HAMK
Hämeen ammatti-
korkeakoulu

Kappaletavaralaitteiston mallinnus

Ammattikorkeakoulututkinnon opinnäytetyö

Sähkö- ja automaatiotekniikan koulutus, Insinööri (AMK)

Kevät 2024

Aleksi Turunen

HAMK
Hämeen ammatti-
korkeakoulu

Koulutuksen nimi Sähkö- ja automaatiotekniikan koulutus
Tekijä Aleksi Turunen
Työn nimi Kappaletavaralaitteiston mallinnus
Ohjaaja Juha Sarkula, Mika Oinonen

Tiivistelmä
Vuosi 2024

Tässä opinnäytetyössä käytiin läpi Valkeakoskella sijaitsevan Hämeen ammattikorkeakoulun ”Sorting station” (suom. Jakaja-asema) nimisen kappaletavaralaitteiston mallinnusta. Mallinnuksessa käytettiin Siemensin “SIMIT SP” nimistä mallinnusohjelmistoa. Tässä opinnäytetyössä toteutettiin esimerkki PLC-ohjelma kyseiselle kappaletavaralaitteistolle, käyttäen ”TIA Portal” nimistä ohjelmistoa. Tämän lisäksi tutkittiin asioita, mitä lukijan olisi hyvä ymmärtää tämän aiheen lisäksi, kuten mikä oli kappaletavara-automaatio, PLC perusteet, Siemens, SIMIT ja TIA Portal. Tavoitteena oli myös luoda riittävän realistinen ja visuaalisesti aidonkaltainen virtuaalimalli, jota voitaisiin käyttää opetusmateriaalina koulun ”virtuaalilaboratorio” tyyppisessä opetuksessa. Tavoitteisiin kuului myös erillisen käyttöohjeen luonti virtuaalimallin jatkokehitystä varten. Tämä antaa opiskelijalle mahdollisuuden luoda PLC-ohjelman kyseiseen virtuaalimalliin ennen todellista laboratoriotyötä.

Tämä opinnäytetyö toteutettiin kolmessa osassa, joista ensimmäisessä osassa työ toteutettiin fyysiselle jakaja-asemalle käyttämällä TIA Portal -ohjelmistoa koululla. Toisessa osassa mallinnettiin jakaja-asema ja sen toiminta SIMIT SP -mallinnusohjelmistolla. Viimeisessä osassa luotiin uusi TIA Portaalin ohjaus samaiseen SIMIT SP -mallinnusohjelmiston mallinnukseen. Samalla viimeisessä osassa tehtiin käyttöohje SIMIT SP -mallille sekä luotiin komponentti SIMIT CTE -komponentin mallinnustyökalulla.

Tämän opinnäytetyön lopputulokseksi saatiin onnistuneet kappaletavaralaitteistojen mallinnukset, jotka toimivat jokaisessa projektin vaiheessa. Tämän työn ollen jakautuneena kolmeen osaan ja jokaisessa osassa oli omat haasteensa, silti tämä opinnäytetyö onnistui moitteetta.

Avainsanat PLC, Siemens, SIMIT CTE, SIMIT SP, TIA Portal
Sivut 60 sivua ja liitteitä 3 sivua

HAMK
Hämeen ammatti-
korkeakoulu

Name of Degree Programme in Electrical and Automation engineering Abstract
Author Aleksi Turunen Year 2024
Subject Modeling of piece goods equipment
Supervisors Juha Sarkula, Mika Oinonen

This thesis examines the modeling of the piece goods equipment called “Sorting station” of Häme university of Applied sciences located in Valkeakoski, using Siemens' modeling software SIMIT SP, and this thesis also examines example of a PLC program would be best suited for this exact piece goods equipment. In addition, this thesis introduces the foundations of this topic, such as what is piece goods automation, PLC basics, Siemens and SIMIT. The goal in this thesis project has been also to create a sufficiently realistic and visually authentic SIMIT model that can be used as teaching material in the school's virtual laboratory, this also includes a separate instruction manual for the further development of the SIMIT model. This allows the student to create a PLC program for that SIMIT model before conducting the actual lab work.

This thesis project was implemented in three parts, the first part of which was the work on the TIA Portal software on campus. In the second part, the sorting station and its operation were modeled in the SIMIT SP modeling software. In the last part, a new TIA Portal control was created for the same SIMIT SP modeling software model. At the same time, in the last part, a user manual was made for SIMIT SP model and a component was created with the SIMIT CTE component modeling tool.

The result of this thesis was the successful modeling of piece goods equipment, which was found to be functional in every step of the project. As this work was divided into three parts, each part had its own challenges, yet this thesis was successfully completed.

Keywords PLC, Siemens, SIMIT CTE, SIMIT SP, TIA Portal
Pages 60 pages and appendices 3 pages

Sisällys

1 Johdanto 1

2 PLC perusteet 1

2.1 Mikä on PLC? 2

2.2 Historia 3

2.3 PLC-ohjelmointi 4

2.3.1 Mitä on PLC-ohjelmointi? 5

2.3.2 Käyttöliittymä 5

3 Siemens 6

3.1 Historia 6

3.2 Simatic 7

3.2.1 Simatic S5 7

3.2.2 Simatic S7 8

3.3 Siemens SIMIT 8

4 Työssä käytettävät tarvikkeet 9

4.1 Festo MPS-alusta ”Sorting station” 9

4.2 Työssä käytettävät ohjelmat 11

4.2.1 Siemens TIA Portal 11

4.2.2 PLCSIM Advanced 12

4.2.3 Siemens SIMIT SP 13

4.2.4 SIMIT CTE 13

4.3 Koulun ohjausjärjestelmä 13

5 TIA Portal -ohjelmoinnin toteutus virtuaalimallille 14

5.1 Virtuaalisen jakaja-aseman toiminnankuvaus 15

5.2 Kappalelaskurien suunnittelu 16

5.3 Sekvenssiohjelman suunnittelu 18

5.4 Erikoistilanteiden määrittäminen 20

5.5 Manuaaliajon mahdollistaminen 22

5.6 Käyttöliittymän luonti 23

5.7 Virtuaalimallin ohjauksen eroavaisuudet fyysisen ohjauksen kanssa 30

6 Virtuaalimallin toteutus 32

6.1 Kuljettimen alkupää ja stoppari 33

6.1.1 Kuljettimen alkupäästä stopparille 33

6.1.2 Stopparin anturitietojen mallinnus 35

6.1.3 Laskurin toiminta stopparilla 37

6.2 Stoppari-kääntäjän väli 39
6.3 Kääntäjä-alumiiniradan väli 40
6.4 Erikoistilanteiden simulointi 43
6.5 Anturien ja alkuradan kappaleiden visualisointi 47
6.5.1 Komponentit virtuaalimallin visualisointiin 48
6.5.2 Virtuaalimallin esimerkkitilanteet 51
6.6 Resettien mallinnus 52
6.7 Käyttöohjeiden laatiminen SIMIT SP:stä 54
6.8 SIMIT CTE 54
7 Yhteenveto 55
Lähteet 57

Kuvat, kaavat ja taulukot

Kuva 1. An original Modicon PLC. Image used courtesy of Schneider Electric 4

Kuva 2. Festo MPS-alusta "Sorting station" 11

Kuva 3. TIA Portal oletusnäkymä 12

Kuva 4. Koulun PLC 14

Kuva 5. Fyysinen I/O-rajapinta 15

Kuva 6. Jakaja-aseman alkupää 17

Kuva 7. Jakaja-aseman sekvenssi 19

Kuva 8. Sekvenssin aloituksen siirtymät 19

Kuva 9. Mustan värisen kappaleen tunnistus 20

Kuva 10. Kääntäjien vikatilanteet 21

Kuva 11. Tunnistamaton kappale ja muut häiriöt 22

Kuva 12. Manuaaliajo 23

Kuva 13. Käyttöliittymän rata täynnä ja kappale tunnistettu 24

Kuva 14. Käyttöliittymä kappaleiden visualisointi radoilla 24

Kuva 15. Kappaleiden tunnistusta päänäytölle 25

Kuva 16. Käyttöliittymä MainScreen suunnittelukuva 26

Kuva 17. Käyttöliittymä MainScreen esimerkki 1 26

Kuva 18. Käyttöliittymä MainScreen esimerkki 2 27

Kuva 19. Käyttöliittymä ManualScreen manuaalitilassa 28

Kuva 20. Käyttöliittymä ManualScreen automaattitila 28

Kuva 21. Käyttöliittymä AlarmScreen, kun ei ole hälytystä 29

Kuva 22. Käyttöliittymä AlarmScreen hälytyksellä ja selityksellä 29

Kuva 23. Laskurin ajastin fyysiselle jakaja-asemalle 30

Kuva 24. Fyysisen jakaja-aseman sekvenssiohjaus 31

Kuva 25. Fyysisen jakaja-aseman automaatti-/manuaaliohjaus 32

Kuva 26. Kuljetin stopparille VP 35

Kuva 27. Vaaleanpunaisen kappaleen anturitiedot 36

Kuva 28. Mustan kappaleen anturitiedot 37

Kuva 29. Metallisen kappaleen anturitiedot 37

Kuva 30. Vaaleanpunaisten kappaleiden laskuri 38

Kuva 31. Vaaleanpunaisen kappaleen mallinnus stopparilta kääntäjälle 40

Kuva 32. Vaaleanpunaisen kappaleen mallinnus kääntäjällä 42

Kuva 33. Vaaleanpunaisten kappaleiden mallinnus alumiiniradalla 43

Kuva 34. Tunnistamaton kappale 44

Kuva 35. Ensimmäinen kääntäjä ei käänny 46

Kuva 36. Ensimmäinen kääntäjä ei palaudu 47

Kuva 37. Visualisoinnin työtila 49

Kuva 38. Kuljettimen moottori 50

Kuva 39. Päänäyttö offline tilassa 50

Kuva 40. Esimerkkitilanne 1 51

Kuva 41. Esimerkkitilanne 2 52

Kuva 42. Reset työtila 1/2 53

Kuva 43. Reset työtila 2/2 54

Kuva 44. SIMIT CTE komponentti 55

Kaava 1. Integraatiokomponentin kaava 34

Kaava 2. BFormulan kaava 36

Taulukko 1. Fyysisen jakaja-aseman I/O:t ja niiden linkitys TIA Portaalissa 10

Liitteet

Liite 1. Laskurin ajastin ennen stopparia

Liite 2. Kappalelaskuri TIA Portaalissa

Liite 3. SIMIT SP -käyttöohjeen kansilehti

1 Johdanto

Tämä opinnäytetyö tehtiin Valkeakoskella sijaitsevalla Hämeen ammattikorkeakoululle. Tämän opinnäytetyön tavoitteena on suunnitella riittävän realistinen kappaletavaralaitteiston malli Siemensin SIMIT SP -mallinnusohjelmistoon, jota voitaisiin käyttää ”virtuaalilaboratorio” tyyppisessä opetuksessa. Tämän opinnäytetyön työkohde on Feston MPS-alusta ”Sorting station”, jonka toiminta perustuu eri materiaalista koostuvien sekä eri värillisten kappaleiden jakamiseen omille kuljettimille. Tämän opinnäytetyön tavoitteet jakautuvat kahteen osioon, jotka ovat PLC-ohjelman vaatimukset ja virtuaalimallin vaatimukset.

PLC-ohjelman tavoitteet ovat dokumentaation mukainen toiminta, johon kuuluu erikoistilanteet (tunnistamaton kappale ja kääntäjäviat), dokumentaation mukainen I/O-rajapinta, HMI-käyttöliittymä, josta voidaan ohjata manuaalisesti yksittäisiä liikkeitä sekä käynnistämään automaatiosekvenssin ja selkeä rakenne hyvin dokumentoidulle ohjelmakoodille. Fyysisen jakaja-aseman PLC-ohjelma toteutettiin opinnäytetyön ensimmäisessä vaiheessa ja virtuaalimallin PLC-ohjelma toteutettiin viimeisessä vaiheessa virtuaalimallin kanssa.

Virtuaalimallin tavoitteet ovat luoda riittävän realistinen malli, johon kuuluu erikoistilanteiden simulointimahdollisuus, kappaleiden sijainnin visualisointi, antureiden ja toimintalaitteiden tilan visualisointi, ja riittävän realistinen toimintamalli Festo MPS-alustasta. Muita tavoitteita ovat rakenteen selkeys, erillisen käyttöohjeen tekeminen virtuaalimallille jatkokehitystä varten ja geneeristen kirjastokomponenttien kehittäminen. Virtuaalimalli toteutettiin opinnäytetyön toisessa vaiheessa.

Automaatioinsinöörin olisi hyvä ymmärtää yleisesti mikä on kappaletavara-automaatio ja miten eri laitteistojen käyttötarkoitus näkyy tuotannossa. Myös olisi erinomaista ymmärtää, miten eri laitteistot toimivat kappaletavara-automaatio ympäristössään niin fyysisesti kuin PLC-ohjelmana ja tähän kuuluu kaikki erikoistilanteet kuten mahdolliset vikatilanteet.

2 PLC perusteet

Tässä luvussa käydään läpi PLC perusteita, sillä ne olisi hyvä ymmärtää tämän opinnäytetyön kannalta. Tässä luvussa vastataan kysymyksiin, mikä on PLC, minkälainen historia PLC:llä on ja miten se on mahdollistanut tämän opinnäytetyön, mikä on

kenttälaitteiden merkitys PLC-ohjelmoinnissa, ja kerrotaan yleisesti PLC-ohjelmoinnista sekä käyttöliittymistä.

2.1 Mikä on PLC?

PLC on lyhenne ”programmable logic controller” (suom. Ohjelmoitava logiikkaohjain), jolla ohjataan yleisesti mitä tahansa kappaletavara-, kone- ja prosessiautomaatiota. Ohjelmoitavaa logiikkaa voitaisiin esimerkiksi käyttää kuljetinjärjestelmän automatisoinnissa. Kuljetinjärjestelmän tulosignaaleina voisivat olla kuljettimen omat paikka-anturit ja mahdolliset kääntimien tila-anturit. Lähtösignaaleina voisivat olla kuljettimen moottori tai kuljettimessa olevat pysäyttäjät. Yleisesti ohjelmoitavalla logiikalla pyritään aina automatisoimaan sellaisia työkohteita, jotka ovat toistuvia, kuten autokorin rakentaminen, tai kuljetinjärjestelmiä. (Salila, 2015, ss. 5–6)

Ohjelmoitava logiikkaohjain muistuttaa rakenteeltaan tietokonetta, mutta eroaa tietokoneesta kuitenkin hyvin paljon. Ohjelmoitavan logiikkaohjaimen sisältä löytyy mikroprosessori (CPU), joka toimii logiikkaohjaimen keskusyksikkönä, eli niin sanottuna aivoina. Tähän keskusyksikköön yhdistyy erinäiset tulosignaalit, jotka ovat joko digitaali- (DI) tai analogituloja (AI). Samaan keskusyksikköön yhdistyvät lähtösignaalit, jotka ovat samalla tyylillä joko digitaali- (DO tai DQ) tai analogilähtöjä (AO tai AQ). Mikroprosessorin työ on suorittaa ohjauskäskyjä sekä kommunikoida muiden laitteiden kanssa, jotka ovat yhdistettynä kenttäväylään. Mikroprosessori suorittaa myös lukuisia aritmeettisia sekä loogisia operaatioita ja sisäistä diagnosointia. Ohjelmoitavan logiikkaohjaimen sisältä löytyy kahta erilaista muistityyppejä, kuten tietokoneesta, eli RAM- (Random-access memory) ja ROM-muisti (Read only memory). Ohjelmoitava logiikkaohjain käyttää ROM-muistia vain lukemiseen, johon on valmiiksi tallennettu käyttöjärjestelmä ja johon tehdään muutoksia ohjelmointiin. RAM-muistia käytetään lähtö- ja tulolaitteiden tilatietojen, ajastimien ja laskureiden arvojen tallentamiseen. RAM-muisti on yleisesti paristovarmennettu, sillä RAM-muisti hukkaa tiedot sähkön katketessa. Ohjelmoitavassa logiikkaohjaimessa on myös oma väyläkortti, joka takaa kommunikointi mahdollisuuden kenttäväylässä olevien laitteiden kanssa. Kenttäväylällä voidaan toteuttaa, esimerkiksi tehtaan sisälle I/O-hajautus ja olla yhteydessä muiden logiikkaohjaimien kanssa. Ohjelmoitavalle logiikkaohjaimelle voi myös asentaa mahdollisille lisätulosignaaleille ja -lähtösignaaleille omat lisäkorttinsa, jotta voidaan taata laajennusmahdollisuus. Ohjelmoitavaan logiikkaohjaimeen on sisäänrakennettu oma virtalähde, joka muuntaa käytettävän verkkovirran logiikkaohjaimelle sopivaksi. (Hagman, 2018, s. 11; Salila, 2015, ss. 7–10)

2.2 Historia

Ohjelmoitavan logiikan historia yltää 1948-vuoteen, jolloin keksittiin transistori, joka mullisti sähkötekniikan ja mahdollisti tehokkaamman sähkönohjauksen. Sähkötekniikan alalla kehitys oli varsin nopeaa transistorin keksittyä, sillä 1960-luvulla keksittiin prosessitietokoneet, jonka piireihin pystyttiin asentamaan useampia transistoreita. (Salila, 2015, s. 5)

1968-vuonna Dick Morley kehitti ensimmäisen ohjelmoitavan logiikan, joka käytti tikapuukaavio nimistä ohjelmointikieltä ja tämän nimeksi tuli alustavasti ”Standard Machine Controller”. Tämä luotiin General Motorsille, heille tullessa ongelmia autojen mallimuutoksien kanssa. General Motor halusi saada keinon, jolla voidaan ohjata koneita entistä tehokkaammin ilman suurempia muutoksia heidän tuotantolinjaansa. Ensimmäisellä ohjelmoitavalla logiikalla oli tarkka vaatimuslista minkä sen piti päästä läpi. Vaatimuksina oli muun muassa, ohjelmoitavassa logiikassa piti olla modulaarisia sekä laajennettavia puolijohdekomponentteja. Ohjelmoitavaan logiikkaan piti mahtua 32 tulosignaalia, joka olisi laajennettavissa 256:een, lähtösignaaleja piti mahtua 16, mutta pitäisi olla laajennettavissa 128:aan. Ohjelmoitavan logiikan pitäisi olla helppo ohjelmoida sekä uudelleen ohjelmoida. Ei saisi hukata ohjelmia sähkökatkoksen tullessa ja pitäisi säilyttää muistinsa 12 tunnin ajan. Ohjelmoitavassa logiikassa pitäisi olla minimissään 1 kB verran keskusmuistia, joka olisi laajennettavissa 4 kB:iin. Piti pystyä hallitsemaan samanaikaisesti kahdeksan eri funktiota 0.1 ja 10 sekunnin välillä. Lopuksi piti olla sisäänrakennettu eristys, joka kestää 120 VAC digitaalisignaalit sekä antaisi kuudelletoista lähtösignaaleille 120 VAC ja 4 ampeeria. Kuvassa 1 on kyseinen PLC ja tämä oli varsin tehokas aikaansa nähden, sillä tämä vähensi General Motorsille seisokkeja 60 %. Tämän ohjelmoitavan logiikan ollessa menestys, sille annettiin uusi nimi ”Modicon 084”, tämän ollen Modiconin 84. projekti. (Ball, 2015; Salila, 2015, ss. 5–6)

Kuva 1. An original Modicon PLC. Image used courtesy of Schneider Electric (Peterson, 2022)

Ohjelmoitavat logiikat ovat kehittyneet vuodesta 1968 jatkuvasti, sillä koko ajan on tullut jokin uusi tapa tehdä ohjelmoitavista logiikoista entistäkin tehokkaammiksi. Ohjelmoitavan logiikan historiassa muita merkittäviä keksintöjä ovat mikropiirit 1970-luvulla, mikroprosessorit 1980-luvulla, softPLC 1990-luvulla ja ohjelmoitavien logiikoiden hajautus mahdollisuus sekä avoimet järjestelmät tulivat 2000-luvulla. 1993-luvulla tuli ohjelmoitavalle logiikalle standardi IEC 61131-3, joka määrää standardeja nimeämiskäytännöille, ohjelmakululle ja muille ohjelmointimenetelmille sekä erilaisille tietotyypeille. Tämä IEC standardi loi myös viisi yhteensopivaa ohjelmointikieltä ohjelmoitaville logiikoille. Nämä kielet ovat instruction list (IL), sekvenssifunktiokaava (SFC), struktuuri teksti (ST), tikapuukaavio (LD) ja toimintolohkokaavio (FBD). (PLCopen, n.d.; Process Solutions, 2020; Salila, 2015, ss. 5–6)

2.3 PLC-ohjelmointi

PLC-ohjelmointi on tässä opinnäytetyössä hyvin isossa osassa, sillä ilman näitä perusoppeja tätä työtä ei voitaisi suorittaa. Tässä alaluvussa käydään läpi PLC-ohjelmoinnin perusteita ja vastataan seuraaviin kysymyksiin. Mikä on PLC-ohjelmointi ja miten siitä saa tehtyä mahdollisimman yksinkertaisen ja helposti luettavan? Mikä on kenttälaitteiden merkitys PLC-ohjelmoinnissa ja mitä eri kenttäväyliä on sekä niiden eroavaisuudet käytännössä? Lopuksi mikä on käyttöliittymä ja mistä koostuu hyvä käyttöliittymä?

2.3.1 Mitä on PLC-ohjelmointi?

Kaikki kohteet mitkä olisivat hyviä automatisoida ohjelmoitavalla logiikalla ovat sellaisia, joissa työ olisi joko hyvin toistuvaa tai tehokkuuden kannalta erittäin hidasta. Kuten tässä opinnäytetyössä todetaan, on jakaja-asema hyvä automatisoida ohjelmoitavalla logiikalla. Tässä työssä esimerkiksi ohjelmoidaan Siemensin TIA Portal -ohjelmointiohjelmistolla, mutta se ei ole ainut ohjelmisto, jolla voisi ohjelmoida ohjelmoitavat logiikat. Muita todella hyviä esimerkki ohjelmistoja ovat Beckhoff TwinCat 3, Omron ja Valmet. Näillä edellä mainituilla ohjelmistoilla ohjelmointikieli eroavat toisistaan vähäsen. (Salila, 2015, s. 5)

PLC-ohjelmointi on loogisten sekä matemaattisten funktioiden selvittämistä ja lopuksi rakentamista. Esimerkiksi jokin yksinkertainen säiliön sekoitin, jonka tehtävänä on sekoittaa nesteistä massaa, estäen jämähtämistä säiliön pohjalle. Tämä esimerkki voitaisiin toteuttaa seuraavalla tavalla. Nappi ”S1” käynnistää ja sammuttaa kyseisen prosessin. Kun S1 on painettu aktiivitilaan niin sekoittimen moottori ”M1” menee päälle tietyllä ehdolla. Tämä ehto voisi olla nesteen määrä kyseisessä säiliössä. Tässä esimerkissä hyvä ehto voisi olla minimissään 300 millilitraa nestettä säiliössä. Jos nestettä on alle 300 millilitraa, niin sitten moottori ”M1” sammuu tai sitten pysyy sammuneena mutta ”S1” on edelleen aktiivisena. Tästä yksinkertaisesta esimerkistä saadaan helposti luotua monimutkaisempi järjestelmä. Tämäkin järjestelmä saataisiin ohjelmoitavan logiikan avulla automatisoitua järkevästi, kunhan on tietyt ehdot selvillä. (Alvela, 2022, s. 15)

Kenttälaitteet ovat pakollisia, jos halutaan toteuttaa mahdollisimman tehokas automaatioratkaisu tehtaille. Kenttälaitteet yleisesti jakautuvat joko mittalaitteisiin tai toimilaitteisiin. Kenttälaitteet yhdistyvät kenttäväylän avulla ohjelmoitavan logiikkaohjaimen I/O-moduuleihin. Ohjelmoitava logiikka ohjaa kenttälaitteita sekä vastaanottaa tietoja, jotka voivat olla joko mitta- tai paikkatietoja. Mittalaitteilla voidaan mitata mitä tahansa haluttuja arvoja, esimerkiksi tilan lämpötilaa tai vesiputken virtausnopeutta. Myös voidaan kerätä tietoja kuljettimelta, esimerkiksi voidaan laskea kappaleiden lukumäärää tai voidaan mitata häiriötietoja. Toimilaitteita kuten sähkömoottoreita voidaan ohjata hyödyntämällä saatuja tietoja. Hyvänä esimerkkinä on kuljetinjärjestelmät, jotka kuljettavat kappaleita määränpäähän käyttämällä mitattuja tietoja. (Korva, 2019, s. 9)

2.3.2 Käyttöliittymä

Käyttöliittymä yhdistää ihmisen työskentelevän koneen kanssa ja mahdollistaa prosessin seuraamisen, käyttöönoton ja mahdollisten vikojen kartoituksen. Käyttöliittymä valmistetaan

yleisesti joko pelkälle näytölle tai erillisille tietokoneille. Yleisesti käyttöliittymä on joko visuaalinen tai tekstillinen. Hyvässä käyttöliittymässä käytetään niin visuaalisia symboleja kuin värejä, ja käytetään tekstiä minkä avulla ilmoitetaan vikatilanteet. Arkipäiväisiä käyttöliittymiä ovat, esimerkiksi tietokoneen hiiri, auton ratti tai television kaukosäädin. (Churchville, 2021)

Kun ohjelmoidaan logiikkaohjainta ja on tarve käyttöliittymälle, niin se täytyy tehdä miettimällä laitteen käyttäjän näkökulmaa. Käyttöliittymän täytyy olla selkeä, helppo ymmärtää, tarkoituksellinen ja visuaalisesti seurattava. Käyttäjän on helpompi ymmärtää, jos käyttöliittymä pidetään selkeänä hyödyntämällä symboliikkaa tai muita visuaalisia merkkejä. Esimerkiksi jos käyttäjä haluaa seurata vesiputken venttiilin toimintaa, niin olisi hyvä suunnitella käyttöliittymään visuaalinen venttiili. Tästä esimerkkiventtiilistä käyttäjä voi seurata sen toimintaa ja tehdä muutoksia aina tarvittaessa. (A. Virtanen, 2012, ss. 14–15)

Käyttöliittymässä on hyvä käyttää värejä. Värien avulla käyttäjä voi helposti seurata, esimerkiksi robottien tai kuljettimien toimintaa. Tässäkin on tärkeä muistaa, ettei käytetä liikaa värejä. Liialliset värit voivat aiheuttaa enemmän hämmennystä kuin hyötyä. Hyvät värit, joita käytetään yleensä ovat vihreä, punainen, keltainen ja harmaa. (Churchville, 2021; A. Virtanen, 2012, ss. 14–15)

3 Siemens

Tässä luvussa käydään läpi Siemensin historiaa ja sen merkittävimpiä keksintöjä, jotka muuttivat tulevaisuuden teknologian suuntaa. Käydään myös läpi Simatic tuotesarjaa sekä ”SIMIT” nimistä ohjelmistoa. SIMIT-ohjelmistosta käydään läpi perusteita sekä ominaisuuksia mitä SIMIT tarjoaa käyttäjälle.

3.1 Historia

Siemens aloitti 1958-luvulla valmistamaan Simaticin tuotesarjaa hyödyntämällä transistoreita ja niiden tuomia etuja verrattuna releisiin ja kontaktoreihin. Ensimmäiset Simaticin tuotteet olivat ohjauspiirejä, joihin oli asennettu germanium-laatalla integroituja transistoreita, nämä olivat nimeltään Simatic G. Simatic G oli suuressa suosiossa 1958-luvusta eteenpäin teollisuuden automatisoinnissa, sillä se toi lisää tehokkuutta tehtaisiin, mitä kaivattiin 1958-luvusta eteenpäin. Siemens jatkoi Simaticin tuotekehittelyä, joka tuotti tuloksia 1964-luvulla, kun Siemens vaihtoi transistorit germanium-laatoista silikonisiin malleihin. Tämän mallisarjan

nimeksi tuli Simatic N. Simaticin suuren suosion ansiota Siemensistä tuli yksi suurimpia markkinoijia automaatioteknologian tuotteissa. Myöhemmin 1979-luvulla Siemens valmisti Siemens S5 ohjelmoitavan logiikkaohjaimen, jota ei enää ohjelmoitu johdottamalla vaan ohjelmoitiin ”Step 5” nimisellä ohjelmistolla. Step 5 -ohjelmisto takasi käyttäjälle helposti ymmärrettävän ohjelmointiympäristön. (Siemens, 2018, 2022a)

Positiivisia keksintöjä mitä Siemens on luonut historiassa ovat pointer telegraph 1847-luvulla, dynamo 1866-luvulla, raitiovaunun suunnittelu ja toteutus 1881-luvulla. Siemens valmisti myös ensimmäisen röntgenlaitteen 1896-luvulla Wilhelm Röntgenin avulla. Siemens loi rahoituksen omien työntekijöiden terveyden ja kuolin tapauksille vuonna 1872. Tämän lisäksi Siemens toi työterveyden heidän tehtaihinsa lääkäreiden avulla 1888-luvulla. Siemens keksi ensimmäisen röntgenputken ”Te Pantix” 1933-luvulla, keksi ensimmäisen gammakameran 1958-luvulla, keksi ultraäänilaitteen 1967-luvulla, ensimmäisen CT-kuvauslaitteen ”SOMATOM Plus-S” 1990-luvulla. Siemensin viimeisin keksintö oli magneettikuvauksen mahdollistaminen 2010-luvulla. (Siemens, 2022a, 2022b, 2022d, 2022e, 2022f, 2022g; Siemens-healthineers.com, n.d.)

3.2 Simatic

Siemens Automatic eli Simatic on Siemensin luoma tuotesarja. Tämän avulla Siemens on tehnyt lukuisia erilaisia ratkaisuja asiakkaiden tarpeille ja tuonut esille automatisoinnin tehokkuuden. Simatic on laaja sekä yksi suurimmista automaatiotuoteperhe markkinoijista. Simaticin tuotteet soveltuvat Siemensin omaan TIA Portal -ohjelmointiohjelmistoon. Tämä takaa asiakkaalle suuria säästöjä niin ajassa kuin tarvittavissa resursseissa suunnittelussa sekä käyttöönotossa. Tässä alaluvussa käydään Simaticin kahta suurinta ohjelmoitavien logiikkaohjainten tuotesarjoja. Nämä tuotesarjat ovat Siemens S5 ja Siemens S7. Tässä alaluvussa myös käydään läpi niille tarkoitetut ohjelmistot. (Siemens, n.d.-a)

3.2.1 Simatic S5

Simatic S5 on suosittu Siemensin tuoteperhe, joka sai alkunsa 1979-luvulla. Simatic S5 tuoteperheeseen kuuluu muun muassa S5-90U, -95U, -100U, -101U, -115U, -135U sekä -155U. Simatic S5 tuoteperheen tukeminen varaosilla sekä korjauksilla lopetettiin ensimmäinen päivä lokakuuta 2020-luvulla. Syy miksi S5 tuoteperheen tukeminen jatkui niin pitkään, johtuu sen saadusta suosiosta. Suosio näkyy S5 tuoteperheen tuotteiden käyttäjien lukumäärässä. (Kytölä, 2023, ss. 19–20; Paasikivi, 2018, s. 7)

Step 5 julkaistiin 1979-luvulla Simatic S5 yhteydessä, jota käytettiin kyseisten ohjelmoitavien logiikkaohjainten ohjelmointiin. Step 5 -ohjelmointiympäristö oli alkuperäisesti CP/M-käyttöjärjestelmässä, mutta 1996-luvulla Siemens julkaisi Step 5:seen version 6.5, joka muutti CP/M-käyttöjärjestelmän MS-DOS-käyttöjärjestelmään. Step 5 ohjelmointikieliä ovat käskylista, tikapuukaavio ja toimilohko. Isoimmat erovaisuudet Step 5:ssä ja Step 7:ssa ovat valmiissa kirjastoissa ja toimilohkoissa. Esimerkiksi Step 5:ssä on erilainen ajastin (SD-ajastin), joka mittaa aikaa 16-bitillä. Step 7:ssa aikaa mitataan sekunneissa, millisekunneissa tai millä tahansa ajan yksiköllä, jonka käyttäjä määrittää. Esimerkiksi Step 7:ssa sekunnit olisivat T#15s, eli 15 sekuntia. (Leinonen, 2020, ss. 24–25)

3.2.2 Simatic S7

Simatic S7 on toinen äärimmäisen suosittu tuoteperhe Siemensillä. Simatic S7 valmistus aloitettiin vuonna 1994. Simatic S7 eroaa S5 sarjasta jo heti muistissa, johon ei tarvita erillistä paristovarmennusta. S7:ssa on myös suurempi muistin määrä ja tehokkuus on parempi. Simatic S7 tuotesarjoja ovat muun muassa S7-300, -400, -1200 sekä -1500, ja I/O moduuli ET 200SP tuotesarja. ET 200SP mahdollistaa suuremmankin hajautuksen tehtaissa. (Pirttijoki, 2023)

Step 7 ilmestyi samaan aikaan, kuin Simaticin S7 tuotesarja ja Step 7:aa käytetään tuotesarjojen 300 ja 400 logiikkaohjainten ohjelmointiin. Tätä vanhempaa Step 7 -ohjelmointityökalua kutsutaan Step 7 Classic:ksi, joista tarkoitetaan Step 7 vanhempia versioita, eli V5.6 ja siitä vanhempia. Step 7 on standardin IEC 611-31-3 mukainen. Step 7 eroaa suuresti Step 5:stä. Esimerkiksi Step 7 on luotu Windowsin käyttöjärjestelmälle MS-DOS-käyttöjärjestelmän sijaan. Step 7:ssa on myös uusi ohjelmointikieli nimeltään Structured Text (ST). Step 7:ssa on myös mahdollista seurata sekä testata omaa logiikkaa simulaatio tyylillä ohjelmointivaiheessa. Tämän avulla säästetään runsaasti aikaa ja mahdollisilta harmeilta mitä voisi tulla käyttöönotossa. (Leinonen, 2020, ss. 25–26; Paasikivi, 2018, ss. 16–17)

3.3 Siemens SIMIT

Siemens julkaisi ”Simulation & Testing”, eli SIMIT:in ensimmäisen asiakkaiden käyttöön kuuluvan version V4.0 2000-luvulla, mutta aloitti jo SIMIT:in kehityksen 1990-luvulla. Tuolloin tarkoituksena oli kehittää sisäistä voimalaitossimulaatiota, jota kutsuttiin ”Projektierungssimulator” (suom. Projektisuunnittelun simulointi). SIMIT:in ensimmäisillä

versioilla pystyttiin simuloimaan pelkästään voimalaitosprosesseja, mutta SIMIT:in simulointimahdollisuudet laajennettiin versiossa 7.0 jossa mahdollistettiin automaatiojärjestelmien simulointi. Tällä hetkellä uusin versio SIMIT:stä on V11.1 joka on julkaistu 25.7.2023. (Hirvonen, 2021, s. 9; Siemens, 2023a; M. Virtanen, 2021, s. 8)

SIMIT on mallinnusohjelmisto, jota käytetään yleisesti kaiken prosessien, toimilaitteiden sekä turvalaitteiden simulointiin ja testaukseen projektien varhaisessa vaiheessa. SIMIT:llä voidaan myös kouluttaa työntekijöitä virtuaalisessa ympäristössä, joka vastaa realistisista prosessia jo heti prosessin rakennusvaiheessa. SIMIT:in avulla voidaan luoda halutusta prosessista digitaalisen kaksosen, eli virtuaalisen version kyseisestä prosessista. Virtuaalimallia voitaisiin käyttää esimerkiksi koulutuksessa. Tämä vähentää kustannuksia yritykselle, parantaa prosessin elinikää, pienentää riskiä tapaturmille ja säästää aikaa käyttöönotossa sekä mahdollisilta laitosseisokeilta. (Hirvonen, 2021, ss. 11–12; Siemens, 2023b; M. Virtanen, 2021, s. 8)

4 Työssä käytettävät tarvikkeet

Tässä luvussa käydään läpi kaikki tarvikkeet, mitä tämä työ tarvitsi. Tässä työssä käytettiin Feston MPS-alustan ”Sorting station” eli jakaja-aseman valmiiksi rakennettua esimerkkikohdetta. Kaikki ohjelmistot mitä käytettiin ovat Siemens TIA Portal, PLCSIM Advanced, SIMIT SP ja SIMIT CTE.

4.1 Festo MPS-alusta ”Sorting station”

Kuvassa 2 nähdään tässä työssä käytetty työkohde. Työkohteelle tulee kolme eri kappaletta, joista on kahta eri materiaalista valmistettua, jotka ovat kolmea eri väriä. Nämä tulevat moottoroidulta kuljettimelta, jotka pysähtyvät ensimmäiselle stopparille, jolloin anturit tarkastavat kappaleen materiaalin sekä värin. Tämän jälkeen ensimmäinen stoppari avautuu ja päästää kappaleen liikkeelle. Kappaleen ollessa liikkeellä joko ensimmäinen tai toinen moottoroitu kääntäjä, tai kuljettimen päädyssä oleva integroitu kääntäjä ohjaa kappaleen pois kuljettimelta. Samanaikaisesti stoppari menee kiinni kappaleen liikuttuaan stopparin ohi. Kuvassa 2 nähdään olevan kolme alumiinista tehtyä rataa, joihin menevät kappaleet, jotka jaetaan valmistetun materiaalin sekä värin mukaan. Esimerkiksi tätä jakaja-asemaa voitaisiin käyttää suuremmassa tuotannon tehtaassa ja tämän tehtävänä olisi jakaa tuotteita asiakkaiden mukaan. Tästä jakaja-asemasta otettiin fyysiset I/O:t talteen, jotka nähdään

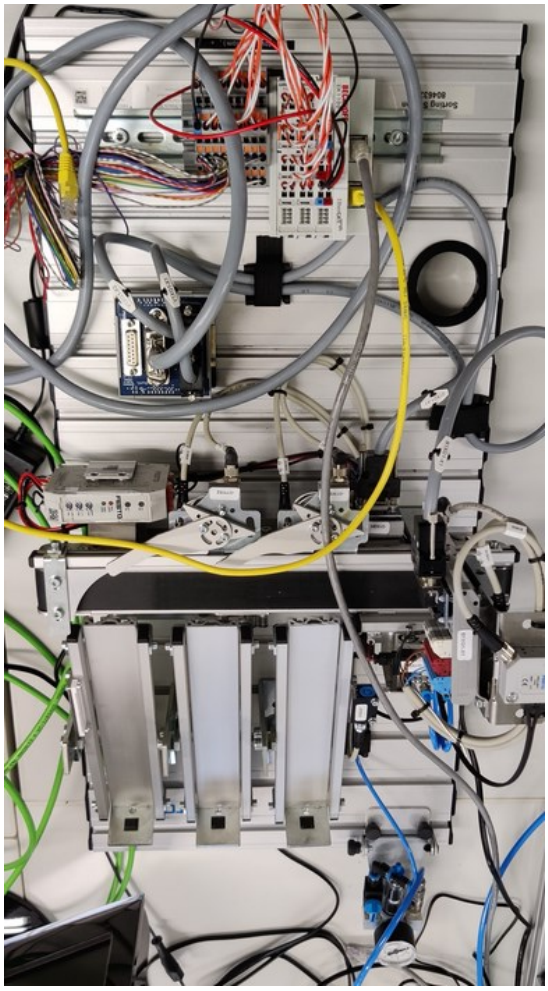
taulukossa 1. Taulukossa 1 nähdään jokaisen tulojen ja lähtöjen vieressä olevan oma TIA Portaalin linkitys. Nämä linkitykset nähdään myös kuvassa 5.

Taulukko 1. Fyysisen jakaja-aseman I/O:t ja niiden linkitys TIA Portaalissa (Festo, n.d.)

I/O	Fyysisen jakaja-aseman tulot	TIA Portaalissa nimetty	I/O
I0.0	Workpiece at beginning of conveyor	Kappale kuljettimen alussa	I0.0
I0.1	Deflector 1 extended	Kääntäjä1 kiinni	I0.1
I0.2	Slide full	Rata täynnä	I0.2
I0.3	Deflector 2 extended	Kääntäjä 2 kiinni	I0.3
I0.4	Workpiece detected	Kappale tunnistettu	I0.4
I0.5	Workpiece not black	Kappale ei ole musta	I0.5
I0.6	Workpiece metallic	Kappale on metallinen	I0.6

I/O	Fyysisen jakaja-aseman lähdöt	TIA Portaalissa nimetty	I/O
Q0.0	Conveyor forward	Kuljetin Päällä	Q0.0
Q0.1	Extend deflector 1	Kääntäjä 1 päälle	Q0.1
Q0.2	Extend deflector 2	Kääntäjä 2 päälle	Q0.2
Q0.3	Retract stopper	Stoppari auki	Q0.3

Kuva 2. Festo MPS-alusta "Sorting station"



4.2 Työssä käytettävät ohjelmat

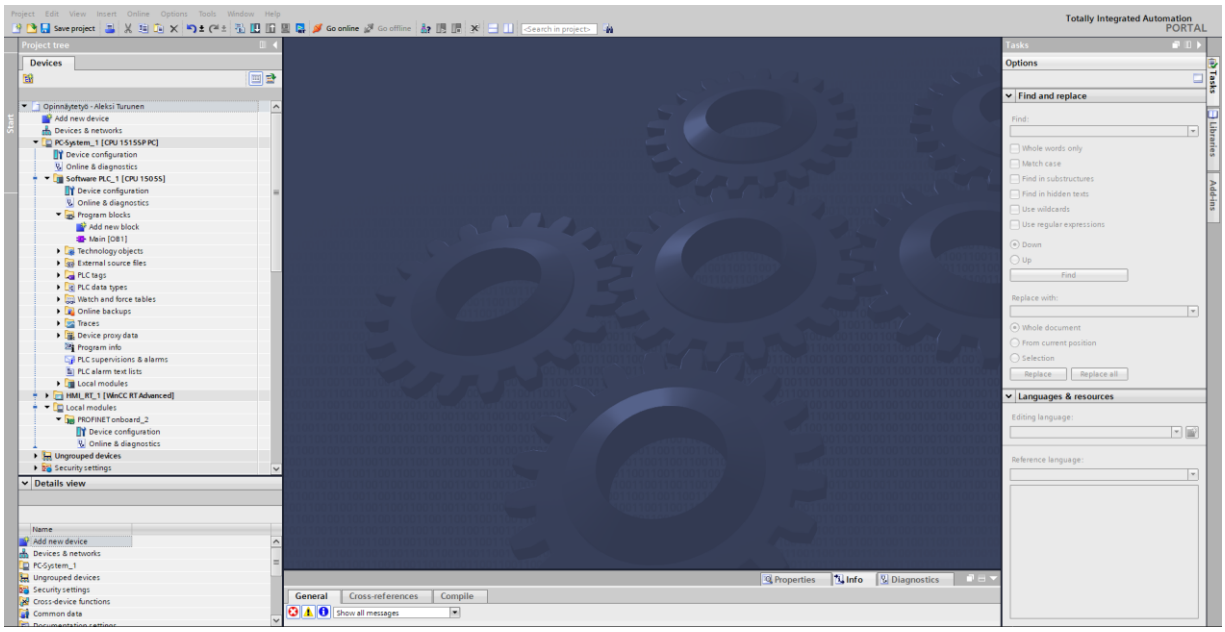
Tässä luvussa käydään läpi mitä ohjelmistoja tarvitaan tämän työn suorittamiseen. Ohjelmistot ovat Siemens TIA Portal, PLCSIM Advanced, SIMIT SP ja SIMIT CTE. Käymme myös läpi, miten nämä ohjelmistot integroituvat toisiinsa.

4.2.1 Siemens TIA Portal

Siemensin “Totally integrated Automation Portal” eli TIA Portal on ohjelmisto, jolla ohjelmoidaan pääsääntöisesti logiikkaa. TIA Portaalilla voi myös suunnitella käyttöliittymän ja toteuttaa erilaisia turvaratkaisuja tarvittaviin kohteisiin. Logiikkaa on mahdollista ohjelmoida kaikilla IEC-standardin kielillä samanaikaisesti projektin sisällä. Nämä ohjelmointikielet ovat tikapuukaavio (LAD), toimilohkokaavio (FBD), strukturoitu ohjelmointiteksti (SCL), statement

list (STL) ja S7-GRAPH. Siemens pyrki toteuttamaan mahdollisimman käyttäjäystävällisen ohjelmiston ja sen näkee verrattuna Siemensin edellisiin ohjelmistoihin. Lisäksi TIA Portaaliin on yhdistetty Siemensin omia ohjelmistoja, joilla on omat käyttötarkoitukset. SIMATIC STEP 7 käytetään logiikan ohjelmointiin. SIMATIC STEP 7 Safety Advancedia käytetään turvalogiikan ohjelmointiin. SIMATIC WinCC:tä käytetään käyttöliittymien suunnitteluun. SIMATIC Startdrivea käytetään taajuusmuuttajien ohjelmointiin. Uusin TIA Portal versio on V19, joka julkaistiin vuonna 2023. Tämä versio toi uusia tapoja luoda käyttöönottoja vanhoille sekä uusille ratkaisuille. Tässä opinnäytetyössä käytettiin TIA Portal V16 versiota, joka toimii moitteettomasti. Kuvassa 3 nähdään kyseisen version oletusnäkymä projektin luotua. (Ervalahti, 2018, s. 13; Ruha, 2019, ss. 12–13; Siemens, 2023c)

Kuva 3. TIA Portal oletusnäkymä



Siemens TIA Portaallissa konfiguroidaan kaikki tarvittava laitteisto aina uuden projektin luonnissa. Laitteistolla tarkoitetaan keskusprosessointiyksikköä (CPU) tulosignaalikortit (DI tai AI), lähtösignaalikortit (DQ tai AQ) ja mahdolliset sähkömoottori lähdöt sekä virtalähteet. (Leppälä, 2023, s. 21)

4.2.2 PLCSIM Advanced

Siemensin TIA Portal -ohjelmointityökalun rinnalle Siemens on valmistanut PLCSIM Advanced simulaatiotyökalun. Tämän simulaatiotyökalun avulla voidaan virtuaalisesti emuloida Siemensin keskusprosessointiyksiköitä kuten, SIMATIC S7-1500 sarjaa tai I/O

modeemi ET 200SP. PLCSIM Advanced mahdollistaa TIA Portaalissa ohjelmoidun logiikan lataamisen emuloidulle laitteistolle määrätylle instanssille. PLCSIM Advanced antaa myös käyttäjän seurata tarkasti simulaatiota niin tuloissa kuin lähdöissä ja mukautuu aina tarvittaessa muuttujien mukaisesti. (Siemens, 2021, ss. 16–17)

4.2.3 Siemens SIMIT SP

SIMIT SP on lyhenne SIMIT Simulation Platform:sta, joka on Siemensin valmistama simulointi alusta. SIMIT SP mahdollistaa tässä opinnäytetyössä TIA Portaali projektin mallinnuksen. Tätä ohjelmistoa käytetään luomaan virtuaalisen kaksosen työn kohteesta. SIMIT SP integroituu TIA Portaalin kanssa käyttämällä PLCSIM Advancedia. Kun SIMIT SP integroidaan TIA Portaalin kanssa, se saa käyttöönsä TIA Portaalissa käytetyt muuttujat, jotka ovat I/O-listauksessa. (Siemens, 2023b)

4.2.4 SIMIT CTE

SIMIT CTE on lyhenne SIMIT Component Type Editor:sta, joka antaa käyttäjälle täyden mahdollisuuden luoda omia komponentteja. Näitä luotuja komponentteja käyttäjä voi ottaa käyttöön omissa töissä. SIMIT CTE antaa käyttäjälle täyden mahdollisuuden luoda minkälaisen komponentin vain. SIMIT CTE:ssä käyttäjä voi asettaa komponentille yhdistin paikat, alkuehdot, matemaattiset funktiot, käyttäjä voi tallentaa erilaisia tietoja komponentille, myös asettaa toistuvia funktioita, komponentille oman topologian. Näistä käyttäjä mallintaa oman komponentin. SIMIT CTE toimii moitteetta muiden SIMIT ohjelmistojen kanssa. (Siemens, 2023b, ss. 367–433)

4.3 Koulun ohjausjärjestelmä

Koulun ohjausjärjestelmä mitä käytettiin opinnäytetyön toteuttamisessa nähdään kuvassa 4. Tämä on varsin erinomainen ohjausjärjestelmä siihen nähden, mitä tämä työ tarvitsi niin laskentateholtaan kuin lähtö- sekä tulosignaalien lukumääriltään. Tämä ohjausjärjestelmä pitää sisällään:

- ET200SP hajautettu I/O-järjestelmä
- 1515SP PC prosessorin (677-2AA40-0AA0)
- 2x DI 8x24 VDC HF (131-6BF00-0CA0)
- 2x DO 8xVDC/0.5A HF (132-6BF00-0CA0)

- Server module (193-6PA00-0AA0)
- Kosketusnäyttö on KTP700 Basic (123-2GB03-0AX0)
- Virtalähde on 24VDC 5A (DNX12AS24)

Ainoa ongelmatilanne tämän ohjausjärjestelmän kanssa tuli käyttöliittymän luonnissa. Kyseisen ohjausjärjestelmän näyttö ei tukenut käyttöliittymän suunnittelussa haluttuja ominaisuuksia kuten pop-up ikkunoita, tai funktio näppäimiä, jotka näkyvät kuvassa 4.

Kuva 4. Koulun PLC

5 TIA Portal -ohjelmoinnin toteutus virtuaalimallille

Tässä luvussa käymme läpi ohjelmointisuunnittelua TIA Portal -ohjelmointiympäristössä, käymme tavoitteet läpi ja kerrotaan, miten päästiin tavoitteisiin. Tämän projektiosuuden tavoitteina oli luoda toimiva kokonaisuus. Tähän kuuluu erikoistilanteet, manuaaliohjaus jokaiselle lähtömuuttujille, koodin selkeys, fyysisen I/O-rajapinnan pitäminen (kuva 5) sekä selkeä ja riittävä dokumentaatio.

Tämän TIA Portaalin ohjelmoinnin tarkoituksena on ohjata automaattisesti SIMIT SP:ssä mallinnettua jakaja-asemaa, SIMIT SP:stä saatujen anturitietojen avulla. Tähän ohjaukseen kuuluu häiriötilanteet, jakaja-aseman sekvenssimäisen ohjauksen ajaminen ja kappaleiden laskenta stopparilla.

Kuva 5. Fyysinen I/O-rajapinta

I/O

	Name	Data type	Address	Retain	Acces...	Writa...	Visibl...
1	Kappale kuljettimen alussa	Bool	%I0.0	☐	☑	☑	☑
2	Kääntäjä1 kiinni	Bool	%I0.1	☐	☑	☑	☑
3	Rata täynnä	Bool	%I0.2	☐	☑	☑	☑
4	Kääntäjä 2 kiinni	Bool	%I0.3	☐	☑	☑	☑
5	Kappale tunnistettu	Bool	%I0.4	☐	☑	☑	☑
6	Kappale ei ole musta	Bool	%I0.5	☐	☑	☑	☑
7	Kappale on metallinen	Bool	%I0.6	☐	☑	☑	☑
8	Kuljetin Päällä	Bool	%Q0.0	☐	☑	☑	☑
9	Kääntäjä 1 päälle	Bool	%Q0.1	☐	☑	☑	☑
10	Kääntäjä 2 päälle	Bool	%Q0.2	☐	☑	☑	☑
11	Stoppari auki	Bool	%Q0.3	☐	☑	☑	☑

5.1 Virtuaalisen jakaja-aseman toiminnankuvaus

Virtuaalisen jakaja-aseman automaattiohjaus alkaa siitä, kun käyttäjä painaa virtuaalimallin käyttöliittymästä nappia ”Black object”, ”Metallic object” tai ”Pink object”. Tämä asettaa kuljettimen alkupäässä sijaitsevan paikka-anturin %I0.0 eteen kyseisen kappaleen. Kun paikka-anturi %I0.0 menee asentoon ”1”, tämä ohjaa kuljettimen moottorin %Q0.0 asentoon ”1” ja kuljettaa kappaleen stopparille. Kappale on edennyt stopparille, kun valoverho %I0.4 on ollut asennossa ”1” yhden sekunnin ajan verran. Kun valoverho %I0.4 on ollut asennossa ”1” yhden sekunnin ajan, se asettaa kuljettimen moottorin %Q0.0 asentoon ”0”. Stopparilla määritetään kappaleen materiaali induktiivisesta lähestymisanturista %I0.6 ja väri diffuusioanturista %I0.5.

Kun kappale on määritetty stopparilla, niin stoppari %Q0.3, kuljettimen moottori %Q0.0 ja mahdollinen kääntäjä %Q0.1 tai %Q0.2 menevät asentoon ”1”. Se mikä kääntäjä menee asentoon ”1”, riippuu täysin tunnistetusta kappaleesta. Vaaleanpunaisen kappaleen radan kääntäjän lähtösignaali on %Q0.1 ja tulosignaali on %I0.1. Mustan kappaleen radan kääntäjän lähtösignaali on %Q0.2 ja tulosignaali on %I0.3. Jos käyttäjä simuloi metallisen kappaleen toimintaa, silloin menisi vain stoppari %Q0.3 ja kuljettimen moottori %Q0.0 asentoon ”1”. Kun kappale on liikkunut 400ms ajan verran, stoppari %Q0.3 menee asentoon ”0”. Tästä eteenpäin kappale etenee omalle alumiiniselle radalle. Kun kappale on kääntynyt

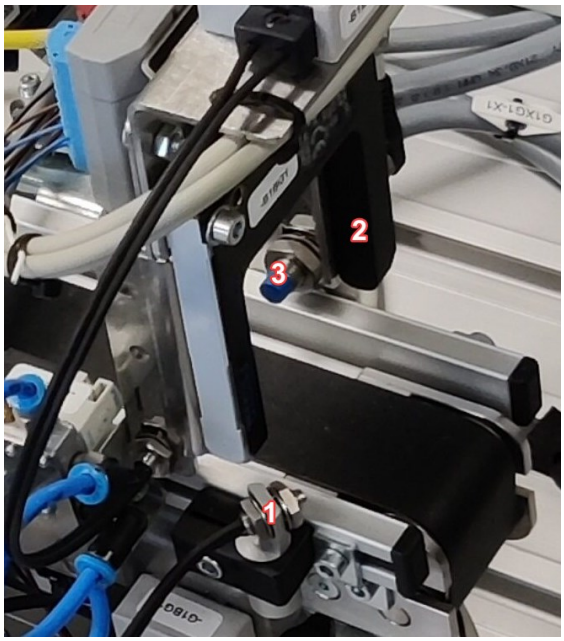
omalle alumiiniselle radalle, se asettaa kyseisen kääntäjän ja kuljettimen moottorin %Q0.0 asentoon ”0”. Samalla alumiiniradalla oleva paikka-anturi %I0.2 menee asentoon ”1”, kun kappale on paikka-anturin kohdalla. Paikka-anturi %I0.2 on alumiiniradalla neljännen kappaleen kohdalla, joka tarkoittaa neljännen kappaleen olevan kyseisen anturin edessä. Normaalin kappaleen toiminnan simulointi päättyy tähän.

Huomioitavia asioita ovat seuraavanlaiset. Stopparin molemmilla puolilla voi aina olla maksimissaan yksi kappale kerrallaan. Käyttäjä ei pysty simuloimaan normaalin tai vikatilanteita, missä on kääntäjävika silloin, kun kyseisen kappaleen alumiinirata on täynnä (5x kappaletta). Käyttäjä pystyy simuloimaan tunnistamattoman kappaleen toimintaa silloin, kun stopparia edeltävä kuljetin on tyhjä. Häiriötila asettaa virtuaalimallin lähtömuuttujat asentoon ”0”. Käyttäjä voi simuloida häiriötilanteita sekaisin normaalin toiminnan kanssa, kunhan yllä mainitut asiat toteutuvat. Virtuaalimalli kuittaa hälytykset SIMIT SP:ssä automaattisesti 5 sekunnin kuluessa, mutta käyttäjä joutuu itse kuitata hälytykset TIA Portaalin käyttöliittymästä. Jos käyttäjä simuloi häiriötilannetta ja kappaleen normaalia toimintaa, häiriötilanne lopettaa kappaleen normaalin toiminnan suorittamisen, kun häiriötila menee asentoon ”1”.

5.2 Kappalelaskurien suunnittelu

SCL-kieli (Strukturoitu ohjelmointikieli) soveltuu täydellisesti kappalelaskurien suunnitteluun, sen ollessa hyvin yksinkertainen ymmärtää sekä suoraviivainen ohjelmointikieli. Työssä käytettävät kappaleet jakautuivat neljään eri kategoriaan. Nämä kategoriat ovat, muoviset mustan väriset sekä vaaleanpunaisen väriset, metalliset ja tunnistamattomat kappaleet. Kuvassa 6 havaitaan anturien olevan eri kohdissa radan alkupäässä. Järjestyksessä ensimmäinen anturi on paikka-anturi, joka mittaa onko kuljettimelle tullut kappale. Toinen anturi on valokenno, joka mittaa onko kappale tunnistettu ennen stopparia. Kolmas anturi on diffuusi anturi, joka tunnistaa kappaleen materiaalin sekä onko kappale muun värinen kuin musta. Kun nämä anturit ovat näin hajanaisia, se tuo ongelman vaaleanpunaisen värisen kappaleen tunnistamisessa. Tämän ongelman esimerkkiratkaisu voisi olla liitteen 1 mukainen. Liitteessä 1 havaitaan TON-ajastin ”Laskurin_Ajastin”, joka viivästyttää vaaleanpunaisen väristen kappaleiden tunnistusta 350 millisekunnilla. Tämän TON-ajastimen resettinä toimii työn sekvenssiohjelman stepit 9, 10 ja 11, ja ratojen omat kuittausnapit, jotka mahdollistavat ajastimelle automaattisen toistavuuden.

Kuva 6. Jakaja-aseman alkupää



Liitteessä 2 nähdään esimerkkitoteutus siitä, miten ohjelma laskee mustan väriset kappaleet. Tätä esimerkkiä toistettiin pienillä tulomuuttujien muutoksilla vaaleanpunaisten väristen kappaleiden sekä metallisten kappaleiden laskemiseen. Laskuri laskee kappaleiden lukumäärää, kun halutut muuttujat menevät päälle. Tässä esimerkissä muuttujat, jotka täytyvät olla päällä ovat ”Musta_Rata_Ok”, joka seuraa mustan väristen kappaleiden radan tilavuutta, ”Laskurin_Ajastin.Q” ja ”Kappale tunnistettu”, jotka havaitaan jo liitteestä 1. Muuttujat, jotka täytyy olla pois päältä ovat ”Kappale ei ole musta” ja ”Kappale on metallinen”. Tämän laskurin toimintaa varten luotiin välimuuttuja ”Musta_Kappale”, joka menee päälle, kun laskuri on laskenut ensimmäisen mustan värisen kappaleen. Tämän välimuuttujan avulla varmistetaan, ettei stoppari päästä kappaletta liian ajoissa läpi, vaan annetaan laskurin ensimmäiseksi laskea kappale.

Liitteestä 2 nähdään myös ”Musta_Rata_Täynnä_Ajastin”, joka on ajastin, joka viivyttää lähtömuuttujaa ”Musta_Rata_täynnä”. Tämä ajastin menee päälle, kun laskuri on laskenut viisi mustaa kappaletta, jolloin tämä ajastin kertoo kyseisen radan olevan täynnä. Jotta tämän ajastimen sekä laskurin saa resetoitua täytyy painaa ”Musta_Rata_Kuittaus” nappia, joka kuittaa kyseisen radan kerätyksi.

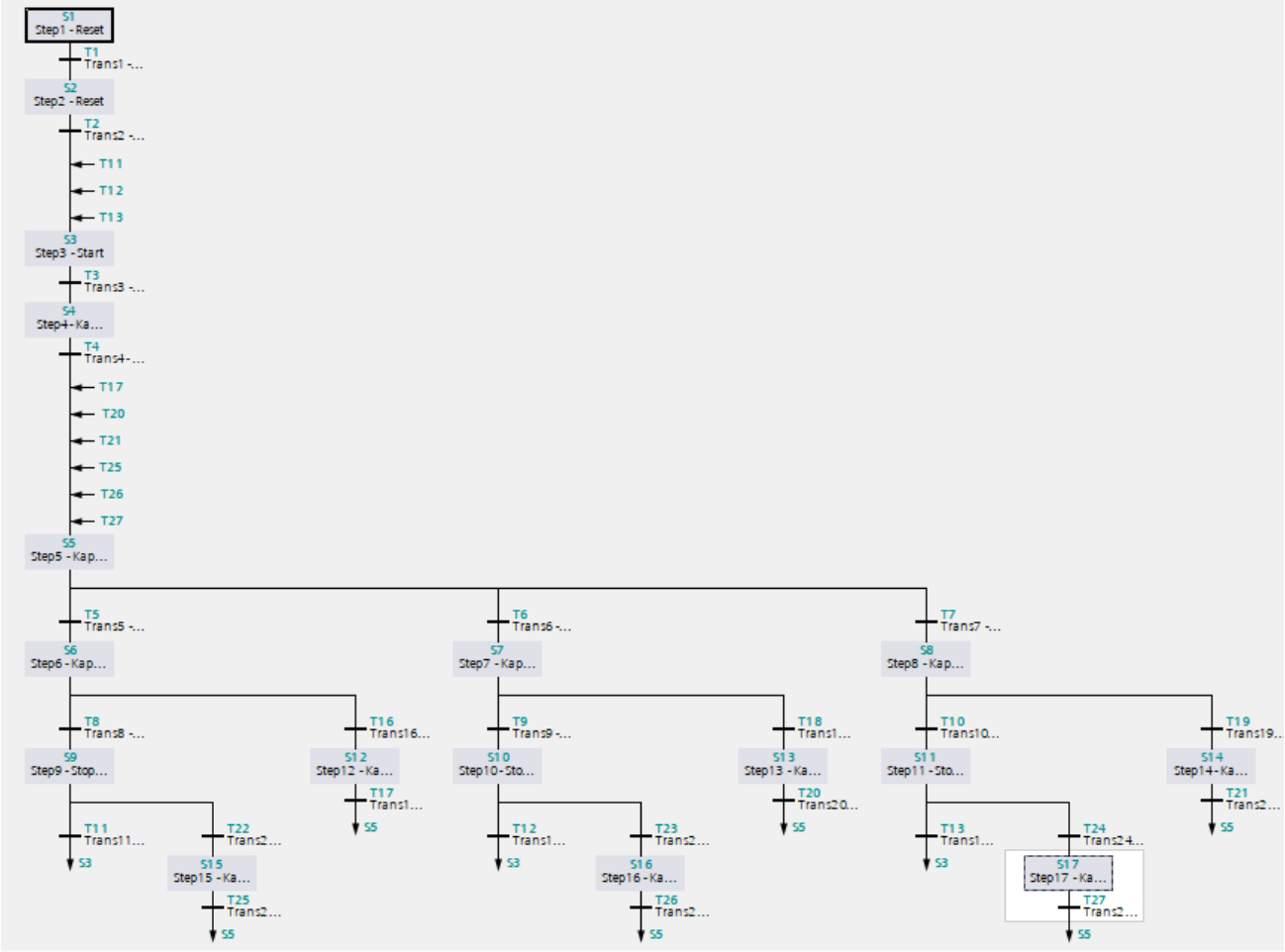
Suurimmat ongelmat tässä funktio blokissa ilmenivät ajastimien sekä laskurien toiminnoissa ja automaatiosekvenssin sovituksessa. Näiden ongelmien selvitys meni hyvin, koska tämä ohjelmointikieli on helppo ymmärtää.

5.3 Sekvenssiohjelman suunnittelu

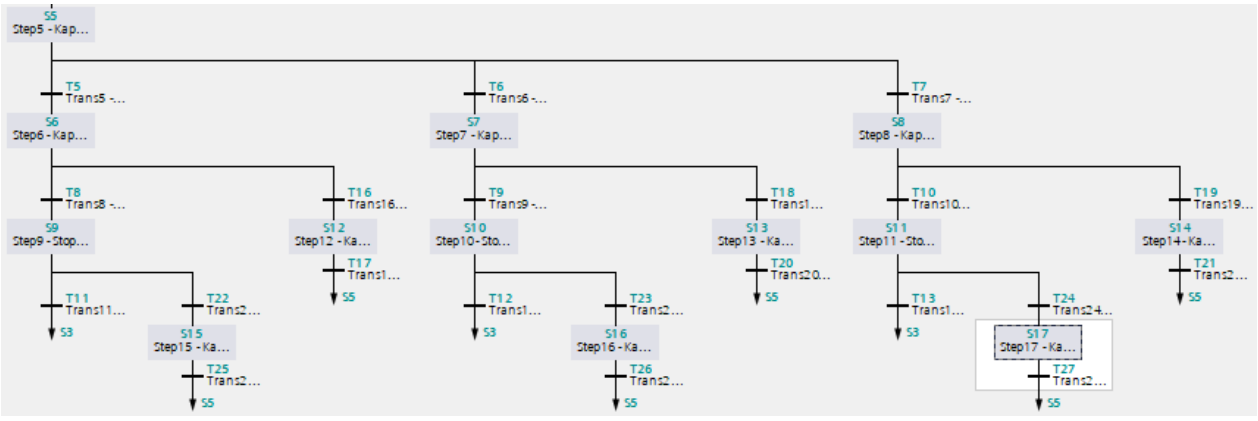
Koulun TIA Portal V16:ssa on yksi ohjelmointikieli, joka soveltuu sekvenssiohjelman ohjaukseen äärimmäisen hyvin ja se on S7-GRAPH-kieli. Kuvassa 7 nähdään esimerkkitoteutus tälle sekvenssille. Tämä ohjelmointikieli on äärimmäisen helppo oppia, sillä tämä ohjelmointikieli antaa käyttäjälle mahdollisuuden seurata täydellisesti koodin etenemisprosessia steppi kerrallaan. Kun katsotaan kuvaa 7, niin havaitaan kolmannen stepin olevan ”Step3 – Start”. Tämä tarkoittaa sekvenssin aloitusta, ja tämä on se steppi mistä lähdetään liikkeelle heti ensimmäisen sekvenssi kierroksen jälkeen. Stepit 1 ja 2 ovat lähinnä häiriötietojen ja niiden ajastimien resetointia varten. Kuvassa 8 nähdään viidennessä stepissä olevan kolme erilaista siirtymää, joista lähtee useita siirtymiä stepeille, joilla on omat tarkoituksensa. Näillä siirtymillä mahdollistetaan virtuaalimallin nopeampi sekä itsenäisempi toiminta. Aina on mahdollista useamman kappaleen joutuvan kyseiselle kuljettimelle joko peräkkäin tai jonkin ajan viiveellä. Tämän takia täytyy olla useampi vaihtoehto siitä, miten sekvenssi etenee. Esimerkiksi vaaleanpunaisen kappaleen stepeissä ”Step9”, ”Step12” ja ”Step15” eroavat toisistaan siitä, miten kuvassa 6 olevan toinen anturi tunnistaa kappaleen. Jos anturi tunnistaa kappaleen joko heti tai viiveellä, niin sitten siirrytään suoraan viidenteen steppiin siirtymiltä ”T17” ja ”T25”. Jos ei tunnista niin sitten edetään kolmanteen steppiin siirtymästä ”T11”.

Kuvassa 8 nähdään, miten viidennestä stepistä lähtee kolme erilaista siirtymää omille stepeille, koska viides steppi on kappale tunnistusta varten. Kuvassa 9 nähdään, miten viidennessä stepissä tunnistetaan mustan väriset kappaleet ja sieltä nähdään tutut muuttujat, jotka ovat pääosin otettu liitteestä 2. Muuttujia mitkä eivät kuulu laskureille ovat kääntäjät 1 ja 2, ja tässä tapauksessa halutaan niiden olevan auki asennossa ”0”. Vaaleanpunaisten väristen ja metallisten kappaleiden tunnistus on samantyylinen kuin mustan väristen kappaleiden tunnistus. Eroavaisuutena ovat halutut muuttujat ja niiden asennot eroavat toisistaan. Steppi 5:stä eteenpäin olevat stepit ovat tarkoitettu kappaleiden kuljetuksille omille tarkoitetuille alumiinisille radoille.

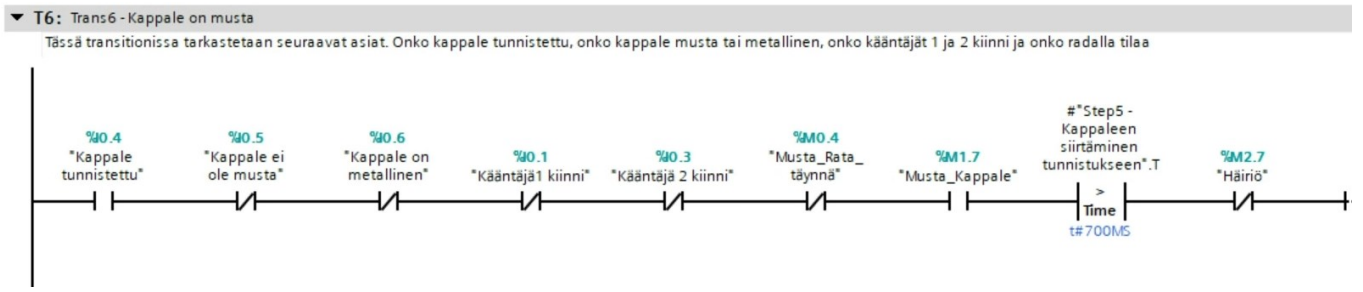
Kuva 7. Jakaja-aseman sekvenssi



Kuva 8. Sekvenssin aloituksen siirtymät



Kuva 9. Mustan värisen kappaleen tunnistus



Tämän sekvenssiohjelman suunnittelu sujui varsin erinomaisesti ilman mitään suurempaa ongelmaa. Ainoat ongelmat ilmenivät kappaleen tunnistuksessa olevien tarvittavien muuttujien ja niiden asentojen kanssa, ja sekvenssiohjelman automaattisen toiminnan toteutuksessa. Nämä ongelmat ratkesivat pienillä hienosäädöillä.

5.4 Erikoistilanteiden määrittäminen

Erikoistilanteiden suunnittelu voi olla haastava prosessi, koska siinä täytyy miettiä kaikki mahdolliset tavat, miten kyseinen työpiste voi mennä vikatilaan. Tämä ei ollut suuri ongelma, koska jakaja-asema on liikkuvilta osilta aika minimaalinen ja tästä syystä mahdollisia vikatilanteita on vähän. Erikoistilanteiden suunnitteluun SCL-kieli soveltuu taas erinomaisesti. Kuvissa 10 ja 11 nähdään kaikki erikoistilanteet mitä jakaja-asema voi kokea ja mitä voidaan seurata automaattitilassa. Näihin erikoistilanteisiin kuuluu kaikki häiriötilanteet, jotka ovat kääntäjien avautumis- sekä sulkeutumishäiriöt tai virhetilanne, jossa jokin kuulumaton kappale pääsee jakaja-asemalle. Kun mikä tahansa vikatilanne menee aktiiviseksi niin muuttuja ”Häiriö” menee asentoon ”1”. ”Häiriö” muuttuja pistää jakaja-aseman pysähdyksiin ja ei anna S7-GRAPH sekvenssin edetä stepeissä kuten nähdään. Kuvassa 11 nähdään hälytysviestit, jotka asettavat kyseiset häiriöt näkyville hälytysten käyttöliittymässä (kuva 21). Nämä häiriöt saa kuitattua ”Häiriö_Reset” muuttujalla, joka nimestä päätellen resetoi kaikki häiriöt. Erikoistilanteiden suunnitellessa ajastimien aikojen säätäminen toi eniten ongelmia, mutta ei mitään ylitsepääsemätöntä.

Kuva 10. Kääntäjien vikatilanteet

```
// Tässä blockissa käydään läpi kaikki mahdolliset erikoistilanteet mitä voi tulla vastaan.
//  Häiriön tullessa tämä koodi ei anna muiden ohjelmointiblockien toimia normaalisti.

// Tässä erikoistilanteessa ensimmäinen kääntäjä ei mene aukiasentoon (0)
//  Tämä ajastin resetoidaan yleisellä "Häiriö_Reset" napilla tai kun Jakaja-Asemal_Graph saavuttaa stepin 5.

"Kääntäjäl_Avautumis_Häiriö".TONR(IN:="Kääntäjäl kiinni" AND NOT "Kääntäjä 1 päälle",
                                  R:="Häiriö_Reset" OR "STEPPI" = 5,
                                  PT:=T#1500MS,
                                  Q=>"Kääntäjäl_Ei_Vapaudu",
                                  ET=>"Kääntäjäl_Vapautumis_Aika");


// Tässä erikoistilanteessa ensimmäinen kääntäjä ei mene kiinniasentoon (1).
//  Tämä ajastin resetoidaan yleisellä "Häiriö_Reset" napilla tai kun Jakaja-Asemal_Graph saavuttaa stepin 5.

"Kääntäjäl_Sulkeutumis_Häiriö".TONR(IN:="Kääntäjä 1 päälle" AND NOT "Kääntäjäl kiinni",
                                    R:="Häiriö_Reset" OR "STEPPI" = 5,
                                    PT:=T#2600MS,
                                    Q=>"Kääntäjäl_Ei_Sulkeudu",
                                    ET=>"Kääntäjäl_Sulkeutumis_Aika");


// Tässä erikoistilanteessa toinen kääntäjä ei mene aukiasentoon (0)
//  Tämä ajastin resetoidaan yleisellä "Häiriö_Reset" napilla tai kun Jakaja-Asemal_Graph saavuttaa stepin 5.

"Kääntäjä2_Avautumis_Häiriö".TONR(IN:="Kääntäjä 2 kiinni" AND NOT "Kääntäjä 2 päälle",
                                  R:="Häiriö_Reset" OR "STEPPI" = 5,
                                  PT:=T#1500MS,
                                  Q=>"Kääntäjä2_Ei_Vapaudu",
                                  ET=>"Kääntäjä2_Vapautumis_Aika");


// Tässä erikoistilanteessa toinen kääntäjä ei mene kiinniasentoon (1).
//  Tämä ajastin resetoidaan yleisellä "Häiriö_Reset" napilla tai kun Jakaja-Asemal_Graph saavuttaa stepin 5.

"Kääntäjä2_Sulkeutumis_Häiriö".TONR(IN:="Kääntäjä 2 päälle" AND NOT "Kääntäjä 2 kiinni",
                                    R:="Häiriö_Reset" OR "STEPPI" = 5,
                                    PT:=T#3500MS,
                                    Q=>"Kääntäjä2_Ei_Sulkeudu",
                                    ET=>"Kääntäjä2_Sulkeutumis_Aika");
```

Kuva 11. Tunnistamaton kappale ja muut häiriöt

```
// Tässä erikoistilanteessa tulee kappale radalle jota ei voida tunnistaa. Tämä kappale voi olla esimerkiksi läpinäkyvä.
//  Tämä ajastin resetoidaan yleisellä "Häiriö_Reset" napilla tai kun Jakaja-Asemal_Graph saavuttaa stepin 5.

"Tunnistamaton_Kappale".TONR(IN:="Kuljetin Päällä" AND NOT "Kappale tunnistettu" AND NOT "Stoppari auki",
                             R:="Häiriö_Reset" OR "STEPPI" = 5,
                             PT:=T#5000MS,
                             Q=>"Kappaletta_Ei_Tunnistettu");



// Tässä koodissa tarkastetaan yllä olevia häiriöitä. Jos häiriöitä tulee niin yleinen input "Häiriö" menee "1" asentoon.

IF "Kääntäjäl_Ei_Vapaudu" OR "Kääntäjäl_Ei_Sulkeudu" OR "Kääntäjä2_Ei_Vapaudu" OR "Kääntäjä2_Ei_Sulkeudu" OR "Kappaletta_Ei_Tunnistettu" THEN
    "Häiriö" := 1;
ELSE
    "Häiriö" := 0;
END_IF;

IF "Kääntäjäl_Ei_Vapaudu" THEN
    "AlmMsgW1TB0" := TRUE;
ELSE
    "AlmMsgW1TB0" := FALSE;
END_IF;

IF "Kääntäjäl_Ei_Sulkeudu" THEN
    "AlmMsgW1TB1" := TRUE;
ELSE
    "AlmMsgW1TB1" := FALSE;
END_IF;

IF "Kääntäjä2_Ei_Vapaudu" THEN
    "AlmMsgW1TB2" := TRUE;
ELSE
    "AlmMsgW1TB2" := FALSE;
END_IF;

IF "Kääntäjä2_Ei_Sulkeudu" THEN
    "AlmMsgW1TB3" := TRUE;
ELSE
    "AlmMsgW1TB3" := FALSE;
END_IF;

IF "Kappaletta_Ei_Tunnistettu"THEN
    "AlmMsgW1TB4" := TRUE;
ELSE
    "AlmMsgW1TB4" := FALSE;
END_IF;
```

5.5 Manuaaliajon mahdollistaminen

Manuaaliajon suunnittelu on tämän opinnäytetyön helpoin osuus ja tähänkin SCL-kieli on erittäin hyvä vaihtoehto. Manuaaliajon tarkoituksena on antaa käyttäjälle mahdollisuuden ajaa lähtömuuttujia aina tarvittaessa. Kuvassa 12 nähdään kaikki lähtömuuttujat, joita käyttäjä voi ohjata manuaalitilassa. Koodi on hyvin yksinkertainen, sillä käyttäjä määrittää erillisistä napeista onko manuaali- vai automaattitila päällä. Kun käyttäjä on asettanut manuaalitilan päälle, niin hän voi ohjata yksittäisiä liikkeitä kuvan 12 mukaisilla lähtömuuttujia.

Kuva 12. Manuaaliajo

```
// Tässä blockissa tarkastetaan onko "Manuaali ON" "1" ja jos on, niin manuaalisesti pystytään liikuttamaan seuraavia outputteja %Q0.0, %Q0.1, %Q0.2 sekä %Q0.3
//  Jos "Automaatti ON" on asennossa "1" niin silloin koodi on automaattitilassa eikä silloin pysty manuaalisesti ohjata yksittäisiä liikkeitä.


IF "Automaatti ON" THEN
    "Manuaali ON" := FALSE;
    "Manual_Kuljetin_ON" := FALSE;
    "Manual_Stoppari_Auki" := FALSE;
    "Manual_Kääntäjä1_Kiinni" := FALSE;
    "Manual_Kääntäjä2_Kiinni" := FALSE;
END_IF;

// Alla olevassa koodissa on manuaalisesti liikutettavat kohteet.

IF "Manuaali ON" AND "STEPPI" = 3 THEN
    "Automaatti ON" := FALSE;
    IF "Manual_Kuljetin_ON" THEN              // Kuljettimen ajo manuaalisesti
        "Kuljetin Päällä" := TRUE;
    ELSE
        "Kuljetin Päällä" := FALSE;
    END_IF;
    IF "Manual_Stoppari_Auki" THEN            // Stopparin ajo manuaalisesti
        "Stoppari auki" := TRUE;
    ELSE
        "Stoppari auki" := FALSE;
    END_IF;
    IF "Manual_Kääntäjä1_Kiinni" THEN         // Ensimmäisen kääntäjän ajo manuaalisesti
        "Kääntäjä 1 päälle" := TRUE;
    ELSE
        "Kääntäjä 1 päälle" := FALSE;
    END_IF;
    IF "Manual_Kääntäjä2_Kiinni" THEN         // Toisen kääntäjän ajo manuaalisesti
        "Kääntäjä 2 päälle" := TRUE;
    ELSE
        "Kääntäjä 2 päälle" := FALSE;
    END_IF;
END_IF;
```

5.6 Käyttöliittymän luonti

Tämä käyttöliittymä on konsepti siitä minkälainen se voisi olla. Puutteellisten lisenssien takia ei pystytty toteuttamaan käyttöliittymää koulun ohjausjärjestelmälle. Suunnittelussa suurin haaste on miettiä, mitä kaikkea käyttäjän olisi hyvä nähdä käyttöliittymästä. On myös mahdollista suunnittelun tarvitsevan lisää ohjelmointia kuten kuvissa 13, 14 ja 15 nähdään.

Kuva 13 on otettu erikoistilanteiden koodiohjelmasta ja kyseisen koodin tarkoituksena on asettaa välimuuttujat päälle, jotka näkyvät MainScreen käyttöliittymässä kuvassa 16. Esimerkiksi jos mustien kappaleiden rata on täynnä ja stopparilla ilmenee olevan mustan värinen kappale. Tällöin tämä välimuuttuja ”Musta_Rata_Täynnä + Musta_Kappale_Tunnistettu” menisi päälle ja ilmoittaisi tästä käyttäjälle käyttöliittymässä. Tämän tyylinen ohjelmointi toistuu vaaleanpunaisten sekä metallisten kappaleiden kanssa.

Kuva 13. Käyttöliittymän rata täynnä ja kappale tunnistettu

```
// Tässä koodissa ei ole niinkään häiriötilanteita vaan on kappale tunnistettu jonka rata on täynnä
// Riveillä 88-92 Mustien kappaleiden rata on täynnä silloin kun mustan värinen kappale on tunnistettu

    IF "Musta_Rata_Täynnä" AND "Kappale tunnistettu" AND NOT "Kappale ei ole musta" AND NOT "Kappale on metallinen" THEN
        "Musta_Rata_Täynnä + Musta_Kappale_Tunnistettu" := TRUE;
    ELSE
        "Musta_Rata_Täynnä + Musta_Kappale_Tunnistettu" := FALSE;
    END_IF;

// Riveillä 96-100 vaaleanpunaisten kappaleiden rata on täymmä silloin kun vaaleanpunaisen värinen kappale on tunnistettu

    IF "Vaaleanpunainen_Rata_Täynnä" AND "Kappale tunnistettu" AND "Kappale ei ole musta" AND NOT "Kappale on metallinen" THEN
        "Vaaleanpunainen_Rata_Täynnä + Vaaleanpunainen_Kappale_Tunnistettu" := TRUE;
    ELSE
        "Vaaleanpunainen_Rata_Täynnä + Vaaleanpunainen_Kappale_Tunnistettu" := FALSE;
    END_IF;

// Riveillä 104-108 metallisten kappaleiden rata on täynnä silloin kun metallinen kappale on tunnsitettu

    IF "Metalli_Rata_Täynnä" AND "Kappale on metallinen" THEN
        "Metalli_Rata_Täynnä + Metalli_Kappale_Tunnistettu" := TRUE;
    ELSE
        "Metalli_Rata_Täynnä + Metalli_Kappale_Tunnistettu" := FALSE;
    END_IF;
```

Kuvassa 14 ohjelmoidaan SCL-kielellä. Tässä koodissa verrataan laskurien arvoja ja sen mukaan laittaa välimuuttujat päälle, jotka ilmoitetaan MainScreenin käyttöliittymässä kappaleiden omissa radoissa. Kuvassa 14 nähdään, miten verrataan mustien kappaleiden laskurin arvoja ja mitä muuttujia menee päälle, kun arvot toteutuvat. Kuvassa 15 koodissa määritetään anturien tiedot (kuva 6) omille välimuuttujille, jotka näkyvät päänäytöllä stopparin alla kuvan 17 tyylisesti, jossa on ”Musta_Kappale_Tunnistettu” päällä.

Kuva 14. Käyttöliittymä kappaleiden visualisointi radoilla

```
6
7  "1. Musta_KPL_Timer_ON".TON(IN:="Musta_Laskuri".CV >=1,
8                              PT:=T#4150MS,
9                              Q=>"1.Musta_Kappale");
10
11 "2. Musta_KPL_Timer_ON".TON(IN:="Musta_Laskuri".CV >=2,
12                             PT:=T#3950MS,
13                             Q=>"2.Musta_Kappale");
14
15 "3. Musta_KPL_Timer_ON".TON(IN:="Musta_Laskuri".CV >=3,
16                             PT:=T#3750MS,
17                             Q=>"3.Musta_Kappale");
18
```

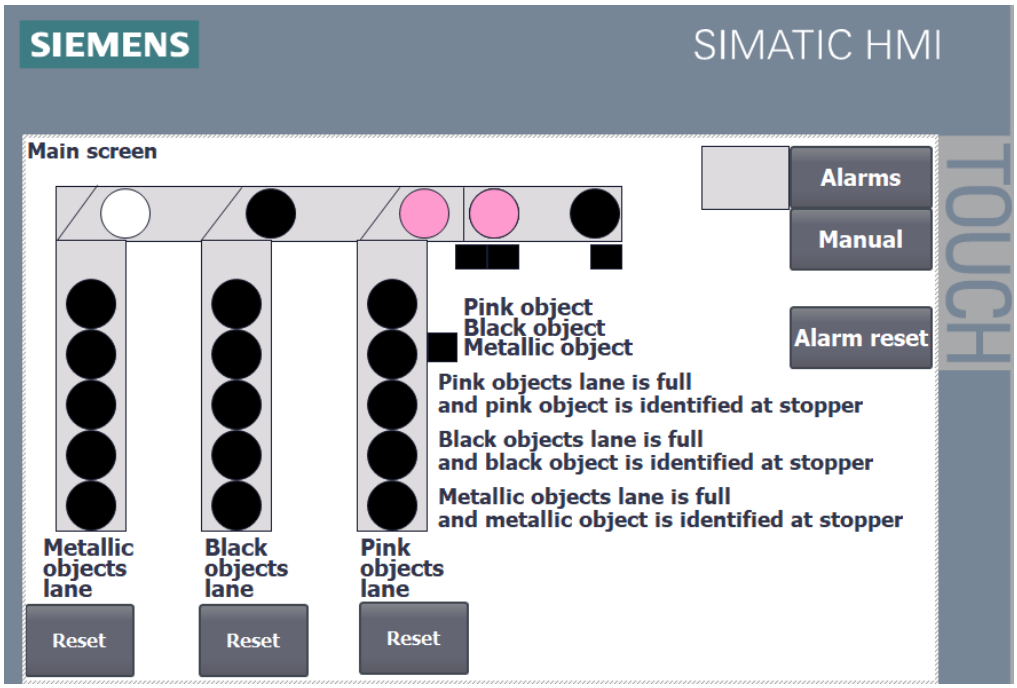
Kuva 15. Kappaleiden tunnistusta päänäytölle

```
IF "Kappale tunnistettu" AND NOT "Kappale ei ole musta" AND NOT "Kappale on metallinen" THEN
    "Musta_Kappale_Tunnistettu" := TRUE;
ELSE
    "Musta_Kappale_Tunnistettu" := FALSE;
END_IF;

IF "Kappale tunnistettu" AND "Kappale ei ole musta" AND NOT "Kappale on metallinen" THEN
    "Vaaleanpunainen_Kappale_Tunnistettu" := TRUE;
ELSE
    "Vaaleanpunainen_Kappale_Tunnistettu" := FALSE;
END_IF;

IF "Kappale on metallinen" THEN
    "Metallinen_Kappale_Tunnistettu" := TRUE;
ELSE
    "Metallinen_Kappale_Tunnistettu" := FALSE;
END_IF;
```

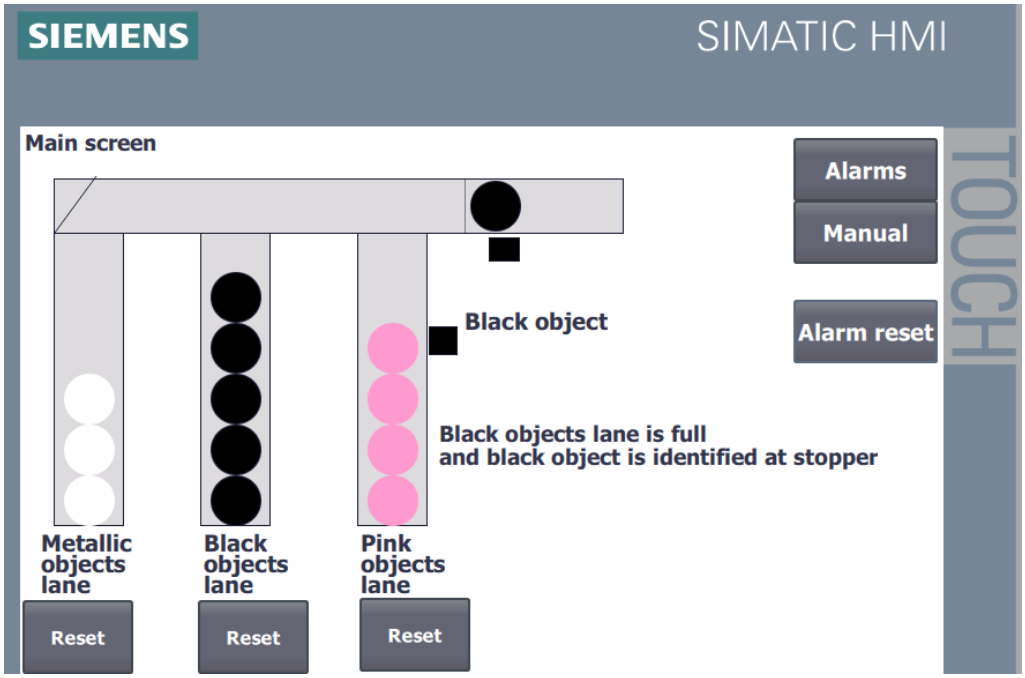
Kuvissa 16, 19 ja 21 nähdään esimerkki käyttöliittymät jakaja-asemalle. Luonnokset eivät ole kauneimmat hyvin niukkojen vaihtoehtojen takia luoda hyvännäköisiä käyttöliittymiä, mutta ajavat silti asiansa. Nämä käyttöliittymät jakautuvat päänäytölle (MainScreen), manuaaliajon näytölle (ManualScreen) ja hälytysnäytölle (AlarmScreen).
”MainScreen” käyttöliittymässä nähdään seuraavia asioita. Onko ensimmäinen paikka-anturi tunnistanut kappaleen. Onko stopparilla kappale ja mikä kappale on kyseessä. Nähdään kääntäjien tilat. Nähdään kuinka monta kappaleita ovat kuljettimilla. Voidaan lukea ovatko radat täynnä ja onko samanlainen kappale tunnistettu, jonka rata on täynnä. Nähdään hälytystila ja ne voidaan resetoida. Voidaan kuitata radat kerätyiksi ja voidaan mennä joko hälytysnäytölle (kuva 21) tai manuaalinäytölle (kuva 19).

Kuvissa 17 ja 18 näkyvät esimerkkitilanteet mitä päänäytöllä voisi näkyä. Esimerkiksi kuvassa 17 nähdään mustien kappaleiden radan olevan täynnä ja samaan aikaan stopparilla on mustan värinen kappale. Samalla vaaleanpunaisten kappaleiden radalla on neljä kappaletta ja metallisten kappaleiden radalla on kolme kappaletta. Kuvassa 18 nähdään miltä päänäyttö voisi näyttää, kun häiriötila menee päälle.

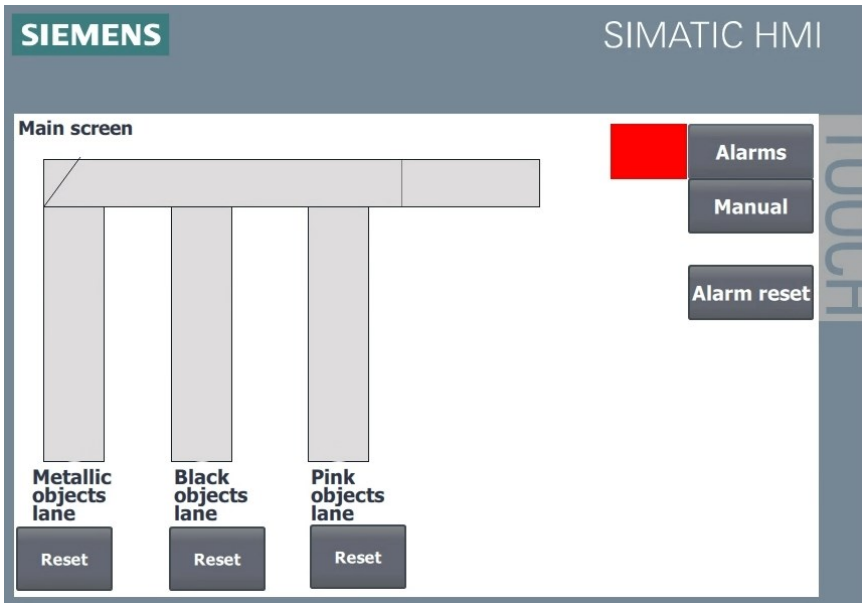
Kuva 16. Käyttöliittymä MainScreen suunnittelukuva



Kuva 17. Käyttöliittymä MainScreen esimerkki 1

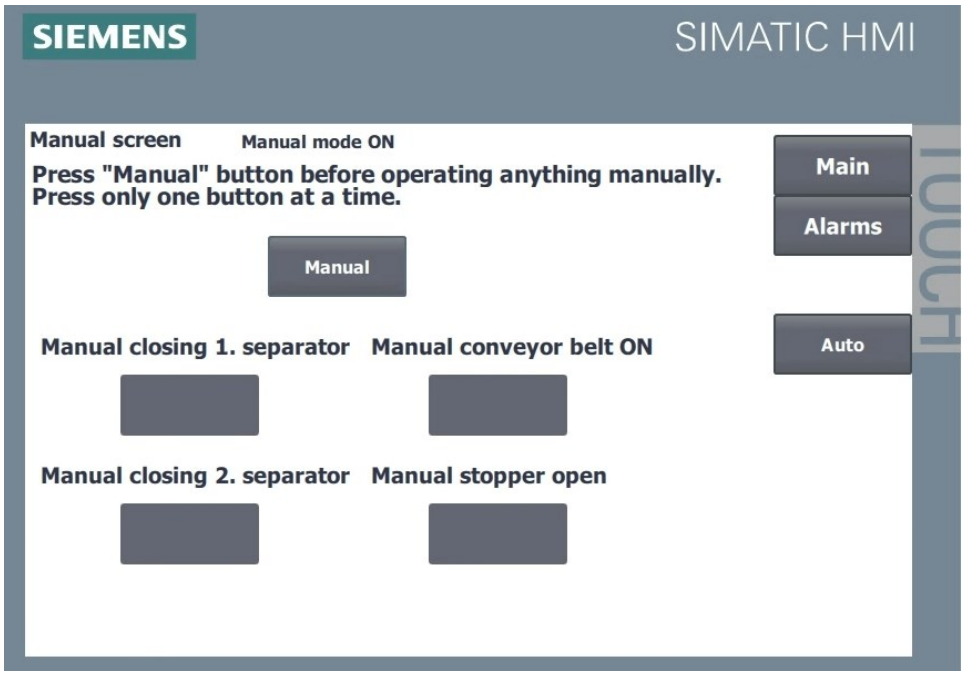


Kuva 18. Käyttöliittymä MainScreen esimerkki 2

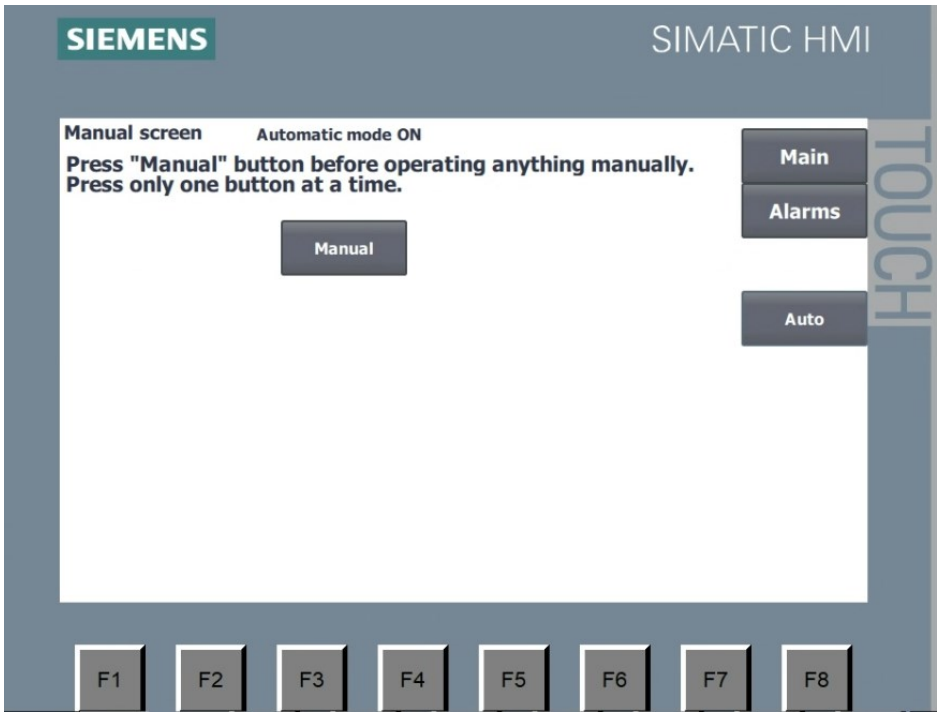


Kuvassa 19 mahdollistetaan manuaaliajo kaikille lähtömuuttujille, kunhan käyttäjä on asettanut manuaalitilan päälle painamalla ”Manual” nappia ManualScreen:ssä. Kun käyttäjä haluaa asettaa jakaja-aseman automaattitilaan, niin se tapahtuu painalla ”Auto” nappia ManualScreen:ssä. Tämän jälkeen manuaaliajoon tarkoitetut napit katoavat niin pitkäksi aikaa, kunnes käyttäjä haluaa asettaa jakaja-aseman manuaalitilaan (ks. kuva 20). Tämä ohjelma toimii samalla koodilla, mikä nähdään kuvasta 12. Tästä näytöstä pääsee samalla tavalla päänäytölle tai hälytysnäytölle.

Kuva 19. Käyttöliittymä ManualScreen manuaalitilassa



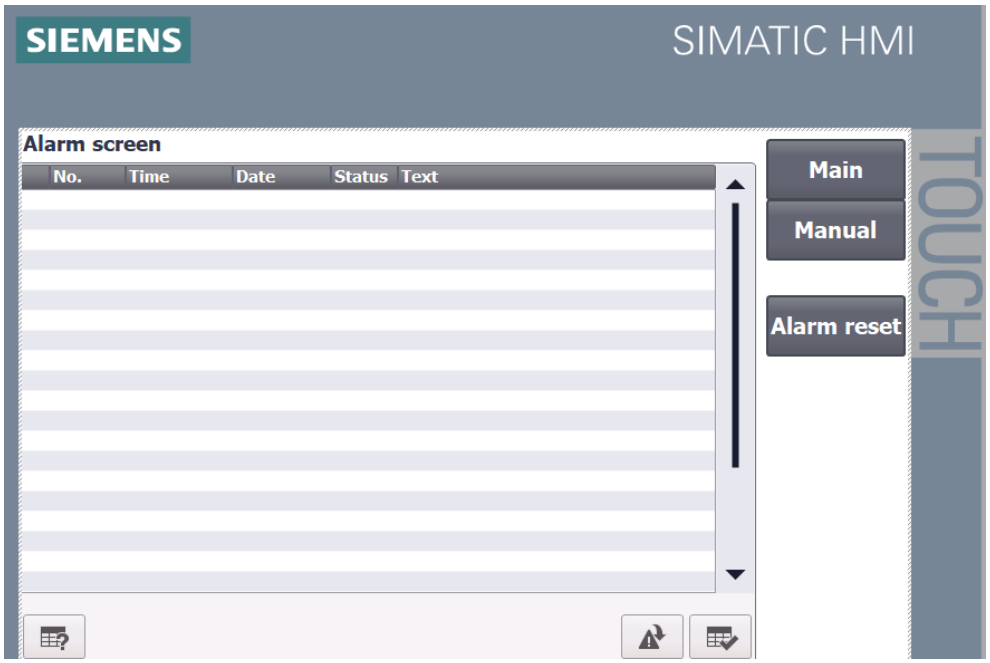
Kuva 20. Käyttöliittymä ManualScreen automaattitila



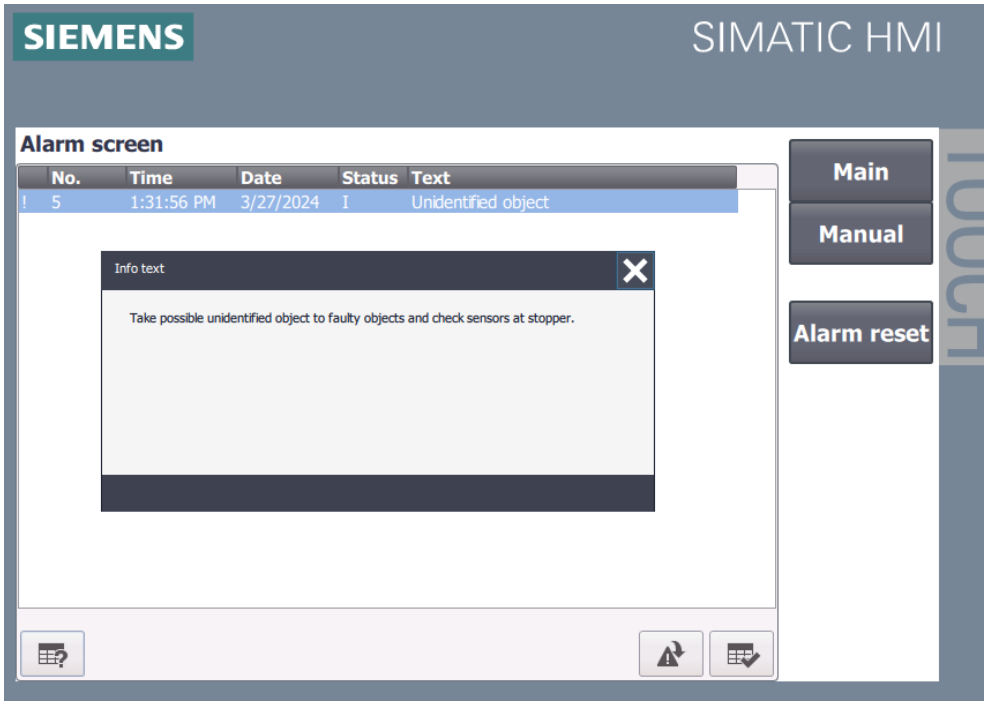
Kuvassa 21 nähdään ”AlarmScreen”, jonka tarkoituksena on näyttää käyttäjälle kaikki hälytykset, mitkä tulevat jakaja-asemalta. Hälytyksen tiedot voidaan katsoa hälytysnäytön vasemmassa alakulmassa olevasta napista ja kuvassa 22 nähdään miltä se näyttää.

Hälytykset voidaan kuitata painamalla ”Alarm reset” nappia. Vian korjattua, se voidaan poistaa listalta painamalla hälytysnäytön oikealla alakulmassa olevaa nappia. Tästä näytöstä päästään päänäytölle sekä manuaalinäytölle.

Kuva 21. Käyttöliittymä AlarmScreen, kun ei ole hälytystä



Kuva 22. Käyttöliittymä AlarmScreen hälytyksellä ja selityksellä



5.7 Virtuaalimallin ohjauksen eroavaisuudet fyysisen ohjauksen kanssa

Ohjaajieni kanssa keskusteltuaan siitä, miten fyysiselle jakaja-asemalle voidaan asettaa useamman kappaleen kerrallaan, ja miten tämän saisi toteutettua virtuaalimallille. Ohjaajieni kanssa päädyttiin lopputulemaan, ettei lähdetä ratkomaan tätä ongelmaa virtuaalimalliin, jolloin virtuaalimallissa on aina yksi kappale ennen stopparia. Tästä syystä virtuaalimallin ohjaus eroaa vähäsen fyysisen jakaja-aseman ohjauksesta (kuva 2). Nämä eroavaisuudet ovat lähinnä kappalelaskennassa ja sekvenssiohjauksessa, koska fyysisellä toimipisteellä kuljetinnopeus ja tapa laskea kappaleita eroaa virtuaalimallista. Kaikki eroavaisuudet nähdään kuvista 23, 24 ja 25.

Kuvassa 23 nähdään ”Laskurin_Ajastin” TON-ajastin, joka eroaa liitteestä 1. Fyysisen jakaja-asemalla oli suurin ongelmana erotella laskurille vaaleanpunaiset kappaleet mustista kappaleista. Tämän ongelman ratkesi kuvan 23 mukainen SCL-koodinpätkä. Tässä käytetään stopparilla olevan toisen anturin (kuva 6) nollauksessa joko alumiiniratojen omia kuittausnappeja tai ”Stoppari auki” muuttujaa. ”Stoppari auki” on kuljettimella sijaitsevan stopparin lähtömuuttuja. ”Kappale tunnistettu” on stopparilla sijaitsevan anturin tulomuuttuja, joka ei aina nollaannu tarvittaessa, ja tämä aiheuttaa ongelmia kappaleiden laskennassa. Kun ”Kappale tunnistettu” muuttuja käy asennossa ”0”, se käynnistää ”Laskurin_Ajastin” TONR-ajastimen uusiksi. Samalla täytyy nollauksessa oleva ”Stoppari auki” muuttujan käyvän asennossa ”1”. Tämä ajastin viivyttää värillisten kappaleiden tunnistusta 450 millisekunnilla.

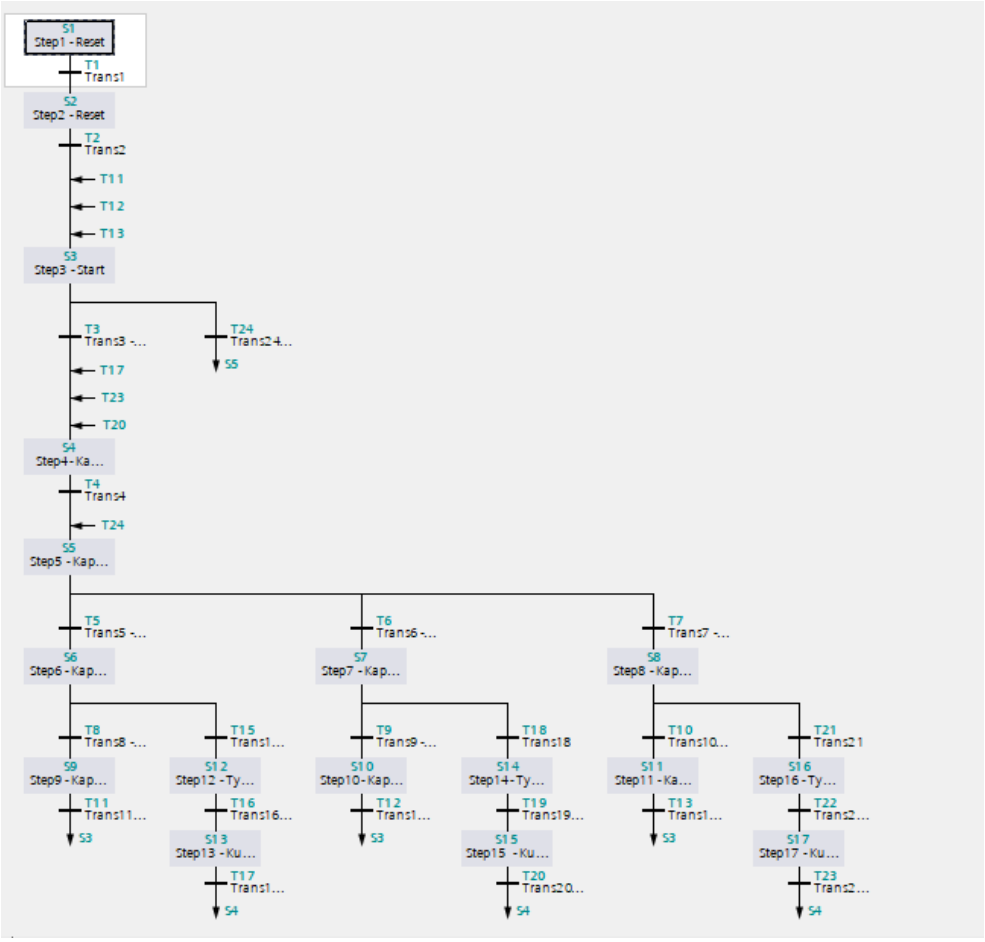
Kuva 23. Laskurin ajastin fyysiselle jakaja-asemalle

```
1  // Tämä FB ohjelman tarkoituksena on laskea kolmelle radalle oikean määrän (5) kappaleita.
2
3  // "Laskurin_Ajastin" on ajastin jonka avulla tunnistetaan vaaleanpunainen kappale.
4  // Tunnistus perustuu Inputin "Kappale tunnistettu" (I%0.4) "1" asennon myöhäistämiseen.
5
6  IF "Musta_Rata_Kuittaus" OR "Vaaleanpunainen_Rata_Kuittaus" OR "Metalli_Rata_Kuittaus" OR "Stoppari auki" THEN
7      "Kappale tunnistettu" := FALSE;
8  END_IF;
9
10 "Laskurin_Ajastin".TONR(IN := "Kappale tunnistettu",
11                         R := "Stoppari auki",
12                         PT := T#450MS,
13                         ET => "Laskurin_Ajastimen_Aika");
14
15
16
17
```

Kuvassa 24 nähdään fyysisen jakaja-aseman sekvenssiohjaus, joka eroaa vähäsen virtuaalimallista (kuva 7). Tätä luodessa piti ottaa huomioon eri tilanteita, milloin kappale voisi

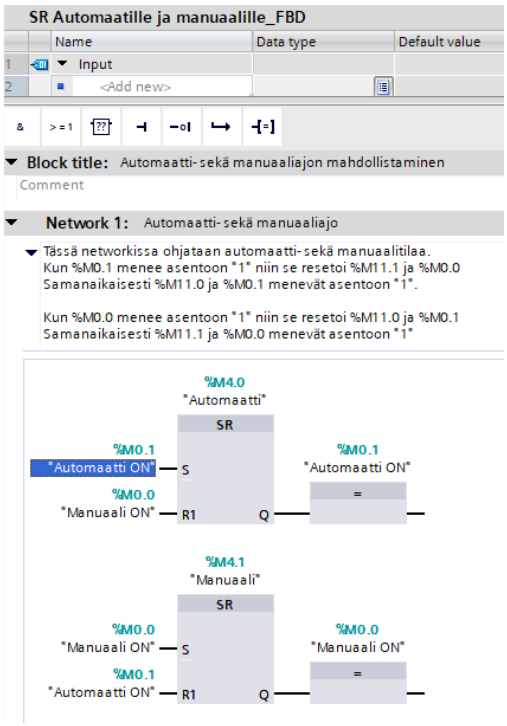
päätyä kuljettimelle. Esimerkiksi stepissä 3 on kaksi siirtymää, jossa tarkastetaan, onko kappale edennyt jo stopparille. Jos kappale on kuljettimen alkupäässä, niin silloin edetään steppiin 4. Jos kappale on edennyt jo stopparille, niin silloin edetään steppiin 5. Toinen esimerkki on vaaleanpunaisen kappaleen kuljettamisen stepissä 6, jossa on myös kaksi erilaista siirtymää. Näissä siirtymissä tarkastetaan myös uuden kappaleen paikkaa kuljettimella. Jos kuljettimen alkupäässä ei ole kappaletta, niin silloin edetään lopuksi steppiin 3. Jos kuljettimen alkupäässä on kappale, niin silloin edetään lopuksi steppiin 4. Stepissä 4 tarkastetaan, onko kappale tunnistettavissa. Jos kappaletta ei voida tunnistaa, niin silloin on kyseessä tunnistamaton kappale ja jakaja-asema menee häiriötilaan.

Kuva 24. Fyysisen jakaja-aseman sekvenssiohjaus



Kuvassa 25 on toimilohkokaavio, jossa määritetään automaatti-/manuaalitila RS-kytkimillä. Jos automaattitila on asennossa ”1”, niin silloin manuaalitila menee asentoon ”0”. Jos manuaalitila menee asentoon ”1”, niin silloin automaattitila menee asentoon ”0”.

Kuva 25. Fyysisen jakaja-aseman automaatti-/manuaaliohjaus



6 Virtuaalimallin toteutus

Tässä luvussa käydään läpi Festo MPS-alustan mallinnusta käyttäen SIMIT SP -mallinnusohjelmistoa. Muistetaan tämän projektiosuuden tavoitteiden olevan riittävän realistinen virtuaalimalli Festo MPS-alustasta. Antureiden ja toimilaitteiden tilojen visualisointi ja kappaleiden sijainnin visualisointi. Vikatilanteiden simulointimahdollisuus kuten kääntäjävika tai tunnistamaton kappale. Työn on oltava selkeä ja täytyy olla helposti luettava rakenne. Lopuksi täytyy luoda erillinen käyttöohje virtuaalimallista ja geneeristen kirjastokomponenttien kehittäminen kuten hihnakuljettimesta.

Tässä luvussa tarkastetaan virtuaalimallia, joka on integroitu TIA Portaalin kanssa. TIA Portaalissa on toteutettu virtuaalimallin ohjaus, joka esiteltiin luvussa 5. Tämän virtuaalimallin tarkoituksena on toimia Feston jakaja-asemana virtuaalisena kaksosena. Esimerkiksi virtuaalimalli lähettää TIA Portaaliin anturitietoja, joiden avulla ohjaus toteutetaan, ja samalla virtuaalimallissa esitetään kyseisten anturien toimintaa sekä kappaleiden mallinnusta.

SIMIT SP työssä on käytetty useita erilaisia komponentteja, joilla on omat syynsä, miksi niitä käytetään. Esimerkiksi ”INT”-komponentti eli integraatiokomponenttia käytetään kappaleiden liikeradan simulointiin. Toisena esimerkkinä ovat erilaiset ajastimet ”TON”, ”TOF” ja ”TP”,

joilla simuloidaan pääsääntöisesti anturitietoja. Ajastimia käytetään myös muuhun tarkoitukseen, joita käydään läpi tässä luvussa. Tässä työssä käytettiin myös RS-komponentteja, joiden määräävänä tulosignaalina on ”R” eli reset. Muita komponentteja mitä käytettiin ovat ”AND”-, ”OR”-, ”Compare”- ja ”NOT”-komponentteja. Jokaista työtilaa luetaan vasemmalta ylhäältä rivi kerrallaan alaspäin.

6.1 Kuljettimen alkupää ja stoppari

Tässä alaluvussa käydään läpi kuljettimen alkupään, stopparin ja stopparilla olevan laskurin toimintaa työn ensimmäisessä versiossa. Tämän työn osuus on samanlainen jokaisen kappaleen kanssa, niin käydään läpi vaaleanpunaisen kappaleen mallinnusta tässä alaluvussa.

6.1.1 Kuljettimen alkupäästä stopparille

Kuvassa 26 nähdään esimerkkimallinnus vaaleanpunaiselle kappaleelle, joka on simuloitu liikkumaan kuljettimen alkupäästä stopparille. Kun käyttäjä painaa päänäytöltä nappia ”Pink object” (vrt. kuva 39), se aloittaa sekvenssin omaisesti vaaleanpunaisen kappaleen simuloinnin jakaja-asemalla. Kuvassa 26 vaaleanpunainen yhdistin on yhdistettynä suureen AND-komponenttiin. Tässä AND-komponentissa tarkastetaan, onko pelkästään yhdistin ”Vaaleanpunainen” aktiivisena. Jos on, niin signaali yhdistyy ensimmäiseen RS-komponentin SET-tulosignaaliin. Ensimmäisen RS-komponentin resettinä toimii myös häiriösignaali (vrt. kuva 34) ja kappaleiden omat anturi resetit (esim. Vaaleanpunainen anturi reset löytyy kuvasta 31). RS-komponentin lähtösignaali yhdistyy toiseen AND-komponenttiin, jossa tarkastetaan, onko kuljetin asennossa ”1”. Toisen AND-komponentin lähtösignaali yhdistyy toisen RS-komponentin SET-tulosignaaliin. Toisen RS-komponentin ”R” tulosignaalissa ovat kappaleiden liike kuljettimen alkupäässä ja kappaleiden omat anturi resetit. Jos alkupään kuljettimella on esimerkiksi musta kappale, se asettaa yhdistimen ”Musta_Liikkeellä” avulla jälkimmäisen RS-komponentin lähtösignaalin arvoon ”0”. Jos kuljetin on tyhjä ja käyttäjä on valinnut vaaleanpunaisen kappaleen skenaarioksi, niin jälkimmäisen RS-komponentin lähtösignaali on arvossa ”1”. Tämä RS-komponentti on kytkettynä yhdistimeen ”Tunnistettu VP KPL”, joka on yhdistettynä NotC-komponentin kautta integraatiokomponentille. Tämä

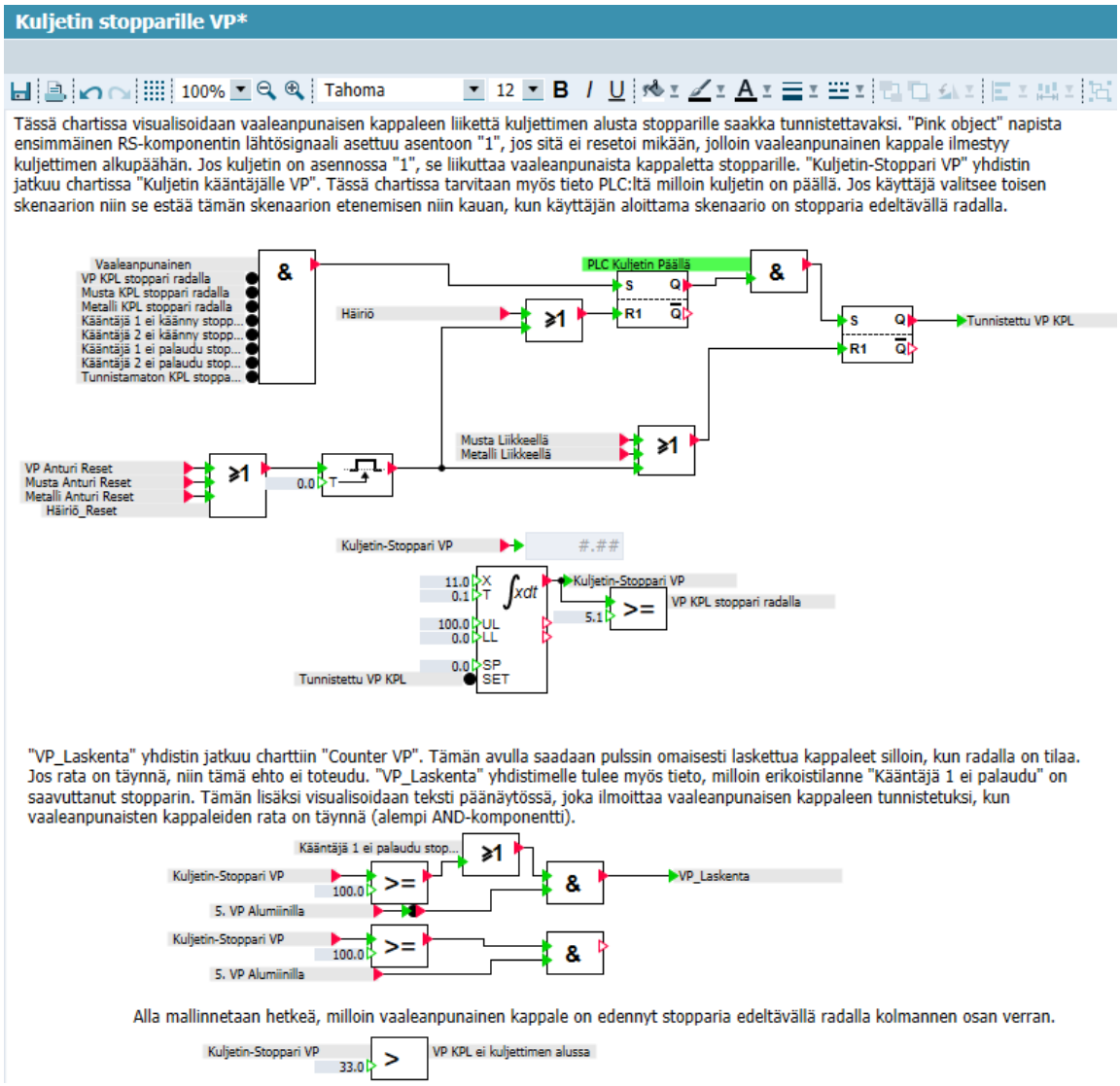
integraatiokomponentti laskee kappaleen paikkaa kaavan 1 mukaisesti. (Siemens, 2023b, s. 468)

Kaava 1. Integraatiokomponentin kaava (Siemens, 2023b, s. 468)

$$y = \frac{1}{T}\int x\,dt$$

Kun katsotaan kuvan 26 alapuolta, siellä tarkastellaan kyseisen integraatiokomponentin arvoa yhdistimeltä ”Kuljetin-Stoppari VP”. Kun integraatiokomponentti on saanut arvon ”100”, niin Compare-komponentin lähtösignaali asettuu arvosta ”0” arvoon ”1”, joka yhdistyy AND-komponenttiin. AND-komponentin toisena ehtona on vaaleanpunaisten kappaleiden alumiiniradalla olevan tilaa vaaleanpunaisille kappaleille. Jos kyseisellä alumiiniradalla ei ole tilaa, niin silloin menee yhdistin ”5. VP Alumiinilla” asentoon ”1”. Tämä asettaa NotC-komponentin lähtösignaalin arvoon ”0”, jolloin AND-komponentin ehdot eivät toteudu. Jos AND-komponentin ehdot toteutuvat, se asettaa yhdistimen ”VP_Laskenta” arvoon ”1”, joka on yhdistettynä työtilassa ”Counter VP” (vrt. kuva 30). Jos vaaleanpunaisten kappaleiden alumiinirata on täynnä ja tämän integraatiokomponentin arvo on ”100”, se asettaa alimman AND-komponentin lähtösignaalin arvoon ”1”. Tämä ilmoittaa käyttäjälle tekstillä alumiiniradan olevan täynnä ja vaaleanpunaisen kappaleen olevan tunnistettuna stopparilla. Alimmassa Compare-komponentissa tarkastetaan integraatiokomponentin arvoa. Jos integraatiokomponentin arvo on yli ”33”, se asettaa yhdistimen ”VP KPL ei kuljettimen alussa” asentoon ”1”. Tämä yhdistin nollaa virtuaalimallin kuljettimen alkupäässä olevan anturin (kuva 6 merkattu ”1”).

Kuva 26. Kuljetin stopparille VP



6.1.2 Stopparin anturitietojen mallinnus

Kuten kuvista 27, 28 ja 29 nähdään näiden työtilojen olevan hyvin samankaltaisia, vaikka tässä käsitellään kuitenkin eri kappaleiden anturitietoja. Nämä anturit sijaitsevat stopparin läheisyydessä kuten kuvasta 6 voidaan tarkastella.

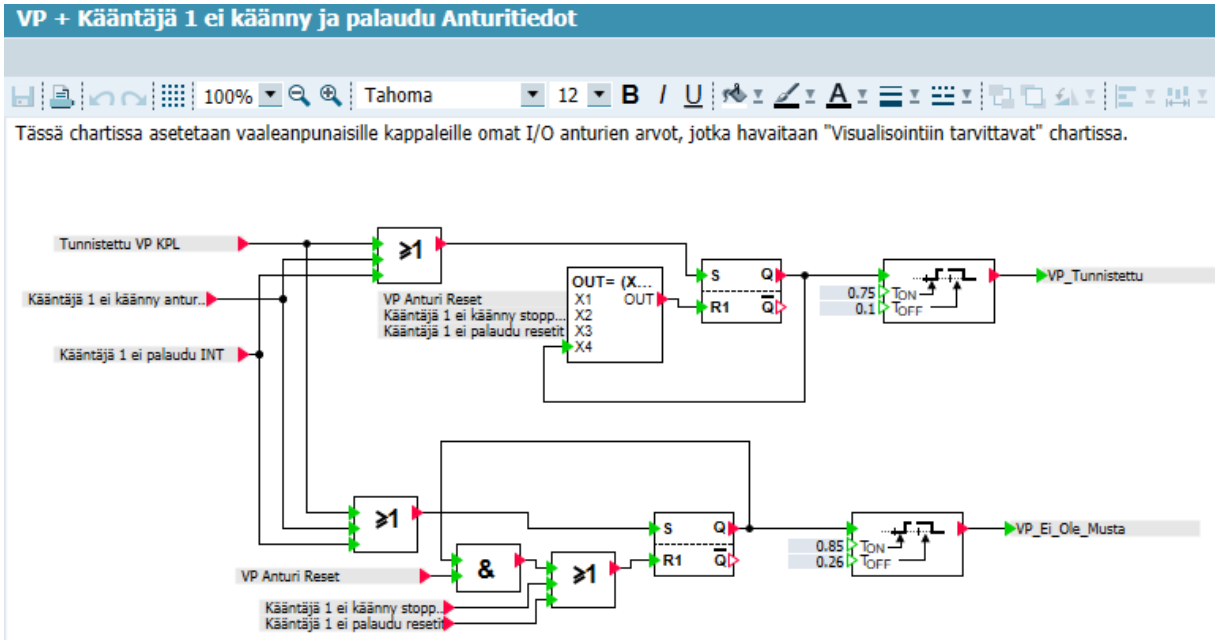
Kuvassa 27 nähdään aikaisemmassa työtilassa oleva yhdistin ”Tunnistettu VP KPL”, jonka löytää kuvasta 26. Toinen yhdistin on ”Kääntäjä 1 ei käänny anturitiedot”, joka tulee työtilasta ”Kääntäjä 1 ei käänny”, ja tämä on vikatilanne, joka käydään läpi luvussa 6.4. Kolmas yhdistin on ”Kääntäjä 1 ei palaudu INT”, joka tulee työtilasta ”Kääntäjä 1 ei palaudu”, tämäkin on vikatilanne, joka käydään luvussa 6.4. Nämä yhdistimet yhdistyvät OR-komponenttien

kautta kahteen RS-komponenttiin ja asettavat ne arvoon ”1”, jos niitä ei resetoi mikään. Esimerkiksi ylemmän RS-komponentin nollauksena toimii BFormula-komponentti, joka päästää lähtösignaalin läpi kaavan 2 mukaisesti. BFormula-komponentin tulosignaaleissa ovat yhdistimet ”VP Anturi Reset” (vrt. kuva 31), ”Kääntäjä 1 ei käänny stopparilla” (vrt. kuva 35), ”Kääntäjä 1 ei palaudu resetit” (vrt. kuva 36) ja RS-komponentin lähtösignaali. Nämä tulosignaalit ovat valittuna automaattisen toiminnan vuoksi. TON-ajastin on asetettuna aikaan 750ms, koska anturi sijaitsee stopparin läheisyydessä, kuten kuvasta 6 nähdään merkattuna ”2” numerolla. TON-ajastimen merkitys pätee samalla tavalla kuvissa 28 ja 29. Alemman RS-komponentin nollauksena on OR-komponentti. OR-komponentissa on yhdistimet ”Kääntäjä 1 ei käänny stopparilla” ja ”Kääntäjä 1 ei palaudu resetit” ja kolmantena ehtona on AND-komponentti. AND-komponentin tulosignaaleissa ovat samat ehdot kuten BFormula-komponentissa, eli ”VP Anturi Reset” ja RS-komponentin lähtösignaali. Alempi TON-ajastin on asetettu eri aikaan kuin ylempi TON-ajastin. Tämän syynä on kyseisen anturi sijainti, joka on stopparin vierellä kuten kuvasta 6 nähdään ”3”:lla merkatulla anturilla. Tämä anturi toistuu samalla tavalla kuvan 29 mukaisesti.

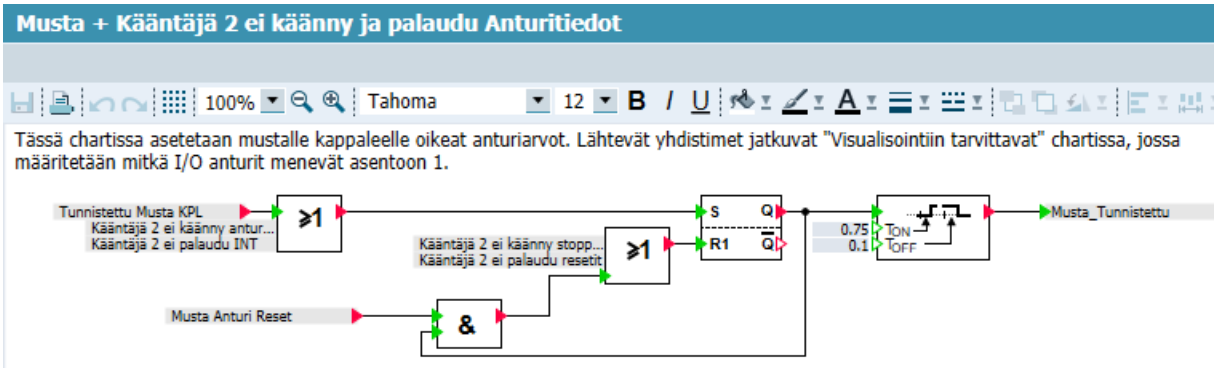
Kaava 2. BFormulan kaava

$$(X1\ AND\ X4)\ OR\ X2\ OR\ X3$$

Kuva 27. Vaaleanpunaisen kappaleen anturitiedot

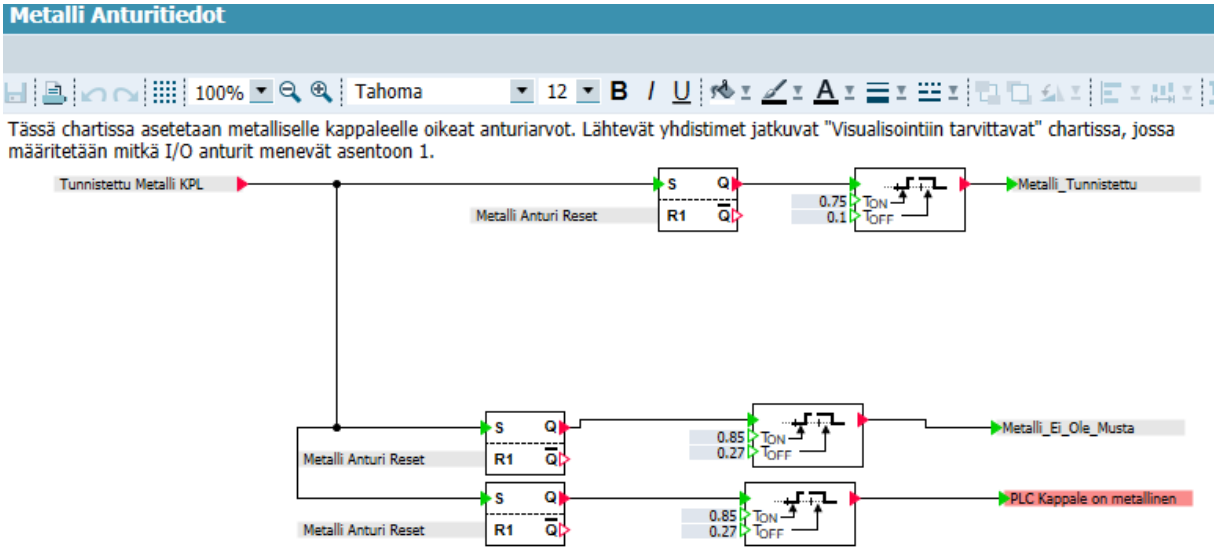


Kuva 28. Mustan kappaleen anturitiedot



Kuvassa 29 nähdään ihan alhaalla olevan punainen yhdistin, joka on PLC:n tulosignaali ”PLC kappale on metallinen”. Tämä muuttuja on jakaja-aseman I/O listauksesta ja tämä anturi nähdään kuvassa 6 ”3”:lla merkattuna oleva anturi. Tämän yhdistimen TON/TOF-ajastimen resetointi tapahtuu samalla tavalla, kuin aikaisemmissa olevien ajastimien resetointi kuvissa 27 ja 28.

Kuva 29. Metallisen kappaleen anturitiedot



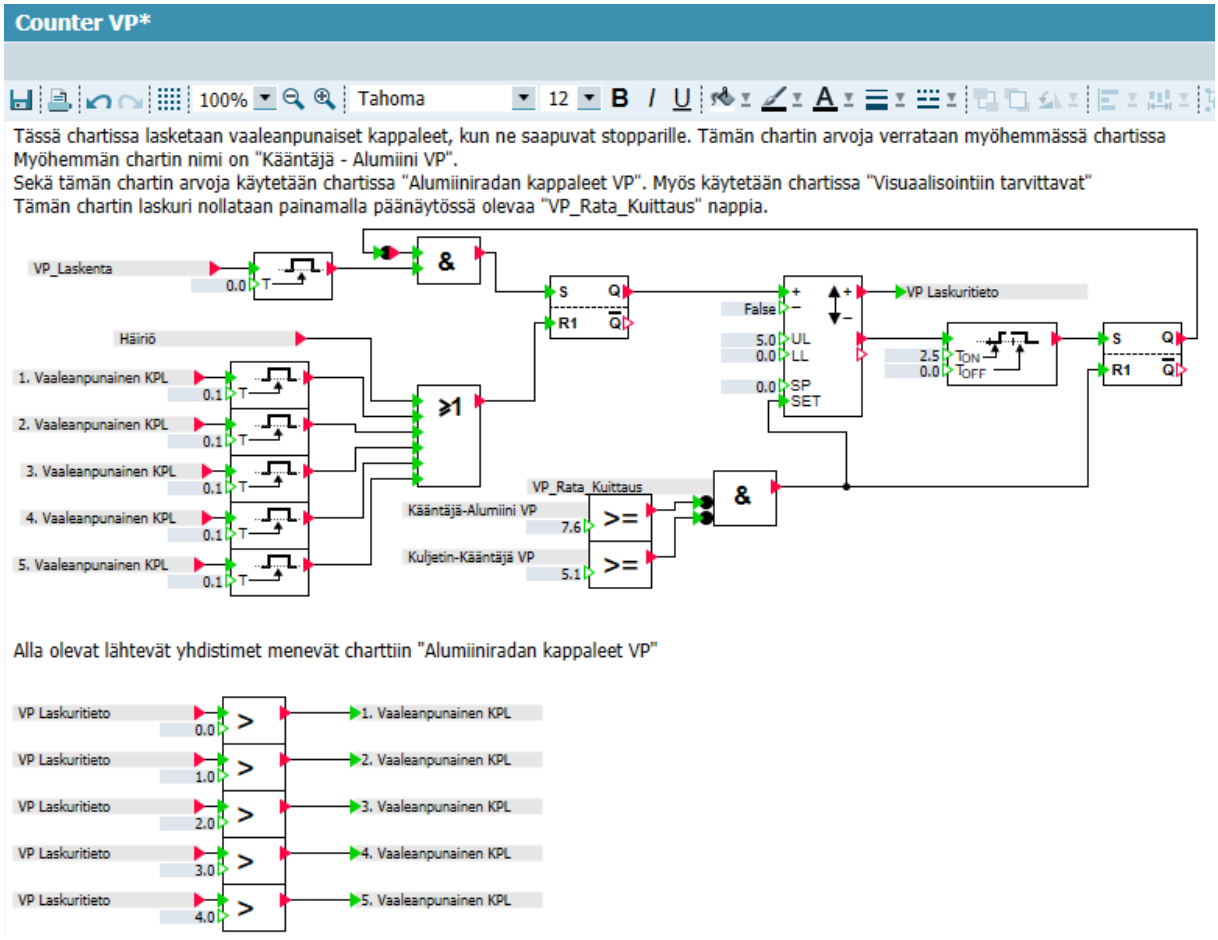
6.1.3 Laskurin toiminta stopparilla

Kuvassa 30 nähdään vaaleanpunaisten kappalelaskurin mallinnus stopparin kohdalla. Kuvassa 30 nähdään vasemmalla yläpuolella tuttu yhdistin ”VP_Laskenta”. Tämä oli aikaisemmassa työtilassa ”Kuljetin Stoppari VP” (kuva 26) toiseksi alimman ehdon yhdistimenä. ”VP_Laskenta” asettaa AND-komponentin toisen ehdon asentoon ”1” pulssin omaisesti. Tämän AND-komponentin toisena ehtona on vaaleanpunaisten kappaleiden

laskurin yläraja lähtösignaali ”UL”. Tämä lähtösignaalin arvo on käännetty NOT-komponentin avulla. Laskurin ylärajan lähtösignaali menee asentoon ”1”, kun tämä laskuri on onnistuneesti laskenut viisi vaaleanpunaista kappaletta. Tämä lähtösignaali on kytkettynä TON-ajastimeen, joka antaa viidennelle vaaleanpunaiselle kappaleelle aikaa mennä omalle alumiiniradalle. Kun viisi sekuntia on kulunut niin, laskurin yläraja lähtösignaali kääntää NOT-komponentin arvoa. Tämän laskurin resetoi vaaleanpunaisen radan kuittausnappi, joka sijaitsee päänäytöllä (ks. kuva 39). Tämän laskurin saa nollattua vain silloin, kun kuljettimella ei ole vaaleanpunaista kappaletta liikkeellä.

Kuvan 30 alapuolella nähdään viisi kappaletta Compare-komponenttia, jotka menevät arvoissa 0–4. Näissä Compare-komponenteissa verrataan laskurin arvoa yhdistimestä ”VP Laskuritieto”. Kun nämä Compare-komponenttien lähtösignaaliarvot ovat asennossa ”1”, nämä päästävät signaalitietonsa omiin yhdistimiin ”X. Vaaleanpunainen KPL”. Nämä yhdistimet jatkuvat työtilassa ”Alumiiniradan kappaleet VP” (vrt. kuva 33), joka käydään tarkemmin läpi luvussa 6.3.

Kuva 30. Vaaleanpunaisten kappaleiden laskuri



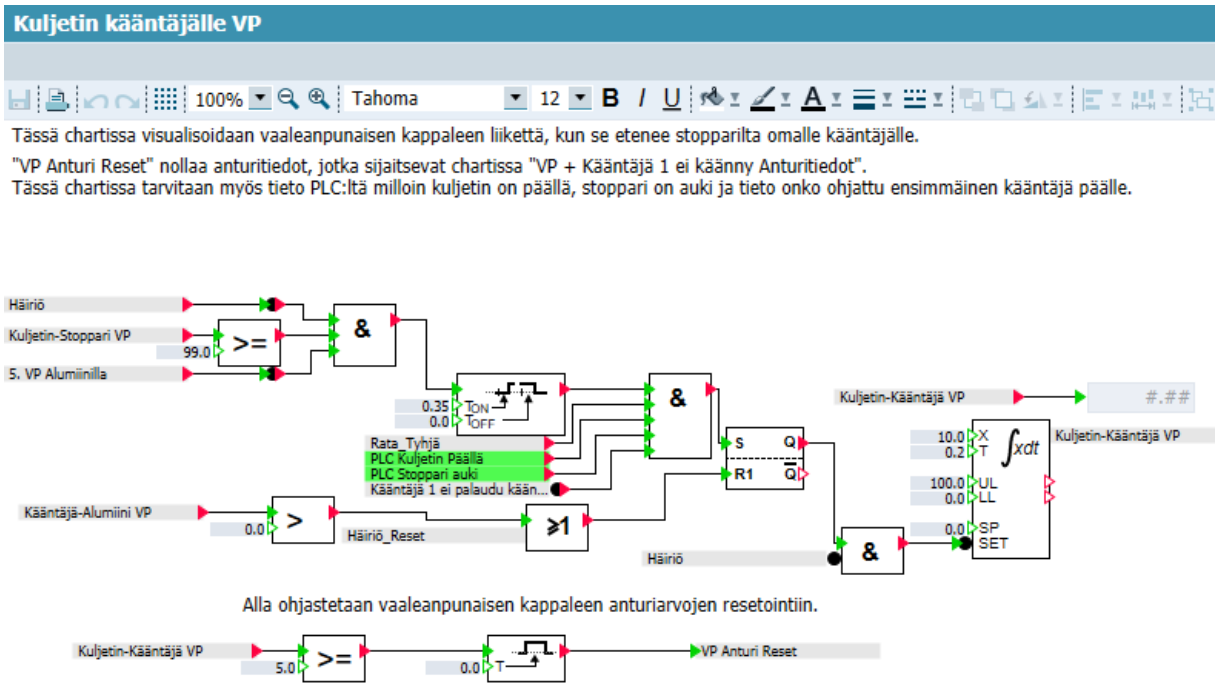
6.2 Stoppari-kääntäjän väli

Kuvan 31 työtilassa mallinnetaan vaaleanpunaisen kappaleen liikettä, kun se etenee stopparilta omalle kääntäjälle. Tässä vaiheessa tulee ensimmäiset eroavaisuudet kappaleiden välillä, ja ne eroavaisuudet ovat integraatiokomponenteissa ja niiden arvoissa. Muistetaan integraatiokomponenttien laskevan kappaleiden paikkaa kuljettimella ja näillä kappaleilla on omat kääntäjät määränpäinä. Esimerkiksi tämän vaaleanpunaisen kappaleen integraatioarvot ovat X= 10 ja T= 0.2, mustan kappaleen integraatioarvot ovat X= 6.66 ja T= 0.2, ja metallisen kappaleen integraatioarvot ovat X= 5 ja T= 0.2. Näistä eroavaisuuksista riippumatta, nämä työtilat on suunniteltu samankaltaisiksi vaaleanpunaisille, mustille ja metallisille kappaleille.

Kuvassa 31 nähdään yhdistin ”Kuljetin-Stoppari VP”, joka on aikaisemman työtilan ”Kuljetin Stopparille VP” integraatiokomponentin yhdistin. Tämä mallinnus alkaa siitä, kun kyseinen integraatiokomponentti on saavuttanut arvon ”99”. Kun integraatiokomponentti on saavuttanut arvon ”99”, Compare-komponentti päästää signaalin läpi AND-komponentin tulosignaaliin. AND-komponentin muita muuttujia ovat ”Häiriö” yhdistin ja ”5. VP Alumiinilla”, jotka ovat käännettynä NotC-komponentin avulla. ”Häiriö” yhdistin on kuvasta 34 ja ”5. VP Alumiinilla” on kuvasta 32. Kun AND-komponentin ehdot toteutuvat, se asettaa TON-ajastimen tulosignaalin asentoon ”1”. Tämä TON-ajastin päästää signaalin läpi 350ms jälkeen. TON-ajastin on yhdistettynä suureen AND-komponenttiin, johon tulee monta yhdistintä. Nämä yhdistimet ovat ”Rata_Tyhjä”, tämän yhdistimen tarkoituksena on tarkastaa, onko loppupään kuljetin tyhjänä (vrt. kuva 43). PLC:n lähtösignaalit ”PLC Kuljetin Päällä” ja ”PLC Stoppari auki”. Alin yhdistin on ”Kääntäjä 1 ei palaudu kääntäjällä” (vrt. kuva 36). Tämä AND-komponentti yhdistyy RS-komponentin SET-tulosignaaliin. RS-komponentin nollauksena toimii tämän kappaleen kääntäjän integraatiokomponentti ja sen yhdistimen ”Kääntäjä-Alumiini VP” (vrt. kuva 32). Tämä yhdistin on yhdistynyt Compare-komponenttiin, joka päästää signaalin läpi OR-komponenttiin sitten, kun integraatiokomponentti saavuttaa suuremman arvon kuin ”0”. OR-komponentin toinen ehto on yhdistin ”Häiriö_Reset” (vrt. kuva 34). Kun RS-komponentin lähtösignaalin menee arvoon ”1”, se päästää signaalin läpi integraatiokomponenttiin. Jos on joko häiriötila päällä tai vaaleanpunainen kappale sijaitsee kääntäjällä, niin näin ei käy.

Kuvan 31 alapuolella on yksi ehto. Ehdossa asetetaan yhdistin ”VP Anturi Reset” asentoon ”1” pulssimaisesti, kun tämän työtilan integraatiokomponentti on saavuttanut minimiarvon ”5”.

Kuva 31. Vaaleanpunaisen kappaleen mallinnus stopparilta kääntäjälle



6.3 Kääntäjä-alumiiniradan väli

Tässä työtilassa mallinnetaan vaaleanpunaisen kappaleen liikettä kääntäjällä. Tässä työtilassa myös lasketaan vaaleanpunaisia kappaleita, jotka etenevät alumiiniselle radalle. Tämän laskurin arvoja verrataan stopparilla sijaitsevan laskurin kanssa (kuva 30) ja näitä arvoja käytetään työtilassa ”Alumiiniradan kappaleet VP” (kuva 33).

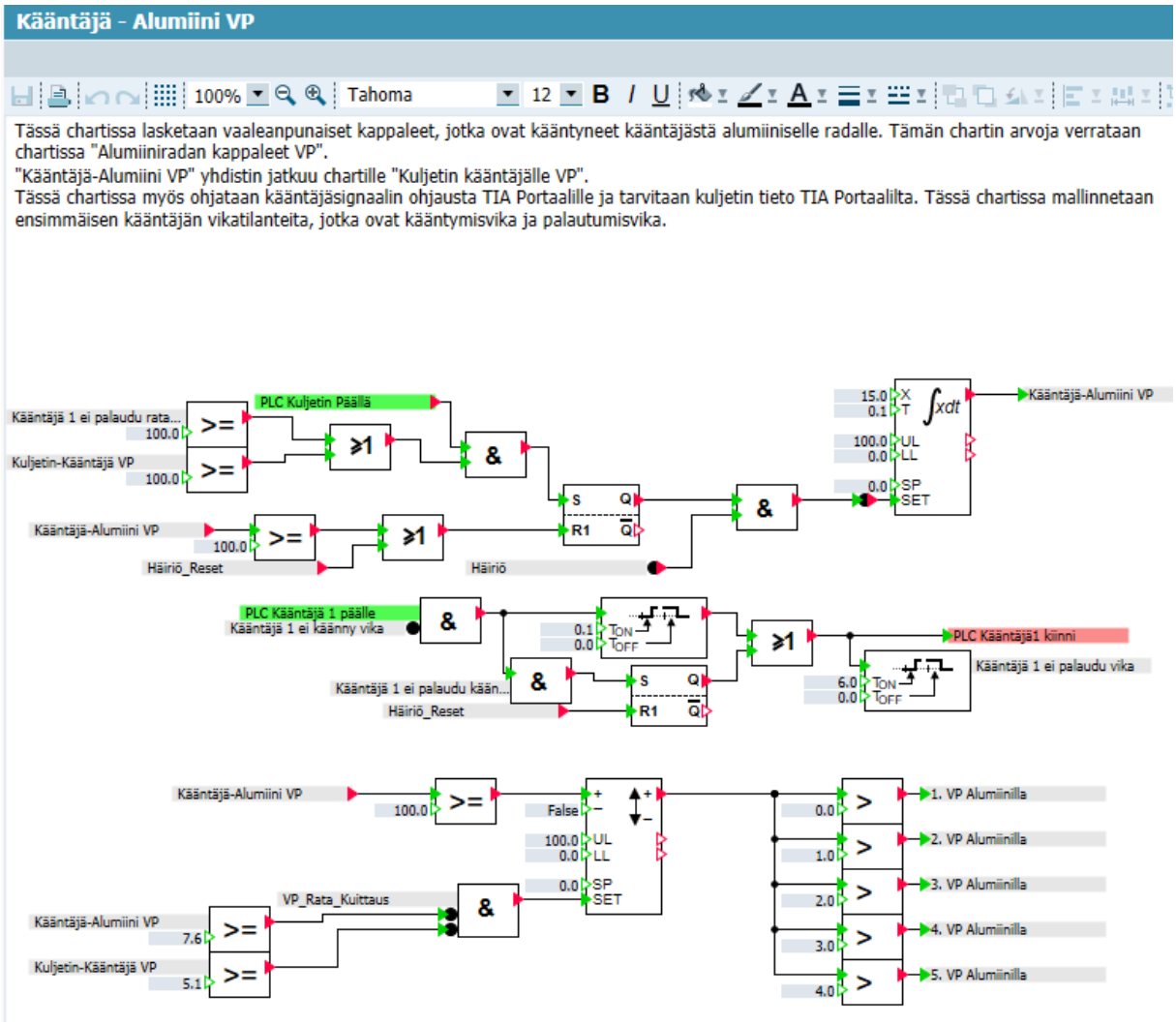
Tässä työtilassa (kuva 32) nähdään yhdistin ”Kuljetin-Kääntäjä VP”, jota käytettiin työtilassa ”Kuljetin kääntäjälle VP” (kuva 31). Tämä on yhdistettynä Compare-komponenttiin, joka päästää signaalin läpi vasta sitten, kun tämän yhdistimen integraatiokomponentti on saavuttanut arvon ”100”.”Kuljetin-Kääntäjä VP” yläpuolella on yhdistin ”Kääntäjä 1 ei palaudu rata tyhjä”, joka on työtilasta ”Kääntäjä 1 ei palaudu”. Tämä on myös yhdistettynä Compare-komponenttiin, joka päästää signaalin läpi, kun kyseisen yhdistimen integraatiokomponentti saavuttaa arvon ”100”. Nämä Compare-komponentit ovat yhdistettynä OR-komponentista AND-komponenttiin. AND-komponentin toisena ehtona on yhdistin ”PLC Kuljetin Päällä”. AND-komponentin lähtösignaali yhdistyy RS-komponentin SET-tulosignaaliin. RS-komponentin lähtösignaali on yhdistettynä toiseen AND-komponenttiin. AND-komponentin toisena ehtona on ”Häiriö” yhdistin, jonka arvo on käännetty NotC-komponentilla. AND-komponentti on yhdistettyä integraatiokomponenttiin, joka aloittaa vaaleanpunaisen

kappaleen paikan laskemisen kääntäjällä. RS-komponentin resetointiin tarkoitetut tulosignaalit ovat OR-komponentin tulosignaalit, jotka ovat joko kuvan 32 integraatiokomponentin arvo ”100” tai ”Häiriö_Reset”. Vaaleanpunainen kappale on kääntynyt alumiiniradalle kun ”Kääntäjä-Alumiini VP” yhdistimen integraatiokomponentti on saavuttanut arvon ”100”.

Kuvassa 32 nähdään keskellä työtilaa ensimmäisen kääntäjän mallinnukset. Mallinnukset ovat normaali toiminta, ensimmäinen kääntäjä ei käänny tai ei palaudu. AND-komponenttiin yhdistyy kaksi yhdistintä. Nämä yhdistimet ovat PLC:n lähtösignaali ”PLC Kääntäjä 1 päälle” ja ”Kääntäjä 1 ei käänny vika”. ”Kääntäjä 1 ei käänny vika” arvo on käännetty NotC-komponentilla. Esimerkiksi jos käyttäjä valitsee vikatilanteen, missä ensimmäinen kääntäjä ei käänny, niin yhdistin ”Kääntäjä 1 ei käänny vika” menee asentoon ”1”. Tämä ei päästä AND-komponentista signaalia eteenpäin. Jos käyttäjä valitsee normaalin toiminnan, niin silloin signaali menee AND-komponentista TON-ajastimeen. TON-ajastin päästää signaalin läpi 100ms ajan jälkeen OR-komponentin kautta PLC:n tulosignaalille ”PLC Kääntäjä1 kiinni”. Jos käyttäjä valitsee vikatilanteen, missä ensimmäinen kääntäjä ei palaudu, niin silloin ensimmäisestä AND-komponentista menee signaali alempaan AND-komponenttiin. Alemman AND-komponentin alempi yhdistin on ”Kääntäjä 1 ei palaudu kääntäjällä”. Alempi AND-komponentti on yhdistettynä RS-komponentin SET-tulosignaaliin, joka on yhdistettynä samaan OR-komponenttiin. OR-komponentti on yhdistettynä toiseen TON-ajastimeen, joka päästää häiriösignaalin läpi 6 sekunnin jälkeen. Tämä toteutuu vain, jos käyttäjä on valinnut kyseisen vikatilanteen.

Kuvan 32 alapuolella on laskuri, joka laskee vaaleanpunaisia kappaleita, jotka ovat kääntyneet alumiiniradalle. Kun kääntäjän integraatiokomponentin yhdistin ”Kääntäjä-Alumiini VP” on saavuttanut arvon ”100”, tämä laskuri nostaa arvoaan yhdellä. Tämä laskuri toimii samalla tavalla kuin aikaisempi laskuri, joka sijaitsee stopparilla. Tästä laskurista lähtee laskuritiedot Compare-komponentteihin, jotka tarkastavat laskurin arvon 0–4 väliltä ja sen mukaan asettavat yhdistimet ”X. VP Alumiinilla” asentoon ”1”. Tämän laskurin nollauksena toimii vaaleanpunaisten kappaleiden alumiiniradan kuittausnappi, joka sijaitsee päänäytöllä (kuva 39).

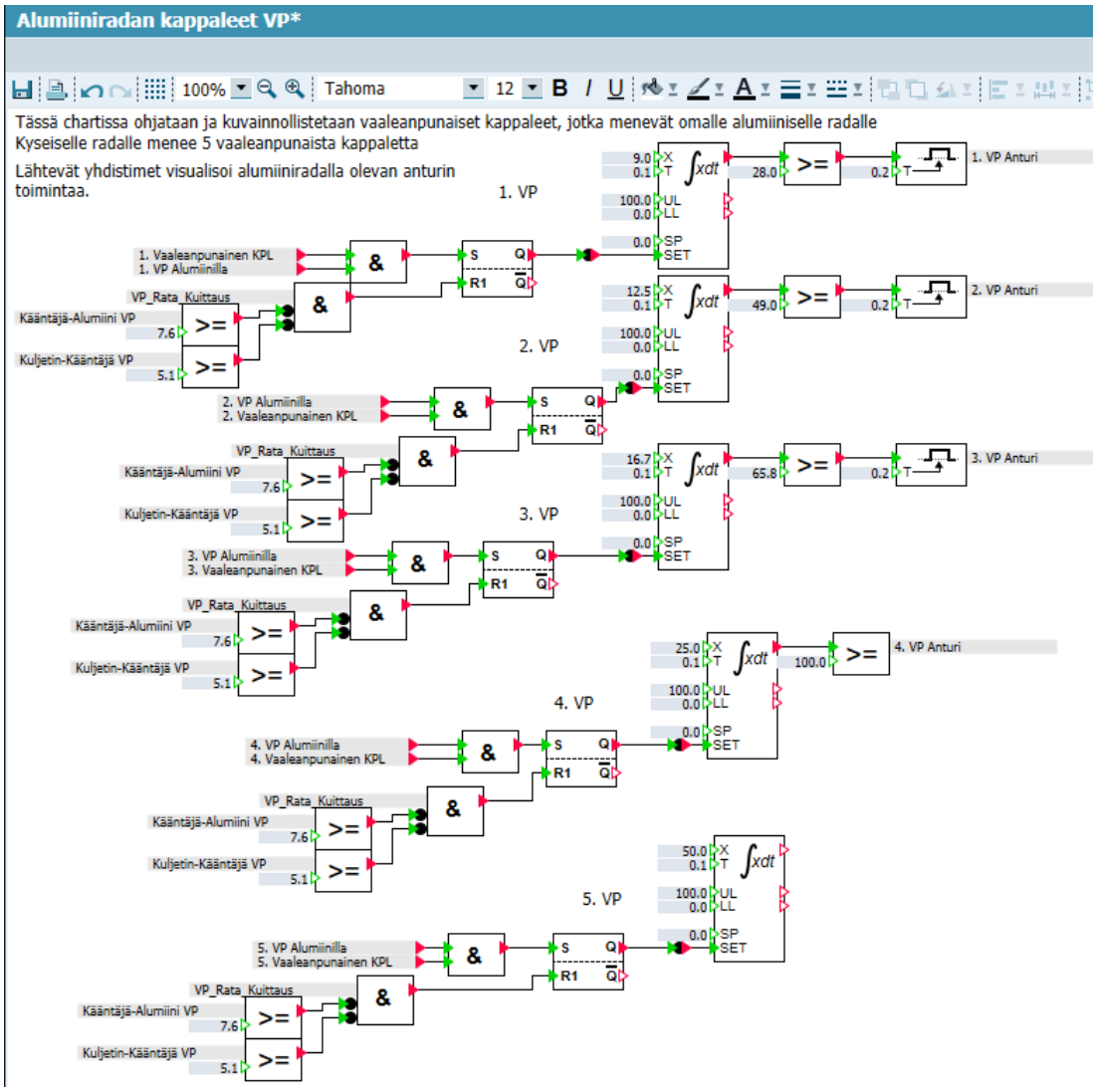
Kuva 32. Vaaleanpunaisen kappaleen mallinnus kääntäjällä



Kuvassa 33 on työtila, jossa mallinnetaan viiden vaaleanpunaisten kappaleen liikettä ja sijaintia alumiiniradalla. Jokaisella vaaleanpunaisen kappaleen integraatiokomponentin toiminta on aivan samanlainen arvoista riippumatta. Tämä johtuu siitä koska jokaisella vaaleanpunaisen kappaleen integraatiokomponentti tarvitsee laskuri arvot työtiloista ”Counter VP” ja ”Kääntäjä – Alumiini VP”. Esimerkiksi ensimmäinen vaaleanpunainen kappale alumiiniradan integraatiokomponentti tarvitsee AND-komponentin ehtojen ”1. Vaaleanpunainen KPL” ja ”1. VP Alumiinilla” täyttyvän. Tämä tyyli toistuu jokaisella eri kappaleella, kappaleen 1–5 lukumäärästä riippumatta. Tässä työtilassa ensimmäisten kolmen vaaleanpunaisten kappaleiden integrointikomponenteista lähtevät signaalit yhdistyvät Compare-komponentteihin, jotka yhdistyvät ”TP”-ajastimeen. TP-ajastin asettaa lähtevän yhdistimen asentoon ”1” pulssimaisesti 200ms ajaksi. Kun neljäs kappale on saavuttanut integraatioarvon ”100”, se asettaa yhdistimen ”4. VP Anturi” asentoon ”1”. Nämä lähtevät yhdistimet jatkavat työtilaan ”Visualisointiin tarvittavat” (kuva 37), jossa nämä yhdistimet

visualisoivat alumiiniradan anturin toimivuutta. Nämä integraatiokomponentit käyttäjä asettaa nollaksi painamalla päänäytöstä (kuva 39) nappia ”VP_Rata_Kuittaus”, kun kuljettimella ei ole vaaleanpunaista kappaletta.

Kuva 33. Vaaleanpunaisten kappaleiden mallinnus alumiiniradalla

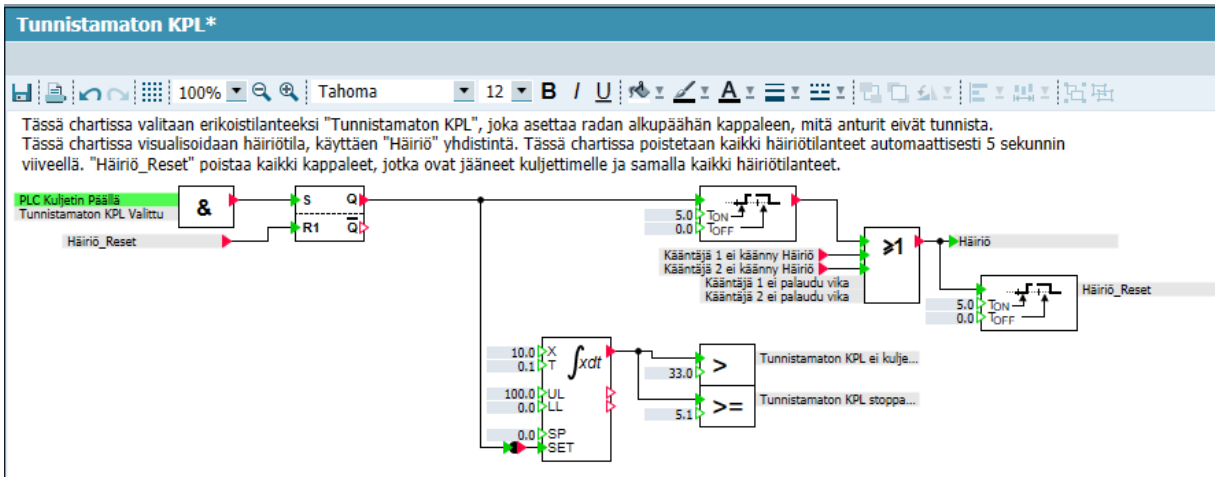


6.4 Erikoistilanteiden simulointi

Tässä alaluvussa käydään läpi kolmea erilaista vikatilannetta, mitkä voisivat tapahtua oikealla jakaja-asemalla, jossakin oikeassa tehtaassa. Nämä vikatilanteet ovat tunnistamaton kappale, ensimmäinen kääntäjä ei käänny tai palaudu. Kääntäjäviat pätevät myös toiselle kääntäjälle. Tunnistamaton kappale vika voi tulla silloin, jos kuljettimelle pääsee kappale minkä ei pitäisi päästä jakaja-asemalle. Kääntäjäviat voivat tulla pitkäaikaisen käytön jälkeen, jolloin palautinjousi pettää tai kääntäjän sylinteri on kulunut.

Kuvassa 34 nähdään työtila, jossa mallinnetaan tunnistamattoman kappaleen liikettä ja häiriötilan aktivointia. Käyttäjä voi aloittaa tunnistamattoman kappaleen skenaarion painamalla nappia ”Unidentified object” päänäytöltä (kuva 39). Jos käyttäjä valitsee tunnistamattoman kappaleen, se asettaa tunnistamattoman kappaleen kuljettimen alkupäähän. Tällöin kuljetin menee päälle ja ensimmäisen AND-komponentin ehdot toteutuvat. AND-komponentista lähtee signaali RS-komponentin SET-tulosignaalin, ja RS-komponentin lähtösignaali asettuu asentoon ”1”, jos ei ole häiriötilaa päällä. RS-komponentti on yhdistettynä tunnistamattoman kappaleen integraatiokomponenttiin, joka laskee tunnistamattoman kappaleen paikkaa. RS-komponentti on myös yhdistettynä TON-ajastimeen, joka päästää signaalin läpi viiden sekunnin jälkeen. viiden sekunnin jälkeen häiriötila menee aktiiviseksi OR-komponentin kautta. Häiriötila voi mennä aktiiviseksi myös, jos käyttäjä valitsee jonkun kääntäjävian. Häiriöt nollaantuvat automaattisesti 5 sekunnin kuluessa.

Kuva 34. Tunnistamaton kappale



Kuvissa 35 ja 36 nähdään ensimmäisen kääntäjän vikatilanteiden mallinnusta. Näissä mallinnetaan samalla tavalla vaaleanpunaisen kappaleen liikettä kuljettimen alkupäästä stopparille, ja stopparilta kääntäjälle. Näissä mallinnuksissa kääntäjä ei käänny (kuva 35) ja ei palaudu (kuva 36). Kuten kuvista 35 ja 36 nähdään, nämä vikatilanteet saadaan mallinnettua varsin samalla tavalla. Ainoat asiat mitkä muuttuvat ovat yhdistimien nimet ja alemman integraatiokomponentin arvot. Alemman integraatiokomponentin arvot eroavat toisistaan, koska nämä kappaleet siirtyvät eri pituisen matkan.

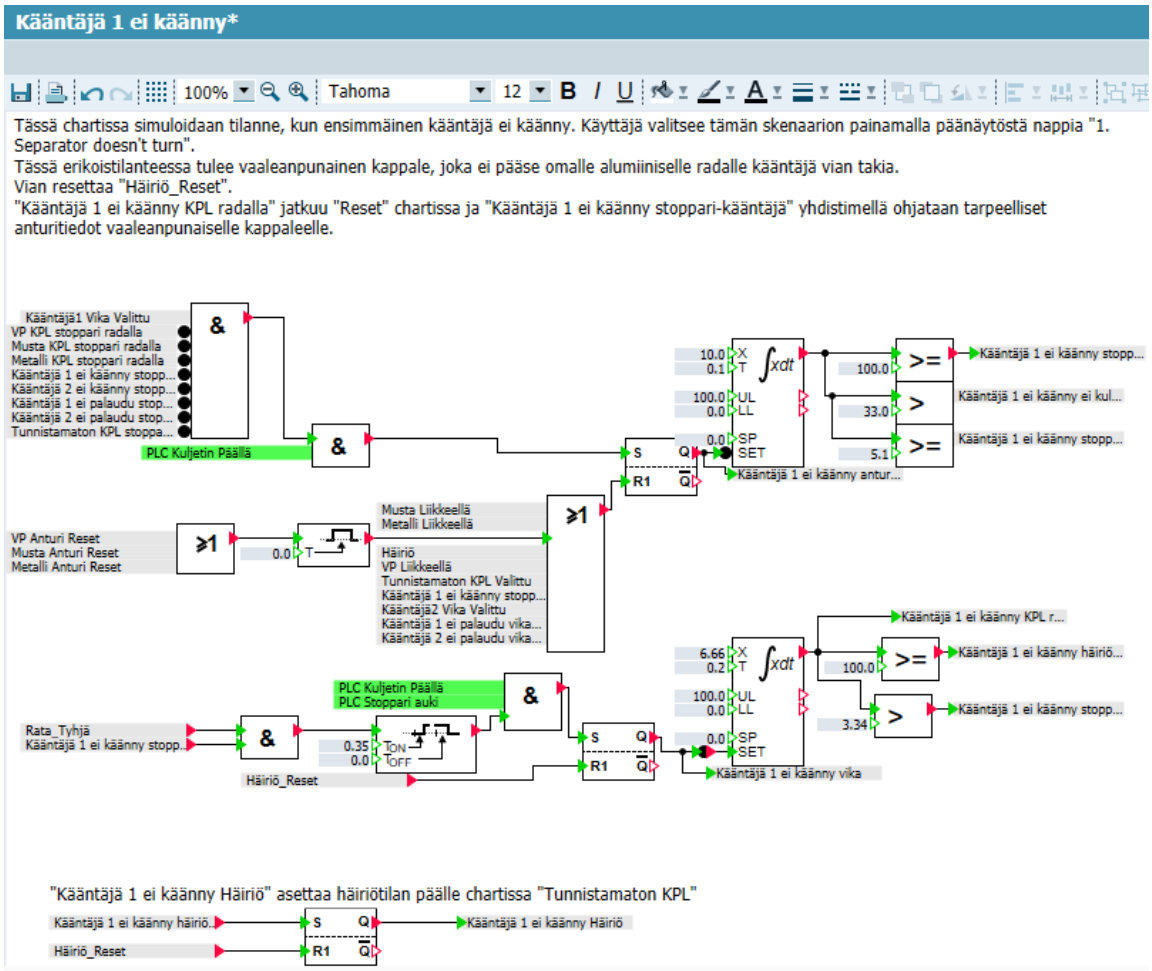
Kuvan 35 mallinnus alkaa siitä, kun käyttäjä painaa nappia ”1. Separator doesn’t turn” päänäytöltä (kuva 39). Kun käyttäjä on valinnut ensimmäisen kääntäjän vikatilanteen, se

asettaa vaaleanpunaisen kappaleen kuljettimen alkupäähän. Mutta jos käyttäjä on aloittanut jonkin muun skenaarion, niin tämä ei toteudu. Suuresta AND-komponentista lähtevä signaali menee seuraavaan AND-komponenttiin missä tarkastellaan, onko kuljetin asennossa ”1”. Toisesta AND-komponentista lähtevä lähtösignaali yhdistyy RS-komponentin SET-tulosignaalin. RS-komponentti päästää signaalin läpi integraatiokomponentille. Integraatiokomponentti laskee vaaleanpunaisen kappaleen liikettä kuljettimen alkupäässä, kunhan ei ole mikään nollaamassa kyseistä RS-komponenttia. RS-komponentin nollauksena toimii samalla tyylillä, kuin kuvan 26 nollaukset. Nämä ovat kappaleiden liike kuljettimen alkupäässä ja kappaleiden anturi resetit. Ennen integraatiokomponenttia löydetään yhdistin ”Kääntäjä 1 ei käänny anturitiedot”. Tämä yhdistin yhdistyy työtilaan ”VP + Kääntäjä 1 ei käänny ja palaudu anturitiedot” (kuva 27).

Tästä integraatiokomponentista tehdään toimenpiteitä kolmesta eri arvoista. Kun integraatiokomponentti saavuttaa minimiarvon ”5.1”, se asettaa yhdistimen ”Kääntäjä 1 ei käänny stoppari radalla” asentoon ”1”. Tämä yhdistin estää käyttäjää käynnistämästä muita skenaarioita. Kun integraatiokomponentti saavuttaa arvon ”33”, se asettaa yhdistimen ”Kääntäjä 1 ei käänny ei kuljettimen alussa” asentoon ”1”. Tämä yhdistin on yhdistettynä ”Visualisointiin tarvittavat” työtilassa (kuva 37). Kun tämä integraatiokomponentti on saavuttanut arvon ”100”, tämä asettaa yhdistimen ”Kääntäjä 1 ei käänny stopparilla” asentoon ”1”. Tämä yhdistin on yhdistettynä AND-komponentin tulosignaaliin, ja tämän AND-komponentin toisena ehtona on ”Rata_Tyhjä” yhdistin (kuva 43). Kun AND-komponentin ehdot täyttyvät, AND-komponentti päästää signaalin läpi TON-ajastimelle, joka päästää signaalin läpi 350ms jälkeen. Tämä ajastin kuvaa sitä hetkeä, kun kappale on stopparilla. TON-ajastin on yhdistynyt AND-komponenttiin, jossa on kaksi lisäehtoa. Muut ehdot tulevat TIA Portaalista ja ne ovat ”PLC Kuljetin päällä” ja ”PLC Stoppari auki”. AND-komponentin lähtösignaali yhdistyy RS-komponentin SET-tulosignaaliin. RS-komponentin nollauksen tulosignaaliin on yhdistynyt yhdistin ”Häiriö_Reset”, joka on häiriöiden nollaukseen tarkoitettu. RS-komponentin lähtösignaali on yhdistynyt integraatiokomponenttiin, joka laskee vaaleanpunaisen kappaleen paikkaa stopparin jälkeisellä kuljettimella. RS-komponentti on myös yhdistynyt yhdistimeen ”Kääntäjä 1 ei käänny vika”. Tämä yhdistin on yhdistynyt työtilaan ”Kääntäjä – Alumiini VP” (kuva 32). Tästä integraatiokomponentista lähtee kolme eri ehtoa eteenpäin. Ylin ehto ”Kääntäjä 1 ei käänny KPL radalla” menee ”Reset” työtilaan, jossa tämä asettaa ”Rata_Tyhjä” yhdistimen asentoon ”0”. Toiseksi ylimmässä ehdossa halutaan saada integraatiokomponentin arvoksi ”100”, jolloin yhdistin ”Kääntäjä 1 ei käänny häiriöhetki” menee asentoon ”1”. Tämä ehto asettaa samaisessa työtilassa alimman RS-komponentin SET-tulosignaalin asentoon ”1”. Kun RS-komponentin lähtösignaali on arvossa ”1”, yhdistin ”Kääntäjä 1 ei käänny Häiriö” ilmoittaa häiriötilan olevan aktiivisena. Alin ehto

haluaa saada integraatiokomponentin minimiarvoksi ”3.34”, jolloin tämä nollaa anturitiedot (kuva 27). Tämän häiriö kuittaantuu ”Häiriö_Reset” yhdistimestä 5 sekunnin kuluessa. Toisen kääntäjän vikatilanne, missä kääntäjä ei käänny on mallinnettu samalla tavalla.

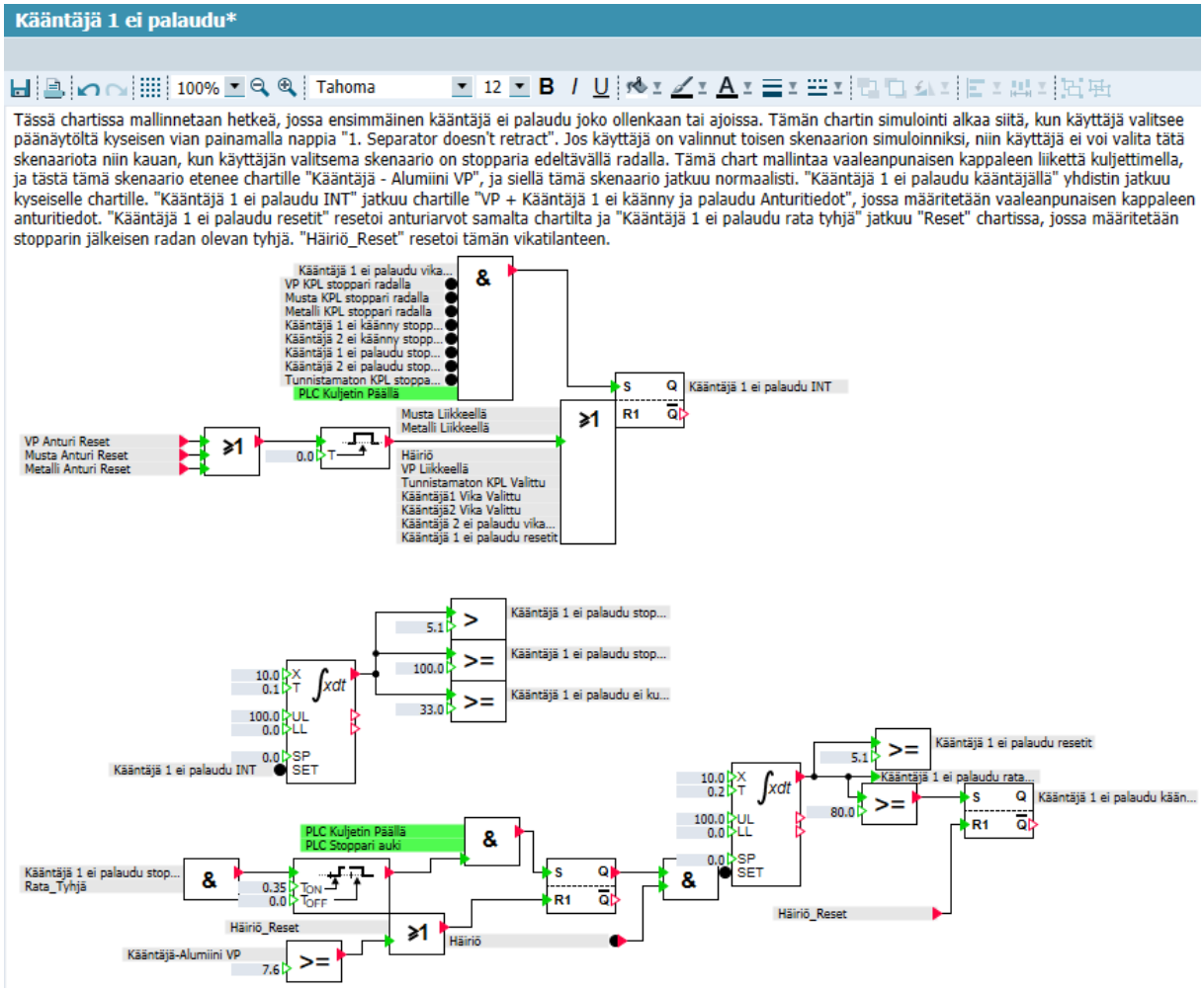
Kuva 35. Ensimmäinen kääntäjä ei käänny



Kuvassa 36 on työtila, jossa mallinnetaan ensimmäisen kääntäjän palautumisvian hetkeä, missä kappale on kuljettimella ennen kääntäjää. Kuten voidaan verrata kuvaa 35 kuvaan 36:een, nähdään näiden työtilojen olevan hyvin samanlaisia. Ainoat eroavaisuudet ovat yhdistimien nimet ja joidenkin yhdistimien käyttötarkoitukset. Muistetaan yhdistimen ”Kääntäjä 1 ei palaudu INT” menevän työtilaan ”VP + Kääntäjä 1 ei käänny ja palaudu” (kuva 27). Kyseisessä työtilassa tämä yhdistin asettaa vaaleanpunaisen kappaleen anturiarvot. Kuvan 36 alemman integraatiokomponentista lähtee yhdistin ”Kääntäjä 1 ei palaudu rata tyhjä” työtilaan ”Kääntäjä - Alumiini VP” (kuva 32). Kyseisessä työtilassa tämä skenaario jatkuisi normaaliin tapaan. Toinen yhdistin, joka lähtee alemmasta RS-komponentista, kun integraatiokomponentin arvo on minimissään ”80” on ”Kääntäjä 1 ei palaudu kääntäjällä”. Tämä yhdistin toimii työtilassa ”Kääntäjä - Alumiini VP” (kuva 32) kyseisen häiriötilan

laukaisijana. Tämä yhdistin estää myös muiden kappaleiden etenemisen stopparilta kääntäjille, jos tämä on aktiivisena. Alemman integraatiokomponentin nollauksena toimii ”Häiriö” ja ”Häiriö_Reset” yhdistimet sekä yhdistin ”Kääntäjä-Alumiini VP” (kuva 32), kun kyseisen yhdistimen integraatiokomponentti saavuttaa minimiarvon ”7.6”.

Kuva 36. Ensimmäinen kääntäjä ei palaudu



6.5 Anturien ja alkuradan kappaleiden visualisointi

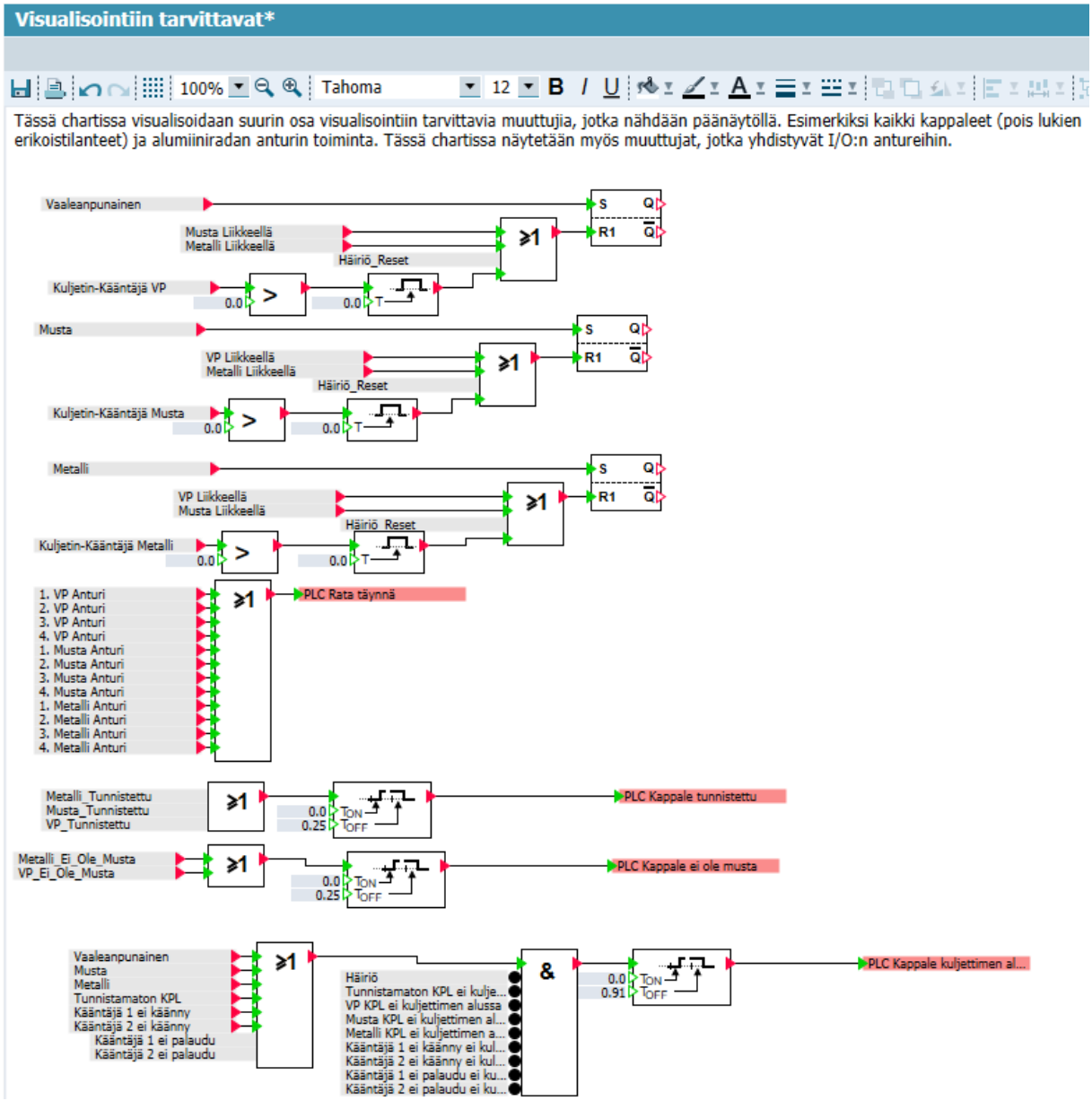
Tässä alaluvussa käydään läpi työtilat, joissa on visualisoitu anturien paikat, kappaleet alkuradalla ja kuljettimen toiminta. Tässä alaluvussa esitetään päänäyttö ja käydään läpi pari esimerkkikohtaa simuloinnista, kuten häiriötilanne ja tilanne missä alumiininen rata on täynnä. Virtuaalimallin visuaaliset tavoitteet olivat mahdollisimman realistinen virtuaalimalli jakaja-asemasta, antureiden, kappaleiden sekä toimilaitteiden visualisointi.

6.5.1 Komponentit virtuaalimallin visualisointiin

Kuvassa 37 nähdään työtila, johon on mallinnettu suurimman osan visualisoinneista tästä projektista. Kun käyttäjä valitsee päänäytöltä (kuva 39) jonkin kappaleen skenaarioksi. Täältä otetaan jonkin RS-komponentti, joka visualisoi kyseisen kappaleen liikettä kuljettimen alkupäässä päänäytöllä. Esimerkiksi jos käyttäjä valitsee vaaleanpunaisen kappaleen, niin silloin menee ylimmän RS-komponentin SET-tulosignaali arvoon ”1”. Tällöin lähtösignaali menee arvoon ”1”, joka piirtää vaaleanpunaisen kappaleen päänäytölle. Tämä tapahtuu, jos RS-komponenttia ei nollaa ennen minkään muun kappaleen liike kuten ”Musta Liikkeellä” (kuva 42). Tai jos vaaleanpunainen kappale on edennyt stopparilta eteenpäin olevalle kuljettimelle. Kyseisen kappaleen integrointikomponentin (kuva 31) arvo on suurempi kuin ”0”, niin se nollaa kyseisen RS-komponentin pulssimaisesti. Sama toistuu niin mustalle kappaleelle kuin metalliselle kappaleelle.

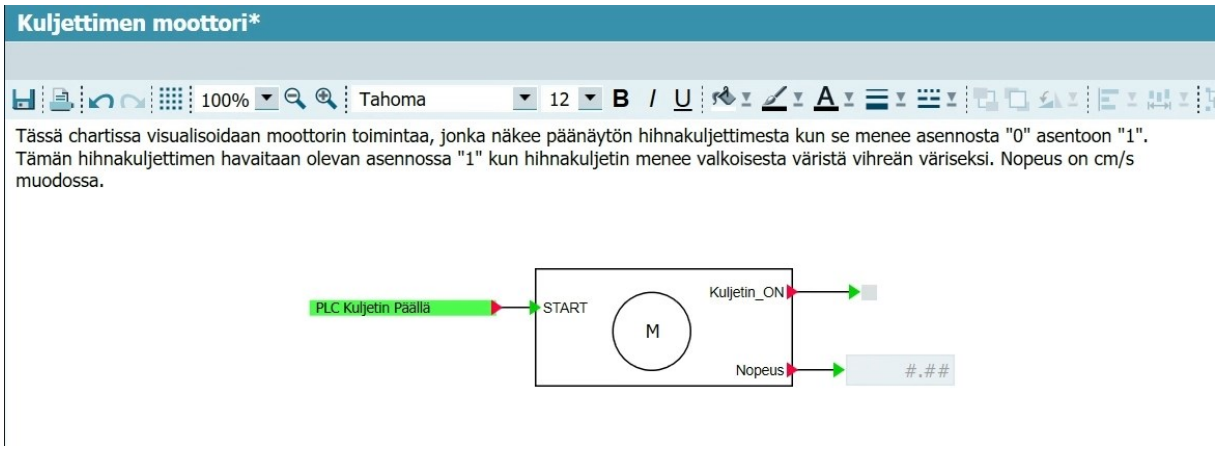
Kuvassa 37 nähdään myös anturitietojen ja -paikkojen mallinnusta. Ainoa puuttuva anturitiedon ja -paikan mallinnus on TIA Portaalin tulosignaali ”PLC Kappale on metallinen”, joka löytyy työtilasta ”Metalli Anturitiedot” (kuva 29). Jokaisella lähtevillä PLC signaaleilla on omat ehtonsa ja näiden ehtojen yhdistimet tulevat aikaisemmista työtiloista. Yhdistin ”PLC Rata täynnä” menee asentoon ”1”, kun esimerkiksi ensimmäinen vaaleanpunainen kappale on edennyt alumiiniselle radalle (kuva 33). Yhdistimet ”PLC Kappale tunnistettu” ja ”PLC Kappale ei ole musta” menevät asentoon ”1”, kun kappaleen anturitiedot ovat annettu aikaisemmilta työtiloilta. Yhdistin ”PLC Kappale kuljettimen alussa” menee asentoon ”1”, kun käyttäjä aloittaa jonkin skenaarion. Yhdistin ”PLC Kappale kuljettimen alussa” pysyy asennossa ”0”, jos jokin kappale on jo valmiiksi jo stopparia edeltävällä radalla. Jokaisella yhdistimellä on oma TOF-ajastin, joiden tarkoituksena on simuloida hetkeä, milloin kappaleet ovat edenneet anturin edestä pois.

Kuva 37. Visualisoinnin työtila



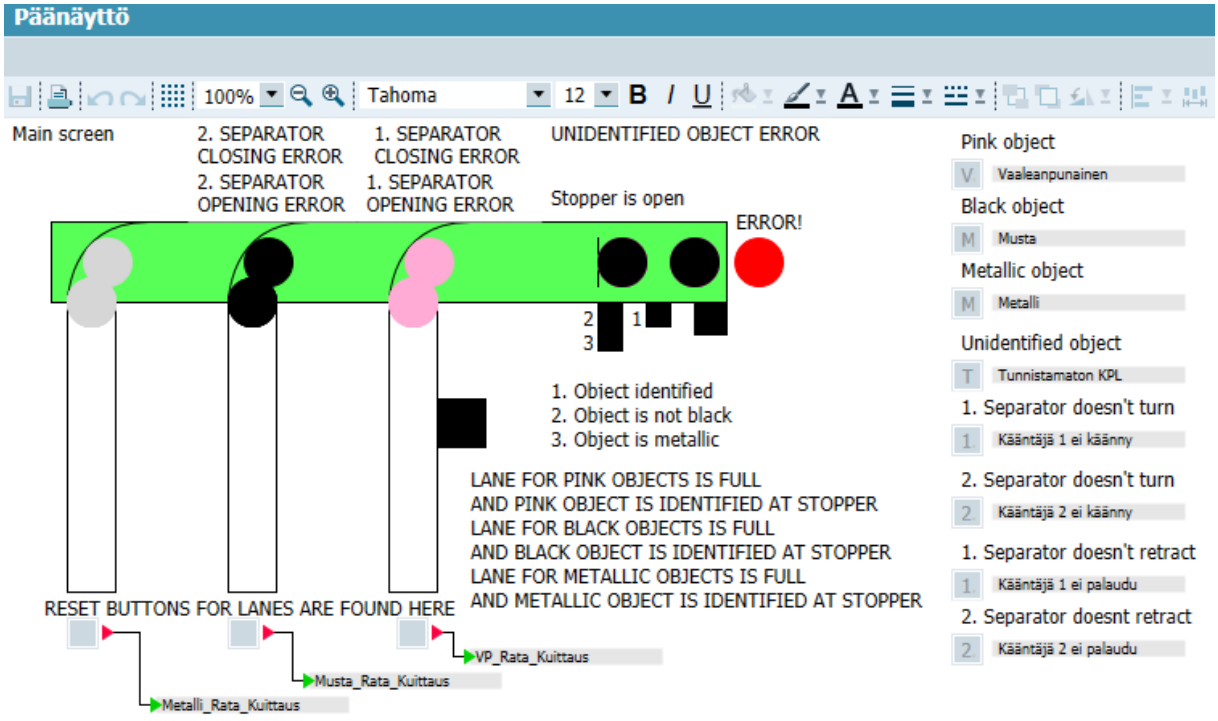
Kuvassa 38 nähdään kuljettimen visuaalinen mallinnus, jossa mallinnetaan kuljettimen tilaa päänäytöllä kuvassa 39. Moottori menee asentoon ”1”, kun TIA Portaalista tulee käsky laittaa lähtösignaali ”PLC Kuljetin Päällä” asentoon ”1”. Kuvasta 38 nähdään myös moottorin simuloitu nopeus (cm/s) ja kuljettimen tila.

Kuva 38. Kuljettimen moottori



Kuvassa 39 nähdään päänäytön ulkonäkö silloin, kun työ on offline-tilassa. Päänäytöltä käyttäjä näkee kaiken tarvittavan, kuten napit mistä saa aloitettua skenaariot, napit mistä saa nollattua alumiiniradat. Käyttäjä voi seurata kaikkia oleellisia tietoja päänäytöltä. Esimerkiksi anturitietoja, stopparin, kääntäjien ja kuljettimen tilatietoja. Käyttäjä voi myös seurata mikä kappale on kuljettimella, kuinka monta kappaletta on kuljettimella ja alumiiniradoilla. Myös nähdään häiriötila sekä mikä häiriö on kyseessä. Käyttäjä voi myös lukea, onko jokin rata täynnä ja onko samanlainen kappale tunnistettuna stopparilla.

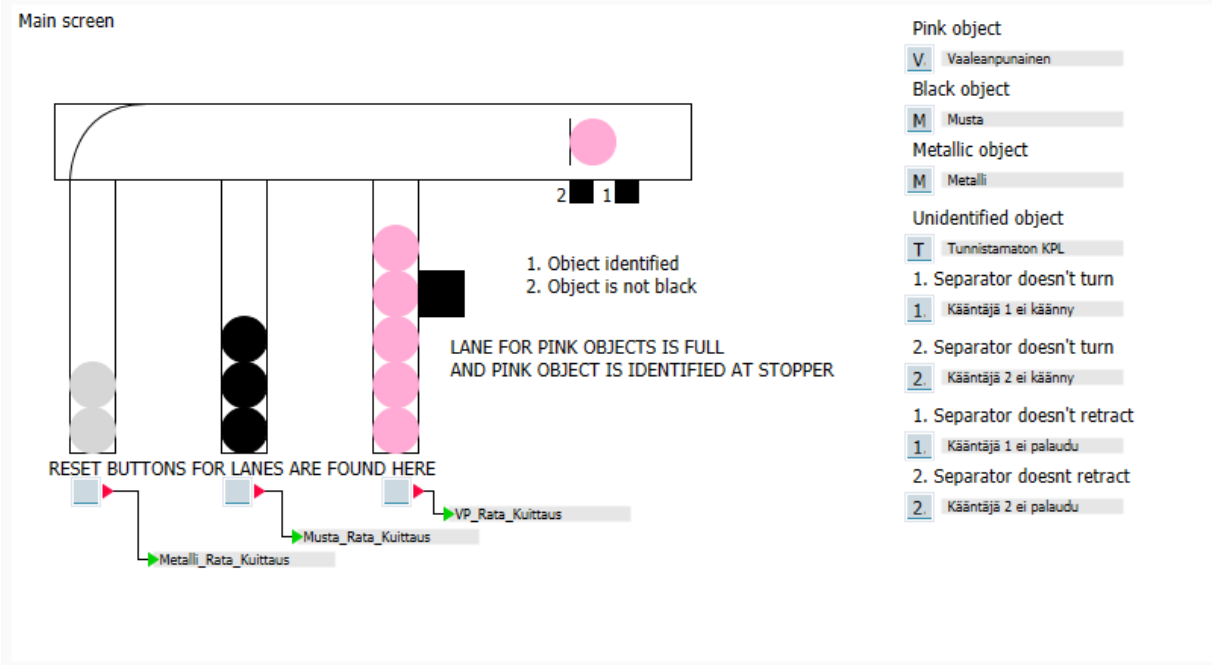
Kuva 39. Päänäyttö offline tilassa



6.5.2 Virtuaalimallin esimerkkitilanteet

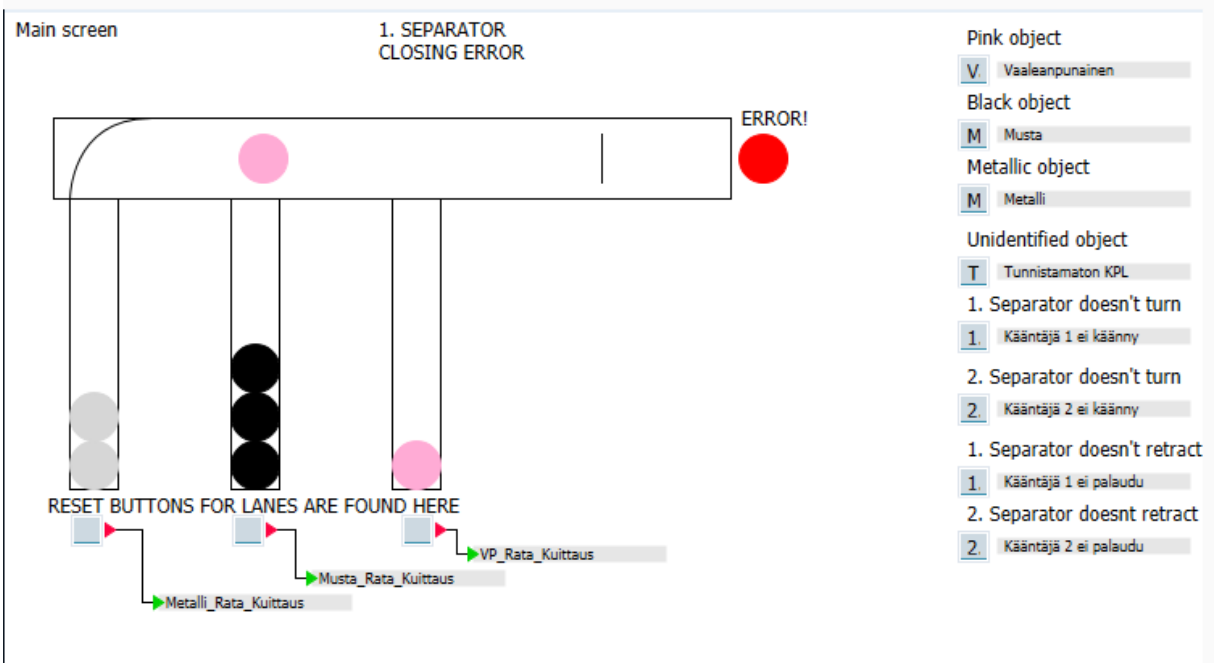
Kuvassa 40 nähdään ensimmäinen esimerkkitilanne, mihin tällä jakaja-asemalla voi joutua. Tässä esimerkkitilanteessa alumiinisilla radoilla on kaksi harmaata kappaletta, kolme mustaa kappaletta ja viisi vaaleanpunaista kappaletta. Samalla havaitaan vaaleanpunaisen kappaleen olevan stopparilla tunnistettuna. Tästä eteenpäin, jos käyttäjä haluaisi jatkavan skenaariota, hän painaisi nappia, jonka yhdistimenä on ”VP_Rata_Kuittaus”. Tämä kuittaisi vaaleanpunaisten kappaleiden alumiiniradan kerätyksi ja päästäisi ylimääräisen vaaleanpunaisen kappaleen liikkeelle.

Kuva 40. Esimerkkitilanne 1



Kuvassa 41 on toinen esimerkkitilanne, mihin tämä jakaja-asema voi joutua käytön aikana. Tässä tilanteessa alumiinisilla radoilla on kaksi harmaata kappaletta, kolme mustaa kappaletta ja yksi vaaleanpunainen kappale. Käyttäjä on simuloinut tilanteen, jossa ensimmäinen kääntäjä ei käännä vaaleanpunaista kappaletta alumiiniselle radalle. Tästä syystä jakaja-asema menee häiriölle. Tästä eteenpäin käyttäjä joutuu odottamaan 5 sekuntia, jotta virtuaalimalli nollaisi tämän hälytyksen. Tämän jälkeen käyttäjä itse nollaa hälytyksen TIA Portaalin omasta käyttöliittymästä painamalla nappia ”Alarm reset”.

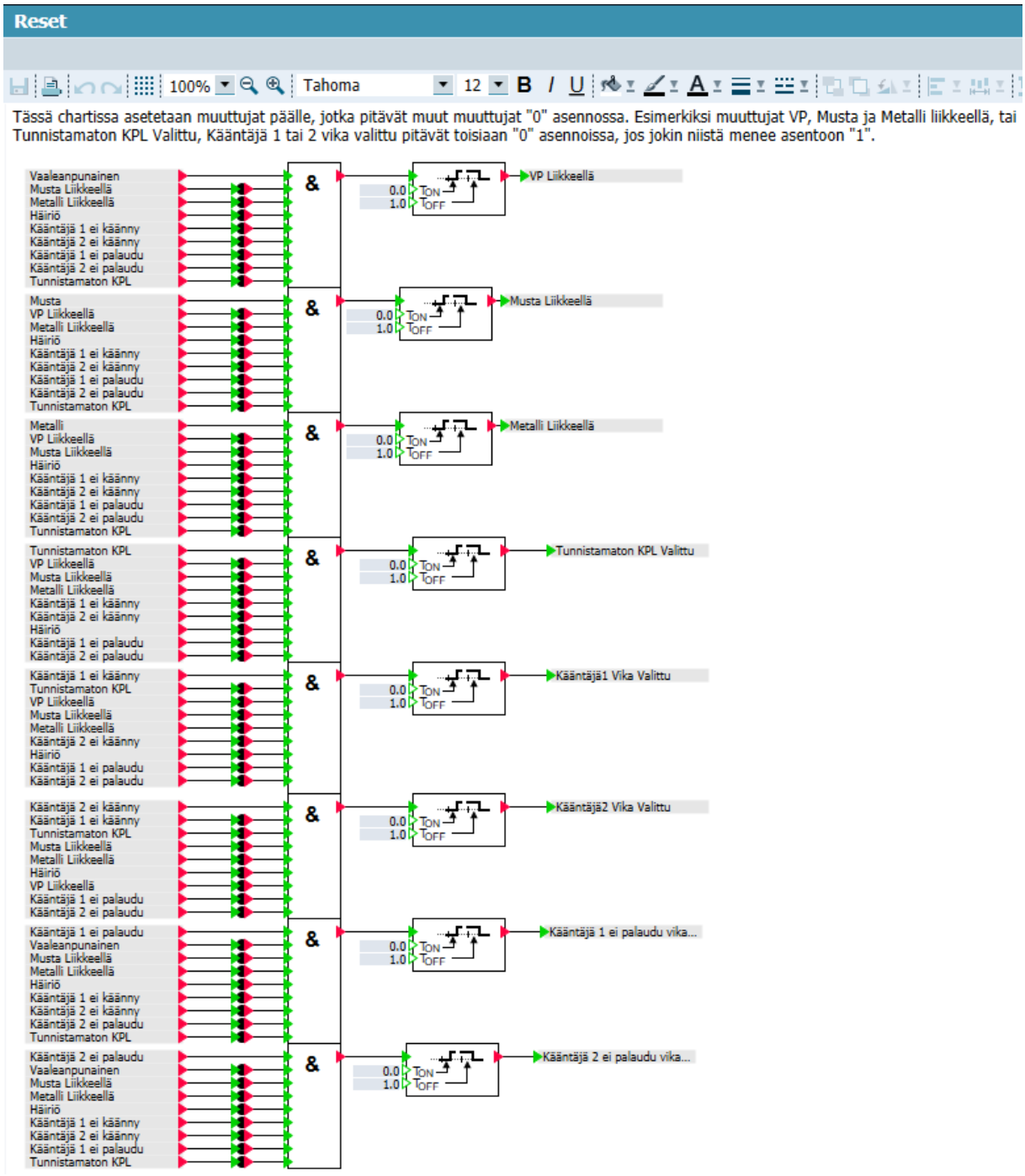
Kuva 41. Esimerkkitilanne 2



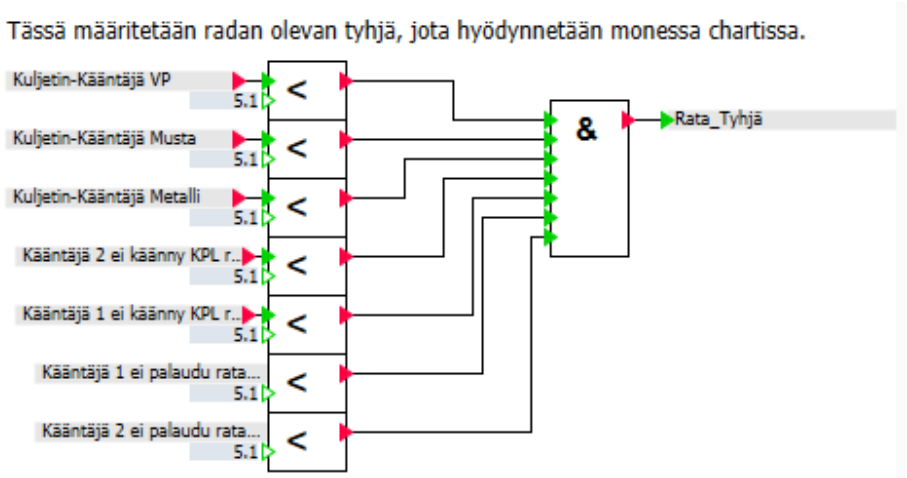
6.6 Resettien mallinnus

Kuvissa 42 ja 43 nähdään työtila, jossa asetetaan muuttujat päälle, jotka nollaavat ehtoja. Hyvinä esimerkkeinä ovat ”VP Liikkeellä”, ”Musta Liikkeellä”, ”Tunnistamaton KPL Valittu” ja ”Rata_Tyhjä”. Nämä lähtevät yhdistimet on nähty aikaisemmissa työtiloissa nollaamassa ehtoja, kuten RS-komponentteja tai AND-komponentteja. Kun käyttäjä valitsee skenaarioksi, esimerkiksi vaaleanpunaisen kappaleen, se asettaa TOF-ajastimelta yhdistimen ”VP Liikkeellä”. ”VP Liikkeellä” estää muiden AND-komponenttien aktivointia yhden sekunnin verran. ”Rata_Tyhjä” yhdistin menee asentoon ”1”, kun jokaisen kappaleen integraatiokomponentin arvo on alle ”5.1”. Jos jonkin integraatiokomponentin arvo on yli ”5.1”, se tarkoittaa kuljettimen loppupäässä olevan jokin kappale. Nämä integraatiokomponentit ovat stopparin ja kääntäjä väliltä, joista nähdään esimerkkimallinnus kuvasta 31.

Kuva 42. Reset työtila 1/2



Kuva 43. Reset työtila 2/2



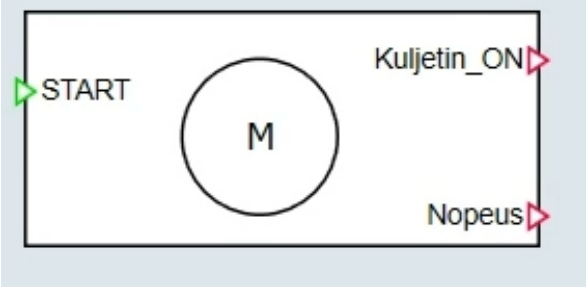
6.7 Käyttöohjeiden laatiminen SIMIT SP:stä

Yksi tavoite SIMIT SP:n projektiosuudesta oli luoda selkeät käyttöohjeet henkilöille, joilla ei ole yhtään kokemusta SIMIT SP -mallinnusohjelmistosta. Käyttöohjeet alkavat perusteellisesti ohjelmiston käynnistyksestä. Käyttöohjeissa käydään myös läpi SIMIT SP:n ja TIA Portaalin integrointia. Käyttöohjeet päättyvät virtuaalimallin esittelyyn ja ohjeistukseen mitä mikäkin työtila tekee ja miten ne linkittyvät toisiinsa. Liitteestä 3 nähdään SIMIT SP -käyttöohjeiden kansi.

6.8 SIMIT CTE

Kuvassa 44 nähdään komponentti minkä suunnittelin käyttäen SIMIT CTE -ohjelmistoa. Tämä komponentti kuvastaa jakaja-aseman kuljettimen moottoria, joka käynnistyy, kun tulosignaali ”START” menee asentoon ”1”. Tämä komponentti laskee sähkömoottorin nopeutta 0–8.25 cm/s. ”Kuljetin_ON” lähtösignaaliin voi kytkeä binäärisen valon, joka ilmoittaa moottorin olevan asennossa ”1”. ”Nopeus” lähtösignaaliin voi kytkeä digitaalisen mittarin, missä näytetään kuljettimen nopeus. Samalla tuli pohdittua miten muita jakaja-aseman osia voisi mallintaa SIMIT CTE -ohjelmistolla.

Kuva 44. SIMIT CTE komponentti



7 Yhteenveto

Tämän opinnäytetyön tavoitteena oli luoda realistinen SIMIT-malli jakaja-asemasta, jota voidaan käyttää tulevaisuudessa ”virtuaalilaboratorio” tyyppisessä opetuksessa. Tämä antaa opiskelijalle mahdollisuuden testata oman PLC-ohjelmansa kyseiselle SIMIT-mallille ja seurata reaaliajassa sen toimintaa. Työn sisäiset tavoitteet jakautuivat kahteen osaan, joista ensimmäinen osa keskittyy TIA Portaalin PLC-ohjelman toimintaan, ja toinen osa keskittyy SIMIT SP -mallinnusohjelmistoon ja sen simulointiin.

TIA Portaalin tavoitteena oli luoda Feston MPS-alustan dokumentaation mukainen toiminta. Tähän kuuluu erikoistilanteet, sekä dokumentaation mukainen I/O-rajapinta. Ohjelmakoodilla täytyy olla hyvä dokumentaatio, joka on selkeä ja sillä on helposti luettava rakenne. Täytyy olla HMI-käyttöliittymä, mistä käyttäjä voi ohjata automaatiosekvenssistä minkä tahansa kappaleen ohjelman simulaatiota, ja mistä käyttäjä voi asettaa manuaalitilan päälle, mistä hän voi ohjata yksittäisiä liikkeitä.

SIMIT-mallin tavoitteena oli luoda realistinen virtuaalimalli kyseisestä Feston MPS-alustasta. Virtuaalimalliin kuuluu kappaleiden liikkeiden ja sijaintien visualisointi, sekä anturien ja toimilaitteiden tilan visualisointi. Täytyy olla erikoistilanteiden simulaatiomahdollisuus kuten tunnistamattoman kappaleen tai kääntäjävian. Virtuaalimallin työtilojen täytyy olla selkeät ja luettavia rakenteeltaan. Täytyy luoda erillinen käyttöohje virtuaalimallista, ja lopuksi luoda komponentti käyttäen SIMIT CTE -ohjelmistoa.

Kappaletavara-automaatio on mitä tahansa tuotantoautomaatiota, jossa tuotetaan kappaleita kuten elintarvikkeita. Tämä Festo MPS-alusta ei tällaisenaan välttämättä sovellu parhaiten kappaletavara-automaatiossa. Jos tehdään pieniä muutoksia alustalle, esimerkiksi alumiiniradat voisi korvata oikeilla kuljettimilla, josta tulisi vähäsen ohjelmakoodin muutosta. Tämän muutoksen jälkeen voidaan kuvitella tämän jakaja-aseman soveltuvan erittäin hyvin

johonkin suurempaan kappaletavara-automaatiotehtaaseen. Tehtaassa tämän tehtävänä voisi olla jaotella, esimerkiksi tuotantolaatikoita omille kuljettimille, jotka voisivat mennä omille roboteille jatkokäsittelyä varten.

Näistä tavoitteista ja käytännön osuudesta voidaan päätellä tämän opinnäytetyön onnistuneen erittäin hyvin. Työn jokaisen osan toimivan mallikkaasti. Samalla TIA Portaalin sekä SIMIT SP osuudet työstä ovat luotu lukijaa ajatellen, ja pyritty pitämään kaiken selkeänä ja helposti ymmärrettävissä.

Lähteet

Alvela, J.-P. (2022). *PLC-ohjelmistojen testaus ympäristö* [AMK-opinnäytetyö, Tampereen ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/782778/Alvela_Janne.pdf?sequence=2

Ball, K. (2015). *The Dawn of the Programmable Logic Controller (PLC)*. https://www.automation.com/en-us/articles/2015-2/the-dawn-of-the-programmable-logic-controller-plc

Churchville, F. (2021). *User Interface (UI)*. https://www.techtarget.com/searchapparchitecture/definition/user-interface-UI

Ervalahti, J. (2018). *Peruspiirien parantaminen Siemens TIA Portal -ympäristössä kunnossapidon tarpeet huomioiden* [AMK-opinnäytetyö, Jyväskylän ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/158232/Theseus_Jani_Ervahti.pdf?sequence=1&isAllowed=y

Festo. (n.d.). *Sorting Station technical data.* https://ip.festo-didactic.com/InfoPortal/MPS/SortingStation/EN/TechnicalData.html

Hagman, J. (2018). *OHJELMAKOMPONENTTEIHIN PERUSTUVA OHJELMOINTITAPA LOGIIKKAOHJELMOINTIIN SIEMENS TIA-PORTAL -OHJELMOINTIYMPÄRISTÖSSÄ* [AMK-opinnäytetyö, Oulun ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/154389/Hagman_Janne.pdf?sequence=1&isAllowed=y

Hirvonen, E. (2021). *SIMATIC SIMIT -SIMULAATTORIN HYÖDYNTÄMINEN AUTOMAATIOTEKNIIKAN OPINNISSA* [AMK-opinnäytetyö, Lapin ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/501004/hirvonen_erkki.pdf?sequence=2&isAllowed=y

Korva, N. (2019). *WIRELESSHART IN INDUSTRIAL ENVIRONMENT* [AMK-opinnäytetyö, Lapin ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/169887/Thesis%20of%20Niilo%20Korva.pdf?sequence=2&isAllowed=y

Kytölä, O. (2023). *KAASUN TÄYTTÖLINJASTON KULJETTIMEN LOGIIKAN MIGRAATIO* [AMK-opinnäytetyö, Oulun ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/793166/Kytola_Olli.pdf?sequence=2&isAllowed=y

Leinonen, L. (2020). *AUTOMAATIOJÄRJESTELMÄN MODERNISOINTI* [AMK-opinnäytetyö, Oulun ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/344833/leinonen_leevi.pdf?sequence=2&isAllowed=y

Leppälä, V. (2023). *Simulaatioalustan hyödyntäminen nosturiautomaatiossa* [AMK-opinnäytetyö, Tampereen ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/795475/Leppala_Veera.pdf?sequence=3

Paasikivi, R. (2018). *OHJELMOITAVIEN LOGIIKOIDEN VARMUUSKOPIOINTI* [AMK-opinnäytetyö, Satakunnan ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/142253/Paasikivi_Roope.pdf?sequence=1&isAllowed=y

Peterson, D. (2022). *The Origin Story of the PLC* [kuva]. https://control.com/technical-articles/the-origin-story-of-the-plc/

Pirttijoki, T. (2023). *Simatic S7 -tuoteperhe uudistuu vauhdilla*. https://blog.siemens.com/2023/08/simatic-s7-tuoteperhe-uudistuu-vauhdilla/

PLCopen. (n.d.). *Status IEC 61131-3 standard*. https://plcopen.org/status-iec-61131-3-standard

Process Solutions. (2020). *A Brief History of Programmable Logic Controllers (PLCs)*. https://processsolutions.com/a-brief-history-of-programmable-logic-controllers-plcs/

Ruha, M. (2019). *Toimilohko- ja kuvageneraattori Siemens TIA Portal -ohjelmointityökaluun* [AMK-opinnäytetyö, Jyväskylän ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/171991/Opinn%C3%A4ytety%C3%B6_julkaistava.pdf?sequence=2&isAllowed=y

Salila, E. (2015). *PLC-JÄRJESTELMIEN ELINKAARIANALYYSI* [AMK-opinnäytetyö, Lahden ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/93319/Salila_Eetu.pdf?sequence=1&isAllowed=y

Siemens. (2018). *SIMATIC throughout history*. https://www.siemens.com/global/en/company/about/history/specials/60-years-of-simatic.html

Siemens. (2021). *SIMATIC S7-PLCSIM Advanced Function Manual*. https://cache.industry.siemens.com/dl/files/879/109798879/att_1071484/v1/s7-plcsim_advanced_function_manual_en-US_en-US.pdf

Siemens. (2022a). *Automation does not come automatically*. https://www.siemens.com/global/en/company/about/history/specials/175-years/simatic.html

Siemens. (2022b). *Real progress for all stakeholders, including employees*. https://www.siemens.com/global/en/company/about/history/specials/175-years/pension-fund.html

Siemens. (2022c). *The best doctor is the one who does not have to come*. https://www.siemens.com/global/en/company/about/history/specials/175-years/company-doctor.html

Siemens. (2022d). *We electrify urban mobility – since 1881*. https://www.siemens.com/global/en/company/about/history/specials/175-years/electric-tram.html

Siemens. (2022e). *X-Ray, the beginning of medical imaging*. https://www.siemens.com/global/en/company/about/history/specials/175-years/x-ray-system.html

Siemens. (2023a). *Sales and delivery release incl. Download: Simulation software SIMIT V11.1*. https://support.industry.siemens.com/cs/document/109820441/sales-and-delivery-release-incl-download-simulation-software-simit-v11-1?dti=0&lc=en-TZ

Siemens. (2023b). *SIMATIC SIMIT Simulation Platform (V11.1)*. https://cache.industry.siemens.com/dl/files/650/109823650/att_1152291/v1/SIMIT_enUS_en-US.pdf

Siemens. (2023c). *TIA Portal V19 tehostaa ohjelmointia entisestään*. https://www.siemens.com/fi/fi/yhtio/stories/tuoteuutiset/tia-portal-v19-tehostaa-tekemista-entisestaan.html

Siemens. (n.d.). *Simatic-automaatiojärjestelmät*. https://www.siemens.com/fi/fi/tuotteet/teollisuus/simatic-automaatiojarjestelmat.html

Siemens-healthineers.com. (n.d.). *Biograph mMR*. https://www.siemens-healthineers.com/magnetic-resonance-imaging/mr-pet-scanner/biograph-mmr#06780752

Virtanen, A. (2012). *Automaatiojärjestelmän käyttöliittymä* [AMK-opinnäytetyö, Turun ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/52555/Virtanen_Aleksi.pdf?sequence=1&isAllowed=y

Virtanen, M. (2021). *Digitaalinen kaksonen vanerin ladontalinjasta* [AMK-opinnäytetyö, Hämeen ammattikorkeakoulu]. https://www.theseus.fi/bitstream/handle/10024/493921/Digitaalinen%20kaksonen%20vanerin%20ladontalinjasta.pdf?sequence=2

Liite 1. Laskurin ajastin ennen stopparia

```
// "Laskurin_Ajastin" on ajastin jonka avulla tunnistetaan vaaleanpunainen kappale.
//  Tunnistus perustuu inputin "Kappale tunnistettu" (%I0.4) "1" asennon myöhäistämiseen.
// Resettinä toimii joko ratojen omat kuittausnapit tai Jakaja-Asemal_GRAPH stepit 9, 10 tai 11.

"Laskurin_Ajastin".TONR(IN:="Kappale tunnistettu",
                        R:="STEPPI" = 9 OR "STEPPI" = 10 OR "STEPPI" = 11 OR "Musta_Rata_Kuittaus" OR "Vaaleanpunainen_Rata_Kuittaus" OR "Metalli_Rata_Kuittaus",
                        PT:=T#350MS,
                        ET=>"Laskurin_Ajastimen_Aika");
```

Liite 2. Kappalelaskuri TIA Portaalissa

```
13  // "Musta_Rata_OK" tarkoittaa radassa olevan tilaa kappaleille.
14  //  Jos radalla on tilaa, niin "Musta_Muisti" on "0" asennossa.
15
16  IF "Musta_Muisti" = 0 THEN
17
18      "Musta_Rata_OK" := 1;
19
20  ELSE
21
22      "Musta_Rata_OK" := 0;
23
24  END_IF;
25
26
27  // Riveillä 32-36 kappale on mustan värinen. "Musta_Laskuri" laskee mustien kappaleiden lukumäärää 0-5.
28  // Kun kappaleiden lukumäärä radalla on 5 niin välimuuttuja "Musta_Muisti" menee "1" asentoon.
29  // Jos halutaan resetoida laskin niin voidaan painaa kuljettimelle tarkoitettua reset painiketta "Musta_Reset".
30  // "Musta_Rata_Kuittaus" kuittaa radan kerätyksi jolloin se resetoi laskurin sekä "Musta_Rata_Täynnä_Ajastin".
31
32  "Musta_Laskuri".CTU(CU:="Musta_Rata_OK" AND "Laskurin_Ajastin".Q AND "Kappale tunnistettu" AND NOT "Kappale ei ole musta" AND NOT "Kappale on metallinen",
33                      R:="Musta_Rata_Kuittaus",
34                      PV:=5,
35                      Q=>"Musta_Muisti",
36                      CV=>"Musta_Laskuri".CV);
37
38  // Riveillä 40-44 tarkastetaan onko ensimmäinen musta kappale laskettu oikein ennen, kuin Jakaja-Asemal_GRAPH etenee stepistä 5 eteenpäin.
39
40  IF "Musta_Laskuri".CV >= 1 THEN
41      "Musta_Kappale" := TRUE;
42  ELSE
43      "Musta_Kappale" := FALSE;
44  END_IF;
45
46  // "Musta_Rata_Täynnä_Ajastin" Viivästää "Musta_Rata_Täynnä" hälytystä T#4000MS ajalla.
47
48  "Musta_Rata_Täynnä_Ajastin".TONR(IN:="Musta_Muisti",                          // Inputtina käytetään "Musta_Muisti" joka on "Musta_Laskuri".Q.
49                                   R:="Musta_Rata_Kuittaus",                    // Radan oma kuittausnappi.
50                                   PT:=T#4000MS,                                // Viivästetty aika.
51                                   Q=>"Musta_Rata_täynnä");                     // Hälytys kun rata on täynnä.
52
```

Liite 3. SIMIT SP -käyttöohje

HAMK
Hämeen ammatti-
korkeakoulu

SIMIT SP -käyttöohje

Ammattikorkeakoulututkinnon raportti
Sähkö- ja automaatiotekniikan koulutus, Insinööri (AMK)
Kevät 2024
Aleksi Turunen