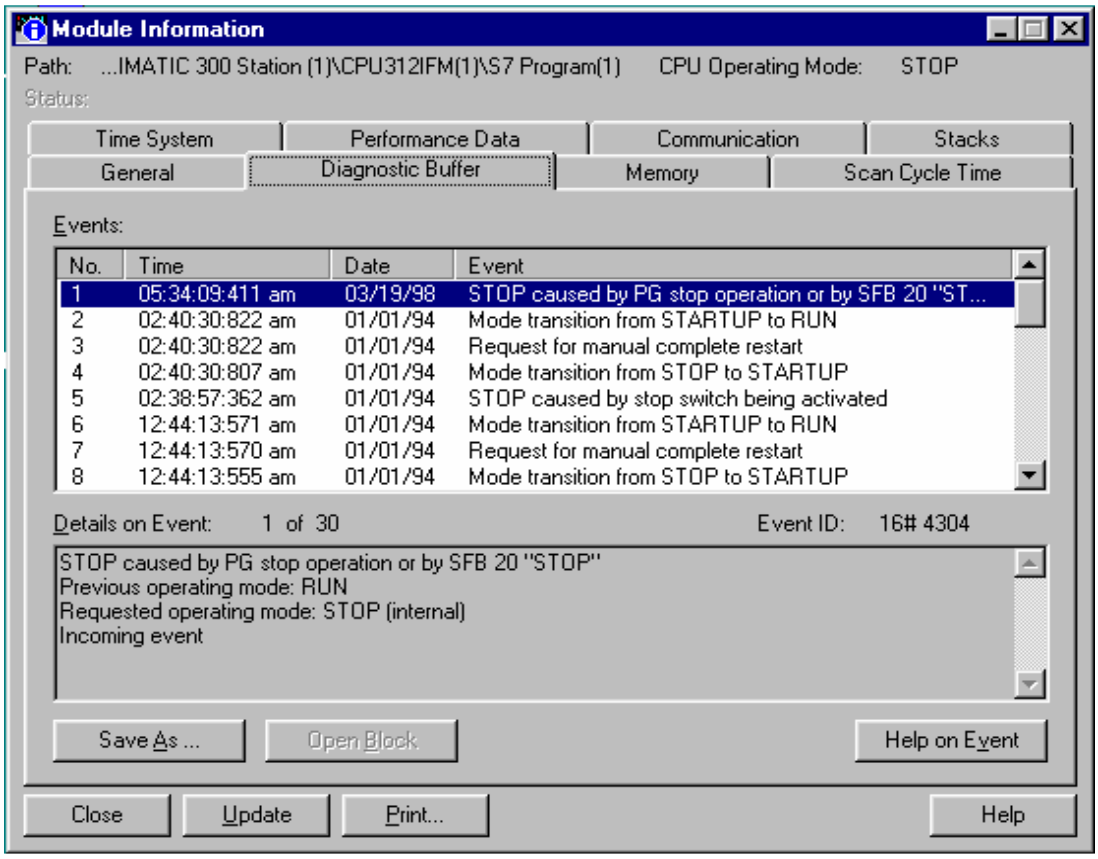
6.3.4 Kontrolleri diagnostika

Programmi moodulite info lugemiseks, vigade ennetamiseks ja tuvastamiseks, protsessori registrite uurimiseks jne. kasutatakse *diagnostikat*. Diagnostikafunktsioonidega saab kiiresti ja lihtsalt avastada programmis või riistvaras tekkinud vead, nende põhjused ja asukohad ning seejärel neid korrigeerida. Et saada infot kontrolleris toimuva kohta, võib *SIMATIC Manager* akna kataloogis *PLC* valida erinevate menüüpunktide vahel (joonis 6.17) nagu *Diagnosing Hardware*, *Module Information*, *CPU Messages* jne.



Joonis 6.17. *SIMATIC Manager'i* menüü *PLC*

Kõiki diagnostika võimalusi siin ei käsitleta. Soovi korral saab lugeja nendega tutvuda käsiraamatutest. Lähemalt vaadelgem menüü *PLC➢Module Information* valikut. Selle abil saab jälgida valitud moodulis toimuvat. Valides menüüst *PLC➢Module Information* ilmub ekraanile uus aken, milles saab jälgida valitud mooduli parameetreid (joonis 6.18), nt. protsessorimooduli puhul mälu, tsükliaega, diagnostikapuhvrisse viimasena kantud teadet, pinuregistrite sisu jne.



Joonis 6.18. Moodulite diagnostika

Joonisel 6.18 esitatud aknas on loetelus esikohal kirje (s. t. *No.1*), et 19. märtsil kell 5.35.09 enne lõunat toimus protsessori viimine *STOP*-talitlusse, mille põhjustas programmaatori poolt väljastatud *STOP*-käsk või süsteemse funktsioonploki SFB 20 toiming.

7 Kasutatud kirjandus

1. Andmevõrkudest automaatikas. Elektriku Kalender. Elektriajamite ja jõuelektroonika instituut. Tallinn. AS Printall. 1998.
2. Programmeeritavad kontrollerid. Elektriku Kalender. Elektriajamite ja jõuelektroonika instituut. Tallinn. AS Printall. 1999.
3. A.Rosin. Entwicklung einer Experimentiereinrichtung zur Untersuchung eines Industrie-Vertical-Knickarm-Roboters mit Prozessvisualisierung. Diploma work at Laboratory of Electrical Drives and Automation. Kempten: Fachhochschule Kempten 1996. 131 lk.
4. Homepage PLCopen. http://www.plcopen.org:81/
5. Homepage WESTERMO. http://www.westermo.se/
6. Homepage SIEMENS-Indutrie. http://www.ad.siemens.de/industry/index00.htm
7. Г. Р. Грейнер, В. П. Ильяшенко и.д.т. Проектирование бесконтактных управляющих логических устройств промышленной автоматики. Москва: Энергия, 1977. 384 lk.
8. Tõnu Lehtla ja Lembit Kulmar. Mikroprotsessortehnika. Tallinn: TTÜ elektriajamite ja jõuelektroonika instituut, 1995. 141 lk.
9. Programmable logic controllers teaching and using in Estonian industry. Publication of the reseach symposium of young scientists “Actual problem of electrical drives and industry automation”. Tallinn Technical University. Dept. of Electrical Drives and Power Electronics. Tallinn-Lohusalu. 1997. 60 lk.
10. Siemens. Speicherprogrammierbare Steuerungen. Grundbegriffe. Siemens AG. 2. überarbeitete und erweiterte Auflage. 1991. 83 lk.
11. SIMATIC Software. S7-300 Programmable Controller Quick Start. Automation Group. Nürnberg: Siemens AG, 1996.
12. SIMATIC Software. Statement List for S7-300 and S7-400 Programming. Automation Group. Nürnberg: Siemens AG, 1996.
13. SIMATIC Software. System Software for S7-300 and S7-400 Program Design. Automation Group. Nürnberg: Siemens AG, 1996.
14. SITRAIN Experimentierbausteinsatz. Speicherprogrammierbare Steuerungen SIMATIC S5-115U. Erlangen: Siemens AG, 1992. 63 lk.
15. SIMATIC. S7-300 and M7-300 Programmable Controllers Module Specifications. Automation Group. Nürnberg: Siemens AG, 1994.
16. The Indutrial Electronics Handbook. Editor-in-Chief J. David Irwin. Boca Raton, Florida: CRC Press LLC, 1996.
17. Your Personal PLC TUTOR. http://www.plcs.net

8 LISAD

8.1 Väikese jõudlusega kontrollerid

SIMATIC S7-200	CPU221 (endine 210)	CPU222 (endine 212)	CPU224 (endine 214)	CPU215	CPU216
Programmi-/ andmemälu	4 kB/ 2 kB	4 kB/ 2 kB	8 kB/ 5 kB	8 kB/ 5 kB	8 kB/ 5 kB
1000 kahendkäsu täimise aeg	0,37 ms	0,37 ms	0,37 ms	0,8 ms	0,8 ms
mälu bitid loendurid taimerid	256/ 256/ 256	256/ 256/ 256	256/ 256/ 256	256/ 128/ 256	256/ 128/ 256
Digitaalsisendid ja -väljundid	kuni 10, sisaldab 10	kuni 46, sisaldab14	kuni 120, sisaldab 24	kuni 120, sisaldab 24	kuni 128, sisaldab 40
Analoogsisendid ja -väljundid	-	kuni 8	kuni 22	kuni 22	kuni 22
Visualiseerimis-seadmete kasutamine	?	jah	jah	jah	jah
Andmesideliidesed	1xRS485	1xRS485	1xRS485	2xRS485, PPI	2xRS485, 2xPPI
Andmesidevõrgud	-	ASI, Profibus-DP	ASI, Profibus-DP	ASI, Profibus-DP	ASI, Profibus-DP
Reaalajakell	lisatav	lisatav	jah	jah	jah

8.2 Keskmise jõudlusega kontrollerid

SIMATIC S7-300	CPU312IFM	CPU313	CPU314IFM CPU314	CPU315 CPU315-2DP	CPU316	CPU318-2DP
Programmi-/ andmemälu	6kB/ 6kB	12kB/ 12kB	24kB/ 24kB	48kB/48kB 64kB/64kB	128kB/ 128kB	512kB/ 256kB
1000 kahendkäsu täimise aeg	0,6…1,2 ms	0,6…1,2 ms	0,3…0,6 ms	0,3…0,6 ms	0,3…0,6 ms	0,1 ms
mälu bitid loendurid taimerid	1024/ 32/ 64	2048/ 64/ 128	2048/ 64/ 128	2048/ 64/ 128	2048/ 64 128	8192/ 512/ 512
Digitaalsisendid ja -väljundid	kuni 144, sisaldab 16	kuni 128, sisaldab 0	kuni 548 (512), sisaldab 36 (0)	kuni 1024, sisaldab 0	kuni 1024, sisaldab 0	kuni 1024, sisaldab 0
Analoogsisendid ja -väljundid	kuni 32, sisaldab 0	kuni 32, sisaldab 0	kuni 69, sisaldab 5	kuni 128, sisaldab 0	kuni 128, sisaldab 0	kuni 128, sisaldab 0
Visualiseerimis-seadmete kasutamine	jah	jah	jah	jah	jah	jah
Andmesideliidesed	1xRS485, MPI	1xRS485, MPI	1xRS485, MPI	1xRS485, MPI/ 2xRS485, MPI	1xRS485, MPI	2xRS485, MPI
Andmesidevõrgud	ASI, Profibus, tööst. Ethernet	ASI, Profibus, tööst. Ethernet	ASI, Profibus, tööst. Ethernet	ASI, Profibus, tööst. Ethernet	ASI, Profibus, tööst. Ethernet	ASI, Profibus, tööst. Ethernet
Reaalajakell	-	-	jah	jah	jah	jah

8.3 Suure jõudlusega kontrollerid

SIMATIC S7-400	CPU412-1	CPU413-1 CPU413-2DP	CPU414-1 CPU414-2DP	CPU416-1 CPU416-2DP	CPU417-4DP	CPU417-4H
Programmi-/ andmemälu	kokku 48 kB	kokku 72 kB	kokku 28kB, kokku 384kB	kokku 512 kB, 800kB/1600kB	kokku 4MB, laiendatav kuni 20MB	
1000 kahendkäsu täimise aeg	0,2 ms	0,2 ms	0,1 ms	0,08 ms	0,1 ms	0,1 ms
mälu bitid loendurid taimerid	4096/ 256/ 256	4096/ 256/ 256	8192/ 256/ 256	16384/ 512/ 512	16384/ 512/ 512	16384/ 512/ 512
Digitaalsisendid ja -väljundid	4096	8192	16384	32768/65536	12000	12000
Analoogsisendid ja -väljundid	256	512	1024/2048	4096	8000	8000
Visualiseerimis-seadmete kasutamine	jah	jah	jah	jah	jah	jah
Andmesideliidesed	MPI	MPI, PROFIBUS-DP	MPI, PROFIBUS-DP	MPI, PROFIBUS-DP	MPI, PROFIBUS-DP	MPI, PROFIBUS-DP
Andmesidevõrgud	Profibus, tööst. Ethernet	Profibus, tööst. Ethernet	Profibus, tööst. Ethernet	Profibus, tööst. Ethernet	Profibus, tööst. Ethernet	Profibus, tööst. Ethernet
Reaalajakell	jah	jah	jah	jah	jah	jah

8.4 Tööstusarvuti

	SIMATIC M7-300		SIMATIC M7-400		
	CPU 388-4	FM356-4	CPU488-3	CPU486-3	FM456-4
Protsessor/ taktsagedus	80486DX2/50MHz	80486DX2/50MHz	Pentium/120MHz	Pentium/75MHz	80486DX4/75MHz
Põhimälu	8MB	4/8MB	8 kuni 32 MB	8 kuni 32 MB	4 kuni 32 MB
SRAM	64 kB	64 kB	64 kB	64 kB	64 kB
MPI-liides	Jah	-	Jah	Jah	-
Mälukaart (plug-in)	2 kuni 16 MB	2 kuni 16 MB	2 kuni 16 MB	2 kuni 16 MB	2 kuni 16 MB
Muud liidesed	1xRS232	1xRS232	Üle alammoodulite	Üle alammoodulite	Üle alammoodulite
Pesade arv alammoodulitele	-	-	2	2	2
Laiendus üle lokaalse laiendussiini	1 moodul kahele alammoodilile, 1 moodul kolmele alammoodulile 1 infosäilitusmoodul (3 ½ ” kettaseade, 520 MB kõvaketas)		Kuni kolme lokaalse mooduli kombinatsioon Kuni 3 laiendusmoodulit kolme alammooduli jaoks Kuni 1 infosäilitusmoodul (3 ½ ” kettaseade, 520 MB kõvaketas) Kuni 3 AT kandja moodulite lühikeste (kuni 164 mm) PC-kaartide jaoks		
Siendid/väljundid, Maksimum	PI (protsessi kuva): 256 baiti iga I/O Otsene adresseerimine: 32000 baiti I/O STEP7 adresseeritav 512 DI/DO või 64 AI/AO; lisaks 80 digit. Ja 30 anal. Kanalit üle M7 liidesmoodulite; lisaks PROFIBUS-DP koos C programmeerimisega		PI (protsessi kuva): 512 baiti iga I/O Otsene adresseerimine 32000 baiti I/O lisaks PROFIBUS-DP koos C programmeerimisega		

8.5 Operaatorpaneeliga kontrollerid

SIMATIC C7	C7-621 C7-621 ASi	C7-623	C7-633	C7-624	C7-634	C7-626 C7-626 DP
Programmi-/ andmemälu	32 kB, 32 kB	24 kB, 24 kB	48/64 kB, 48/64 kB	24 kB/ 24 kB	48/64 kB, 48/64 kB	96 või 128 kB, 96 või 128kB
1000 kahendkäsu täimise aeg	0,3 kuni 0,6 ms	0,3 kuni 0,6 ms	0,3 kuni 0,6 ms	0,3 kuni 0,6ms	0,3 kuni 0,6 ms	0,3 kuni 0,6 ms
Digitaalsisendid ja -väljundid	kuni 160, sisaldab 32/0	kuni 768, sisaldab 32	kuni 768, sisaldab 32/0	kuni 768, sisaldab 32	kuni 768, sisaldab 32/0	kuni 768, sisaldab 32
Analoogsisendid ja -väljundid	kuni 37, sisaldab 5/0	kuni 192 sisaldab 8/5	kuni 192 sisaldab 8/0	kuni 192, sisaldab 8/5	kuni 192, sisaldab 8/0	kuni 192, sisaldab 8/5
Andmeside liidesed	MPI AS-Interface	MPI	MPI PROFIBUS-DP	MPI	MPI PROFIBUS-DP	MPI PROFIBUS-DP
Operaatorpaneel	LED valgustus, LC kuvar	LED valgustus, LC kuvar	LED valgustus, LC kuvar	LED valgustus, LC kuvar	LED valgustus, LC kuvar	CCFL valgustus, LC kuvar
Joonte/tähemärk. arv reas; resolutsioon	2/20 (5mm kõrgus)	4/20 (5mm kõrgus)	4/20 (8mm kõrgus)	4/20 (5mm kõrgus) või 8/40 (4,5mm kõrgus)	4/20 (11mm kõrgus) või 8/40 (6mm kõrgus)	320 x 240 punkti
Operaatorpaneeli nuppude arv	5	4	4	8	8	14
Funktsioonide nupud koos valgustusega (LED)	-	16	16	16	16	10
Printeri liides	-	Jah	Jah	Jah	Jah	Jah
Graafika/teksti mälu	128 kB	128 kB	128 kB	256 kB	256 kB	1 MB

8.6 Tööjaamad

	Raamsüsteem PC RI25/45	Lameda kuvariga PC FI25/45,	 FI15/FI10	Raamsüsteem PC BI10/BI45 P II
Protsessor	RI25(P): Pentium/Pentium MMX RI45: Pentium MMX RI45 P II: Pentium II	FI25: Pentium/Pentium MMX FI45: Pentium II	Pentium	BI10: Pentium BI45 PII: Pentium II
Töömälu	RI25: 8 kuni 128 MB, RI45: 16 kuni 128 MB	FI25: kuni 128 MB FI45: kuni 384 MB	32 kuni 128 MB	BI10: kuni 128 MB BI45: kuni 384 MB
Kõvaketas	RI25(P): 2,1 GB RI45/RI45 PII: 5,1 GB	FI25: 2,1 GB EIDE FI45: 2,5 GB ATA-33-IDE	2,1 GB EIDE	BI10: 2,1 GB EIDE BI45 PII: 2,5 GB ATA-33-IDE
Vabad pesad	RI25(P): 2xPCI, 6xISA RI45: 2xPCI, 5xISA RI45 PII: 2xPCI, 3xISA long, 3xISA long	FI25: 5/4 x ISA FI45: 2 x PCI, 1x PCI/ISA, 2xISA	1 x PCI/ISA, 1 x ISA, 1 x PCMCIA Type III	BI10: 1 x PCI/ISA, 1x ISA, 1x PCMCIA Type III BI45 PII: 2 x PCI, 1x PCI/ISA, 2 x ISA
Graafika	Väline monitor SVGA 1024x768 1600 x 1200 (RI45 PII)	Lameekraan FI25: 10,4” DSTN/TFT, värviline VGA/SVGA FI25: 13,3” TFT, värviline XGA	Lameekraan 10,4” DSTN/TFT, värviline VGA	Väline monitor SVGA 1024x768
Andmesideliidesed	MPI, COM1, COM2, LPT1, VGA, hiir, klaviatuur	MPI, COM1, COM2, LPT1, VGA, klaviatuur, FI45: lisaks PROFIBUS-DP	MPI, COM1, COM2, LPT1, VGA, hiir, klaviatuur,	MPI, COM1, COM2, LPT1, VGA, hiir, klaviatuur, BI45 PII: lisaks PROFIBUS-DP