Saimaan ammattikorkeakoulu
Tekniikka Lappeenranta
Konetekniikka

Tomi Pinomäki

Offset-painolinjan sivurekisterin säätö

Opinnäytetyö 2019

Tiivistelmä

Tomi Pinomäki
Offset-painolinjan sivurekisterin säätö, 52 sivua, 7 liitettä
Saimaan ammattikorkeakoulu
Tekniikka Lappeenranta
Konetekniikka
Opinnäytetyö 2019
Ohjaajat: lehtori Timo Eloranta, Saimaan ammattikorkeakoulu

Opinnäytetyön tavoitteena on tuottaa ratkaisu Tetra Pak Production Oy:n offset-painolinjan sivurekisterin säädön automatisointiin ja näin ollen vähentää rekisterin säädön vaatimaa työaikaa linjan käyttäjiltä sekä vähentää sivurekisterin säädön aikaista tuotantohävikkiä.

Teoriaosuus käsittelee automatisoidun laitteiston keskeisimpiä automaatiokomponentteja sekä niiden käyttöön ja toimintaan liittyvää perustietoa. Teoriaosuudessa pyritään antamaan riittävä perustieto tuotetun ratkaisun toiminnan ymmärtämiseksi. Lisäksi esitellään offset-painolinjan toiminta, keskeisimmät laitteet ja prosessit.

Toteutuksen suunnittelua varten oli selvitettävä toiminnalliset tarpeet ja lähtökohdat. Ratkaisu osoittautui teknisesti laaja-alaiseksi ja toimivaksi. Opinnäytetyössä käsiteltiin pääasiassa ohjelmoitavan logiikan ohjelmaa, sen kehitystä, toteutusta ja toimintaa. Kehitettyä laitteistoa on mahdollista tehostaa edelleen tulevaisuudessa.

Asiasanat: sivurekisteri, offset-paino, ohjelmoitavalogiikka

Abstract

Tomi Pinomäki
Automated side register for Offset-printing press, 52 Pages, 7 Appendices
Saimaa University of Applied Sciences, Lappeenranta
Degree Programme in Mechanical engineering
Bachelor´s Thesis 2019
Instructor: Mr. Timo Eloranta, Lecturer, Saimaa University of Applied Sciences

The purpose of the thesis is to create a solution for an automated side register in Offset printing press of Tetra Pak Production Oy. The automated side register would decrease the amount of process waste during a set up in the printing press and save time of the process operators.

Theoretical part of the thesis provides basic knowledge of an Offset-printing process and line, components of the solution and functionality. This part will give a needed information to be able to understand the created solution.

Requirements for the solution had to be clarified in the beginning. The thesis mostly explains the program of programmable logic controller, how it was developed and how it works. The solution proved to be technically complex, functional and can be developed even further in the future.

Keywords: side register, offset-printing, programmable logic controller

Sisällys

1 Johdanto

1.1 Opinnäytetyön tavoitteet ja rajaus

Tetra Pak Production Oy:n offset-painolinjan sivurekisteriä säädetään käsin visuaalisten kohdistusmerkkien näyttämän tarpeen mukaisesti. Kohdistus on hidasta ja tuotannon käynnistyksessä (Set Up) tuleva hävikki on suurta, ennen kuin sivusuuntainen kohdistus on haluttu. Tämä säätöprosessi on tarvetta automatisoida.

Opinnäytetyön tavoite on tuottaa ratkaisu sivurekisterin säädön automatisointiin. Opinnäytetyö toimii myös tehtaan automaatiokunnossapidon oppaana, jossa säädön toteutus selvitetään mahdollisimman ymmärrettävästi. Tämä auttaa mahdollisissa ongelma- ja vikatilanteissa sekä tulevissa laitteiston kehitysprojekteissa. Opinnäytetyö keskittyy säädön automaation syvälliseen selvittämiseen, minkä lisäksi mekaaninen ratkaisu sekä säädön tarvitsemat sähkö- ja automaatiolaitteistot esitellään periaatetasolla.

1.2 Tetra Pak

Imatralla sijaitseva 2011 perustettu Tetra Pak Production Oy kuuluu Tetra Pak -yhtiöihin, joka taas kuuluu Tetra Laval -konserniin. Tetra Pak on kansainvälinen 1951 perustettu ruotsalaislähtöinen yritys, joka on erikoistunut nestepakkausmateriaaleihin ja täyttökoneisiin. Tetra Laval -konsernin kautta Tetra Pak pystyy toimittamaan koko tuotantoketjun. (Tetra Pak 2019)

Tetra Pakin tunnetuin tuote on nelisivuinen monitahokas pakkaus eli tetraedri. Pakkauksen muoto tunnetaan “tetrana”. Tämän pakkauksen voidaan sanoa käynnistäneen nesteiden pakkaamisen mullistuksen, jonka myötä lasipakkaukset vähenivät nopeasti esimerkiksi meijerituotteissa. (Tetra Pak 2019)

Nykyään Tetra Pak on maailman suurin elintarvikepakkausten valmistaja vuoden 2018 liikevaihdon ollessa 11,2 miljardia euroa. Tetra Pak työllistää noin 25500 henkilöä maailmanlaajuisesti. (Tetra Pak 2019)

Tetra Pak Production Oy työllistää Imatralla hieman alle viisikymmentä henkilöä ja prosessi koostuu pääasiassa offset-painolinjasta ja sivusaumauslinjasta oheislaitteineen ja prosesseineen.

2 Offset-painolinja

2.1 Linjan pääosat

Tetra Pak Production Oy:n offset-painolinja on rullalta arkiksi -tyyppinen, mikä tarkoittaa sitä, että linjan alkupäähän tuodaan kartonki rullalla ja linjan loppupäästä saatava tuote on yksittäisinä arkkeina. Usein tämänkaltainen elintarvikepakkausmateriaaliteollisuuden painolinja tuottaa rullalta rullalle, jolloin linjalta saatava tuote on uudelleen rullattu painoprosessin jälkeen. Arkeiksi erittely tapahtuu vasta, kun tuote jatkokäsitellään pidemmällä prosessissa.

Offset-painolinja alkaa aukirullaimesta, jossa elintarvikepakkausmateriaalikartonki aukirullataan linjalle ajettavaksi. Yleensä aina aukirullaimissa on kaksi asemaa, jolloin uusi rulla voidaan valmistella toiseen asemaan, kun edellinen rulla on ajossa. Näin ollen vältytään tuotannon keskeytyksiltä rullan loputtua.

Aukirullaimelta kartonki kulkee jatkoskoneen läpi, jossa edellisen rullan loputtua tyhjentyneen rullan loppupää ja täyden rullan alkupää liitetään tarkoitukseen sopivalla karviteipillä yhteen automaattisesti. Karviteippi valmistellaan käsin jatkoskoneeseen.

Jatkoksen teon aikana prosessiin ajetaan kartonkia festoonista, joka toimii puskurina sen ajan, kun kartonkirullat ovat hitaassa liikkeessä jatkoksen teossa. Festoon on usean telan järjestelmä, johon voidaan varastoida kartonkia useita kymmeniä metrejä. Tehtaan festoon on pystymallinen eli kartonkirata kulkee ylös ja alas telojen väliä. Festooneja on myös vaakamallisia, mutta toimintaperiaate on täysin sama. Tämän jälkeen uusi rulla kiihdytetään linjan ratanopeuteen ja festoon ajetaan hiljalleen täyteen kartonkia.

Kartongin täytyy kulkea suoraan ja tasaisella kireydellä läpi painoprosessin. Festoonin jälkeen on radan ensimmäinen rataohjuri ja puristustela. Puristustelalla kontrolloidaan radan kireys kiihdyttämällä ja jarruttamalla, joten siitä usein käytetään englanninkielistä nimitystä “pull & break” ja paperiteollisuudessa siitä käytetään nimitystä “nippi”. Radan kireyttä valvotaan analogisilla mittauksilla tietyistä mittateloista sekä myös momentti- ja virta-arvojen valvonnalla servo-ohjaimissa.

Ensimmäisen puristustelan ja radanohjauksen jälkeen kartonki sähkökäsitellään eli koronoidaan. Käsittelyssä siihen tehdään tasaisella sähkön läpilyönnillä (valokaari) pieniä reikiä. Nämä reiät auttavat painovärin kiinnittymisessä kartonkiin. Usein kartonki on valmiiksi sähkökäsitelty kartonkikoneella. Sähkökäsittelyasemassa myös tehdään kartonginpinnan puhdistus pienestä pölystä, joka on jäänyt kartonkiin tekoprosessista ja kartongin sähköstaattisuudesta johtuen.

Sähkökäsittelyn jälkeen alkaa varsinainen painoprosessi. Tetra Pak Production Oy:n painolinja sisältää kuusi painoyksikköä, joiden jokaisen jälkeen on yhdestä kahteen ultraviolettikuivainta. Kuivaimissa on voimakas ultraviolettivalaisin, joka kuivaa edellisen painoyksikön ultraviolettiaktiivisen painovärin riittävän kuivaksi prosessin seuraavaa yksikköä varten. Kuivaimilta vaaditaan hyvin paljon tehoa ratanopeuden noustessa kolmeensataan metriin minuutissa ja painoyksiköiden välin ollessa vain noin puolitoistametriä.

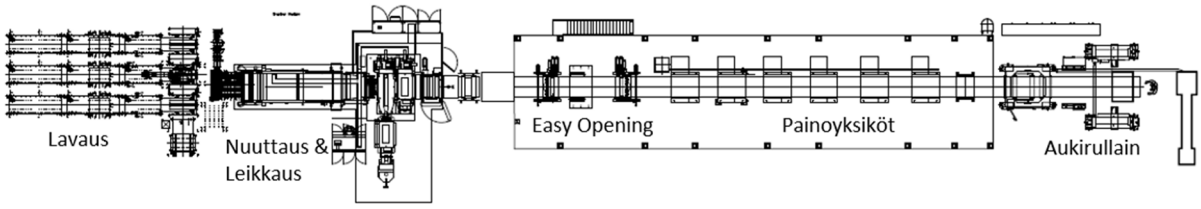
Painoyksiköissä painoväri siirretään värikaukalosta erityisten telojen kanssa painopellille, josta pellille tehdyn osakuvan kautta väri siirtyy kumitelan kautta kartongin pintaan. Painopellit on valmisteltu PrePress-osastolla ja haluttu painokuva koostetaan rasteripisteillä, tarvittavien värien muodostamista osakuvista. Jokaisessa painoyksikössä on oma värinsä eikä värejä vaihdeta yksiköissä eri tuotteille. Yksiköt yhdestä neljään ovat perusvärejä eli musta, syaani, magenta ja keltainen. Kaksi viimeistä yksikköä ovat erikois- tai lisävärejä, joita ei saada tehtyä riittävän hyvin yhdistämällä neljää perusväriä. Erikoisvärinä viidennessä yksikössä on tummansininen ja kuudennessa yksikössä oranssi. Seitsemäs painoyksikkö on myös jo suunnitteilla, jolloin painolaatua saadaan parannettua entisestään.

Kuvan painatuksen jälkeen on toinen puristustela, jolla pidetään ratakireys painoyksiköissä haluttuna. Seuraavana prosessissa voidaan halutut kohdat pinnoittaa lakalla, mikä tuo kiiltoa painokuvaan. Tämän jälkeen levitetään Easy Opening -neste (EO), jos pakkaus on kaatonokka-tyyppinen. Neste levitetään kaatonokan sauman kohtiin, jolloin täyttökoneella suljettavan pakkauksen kaatonokkasaumat ovat sen verran heikentyneet, että ne voidaan avata taivuttamalla. Korkilliset nestepakkaukset yleistyvät koko ajan, joten EO-nesteen käyttötarve vähenee samalla.

EO-nesteen levittämisen jälkeen painettu kuva tarkastetaan konenäkölaitteistolla, joka vertaa jokaisen painetun nestepakkausaihion kuvaa sille opetettuun referenssikuvaan. Jos asetettu hylkäysraja ylittyy, laite antaa siitä tiedon jäljempänä linjalla tulevalle ejektorille, jossa hylätty aihio poistetaan.

Kuvanlaadun automaattisen tarkastamisen jälkeen kartonkirata jälleen suunnataan ennen viimeistä puristustelaa. Tällä puristustelalla säädetään ratakireys halutuksi ja syötetään kartonki nuuttaukseen. Nuuttauksessa pakkausaihioiden taittokohtiin tehdään “saranointi”, jotta aihio taittuu halutuista kohdista. Nuuttauksen jälkeen aihiota leikataan irti toisistaan leikkurissa ja syötetään ejektorin läpi tarkastuspöydälle kolmena limittäisenä materiaalivirtana. Tältä pöydältä poimitaan aihiota käsin tehtävää laadunvalvontaa varten.

Tarkastuspöydältä materiaalivirrat menevät pinoajaan, jossa aihioista tehdään hieman alle tuhannen kappaleen pinoja. Robotti nostaa pinot lavausasemalla tuotteen mukaisille lavoilleen. Painolinjan layout on nähtävissä kuvasta 1 nimettyine pääsektioineen.



Kuva 1. Painolinja (Tetra Pak Production Oy)

2.2 Offset-painotekniikka

Offset-painamista kutsutaan myös nimellä laakapaino, sillä siinä värin painopinnalle siirtävät ja siirtämättömät kohdat ovat samassa tasossa toisin kuin muissa painomenetelmissä. Offset-painomenetelmässä hyödynnetään painovärin tarttumista haluttuihin kohtiin painopellissä. Painopellin pinnasta poistetaan laserlaitteella ne kohdat, joihin ei haluta painoväriä tarttuvan. Tämän jälkeen painopelti saa vielä kemiallisen käsittelyn, jota kutsutaan kehittämiseksi. (Graafinen 2015)

2.2.1 Painokuvien kohdistus

Painopellit kiinnitetään painokaseteissa oleviin painorumpuihin. Peltien täytyy olla linjattuina tarkalleen kohdistusmerkkeihinsä ja kiristettyinä haluttuun kireyteen. Pellin alus on kumipintainen, mikä tasaa painetta pellin pinnalla.

Painojäljen pituussuuntainen rekisterisäätö eli kohdistus tehdään säätämällä painoyksikön painorummun pyörimisnopeutta nopeammaksi tai hitaammaksi suhteessa linjan ratanopeuteen. Jokaista painoyksikköä kohdistetaan itsenäisesti.

Painojäljen sivuttaissuuntainen rekisterisäätö tehdään siirtämällä painorumpua linjaan nähden kohtisuoraan eli linjan keskiviivasta katsottuna oikealle tai vasemmalle. Jokaista painoyksikköä säädetään itsenäisesti. Kuvien kohdistus tapahtuu millin kymmenysten tarkkuudella mahdollisimman nopeasti huonosta laadusta johtuvan hävikin minimoimiseksi.

2.2.2 BST Eltromat Registar

Linjan nykyisen pituussuuntaisen rekisterisäädön on toimittanut saksalainen BST Eltromat International ja kohdistuslaitteiston kauppanimi on Registar. Laite sisältää kaksi kameraa, joista toinen on painoyksiköiden jälkeen painojäljen kohdistusta varten ja toinen ennen nuuttausta. Jälkimmäinen kamera on nuuttauksen ja leikkauksen kohdistamiseen. Kamerat ovat tahdistettu kuvaamaan kuvassa 2 nähtäviä painopeltien kohdistuspisteitä. Jokainen painoyksikkö tuottaa omanvärisensä pisteen. Registarin teollisuustietokonepohjainen sovellus arvioi kohdistuspisteiden linjausta suhteessa toisiinsa ja kommunikoi servo-ohjaimien ja ohjelmoitavan logiikan kanssa kohdistustarpeesta.

Kuva 2. Kohdistuspisteet (Tetra Pak Production Oy)

Registariin on saatavissa optiona sivusuuntaisen kohdistustarpeen ohjaustiedot, joita on tarkoitus hyödyntää tässä projektissa.

2.2.3 Sivurekisterin säätäminen

Painokasetin painopellin rummussa on kuuden millimetrin sivusuuntainen kokonaissäätöalue, jota voidaan säätää käsipyörällä. Käsipyörässä on analoginen asennonosoituskello.

3 Logiikalla ohjattava servosäätöjärjestelmä

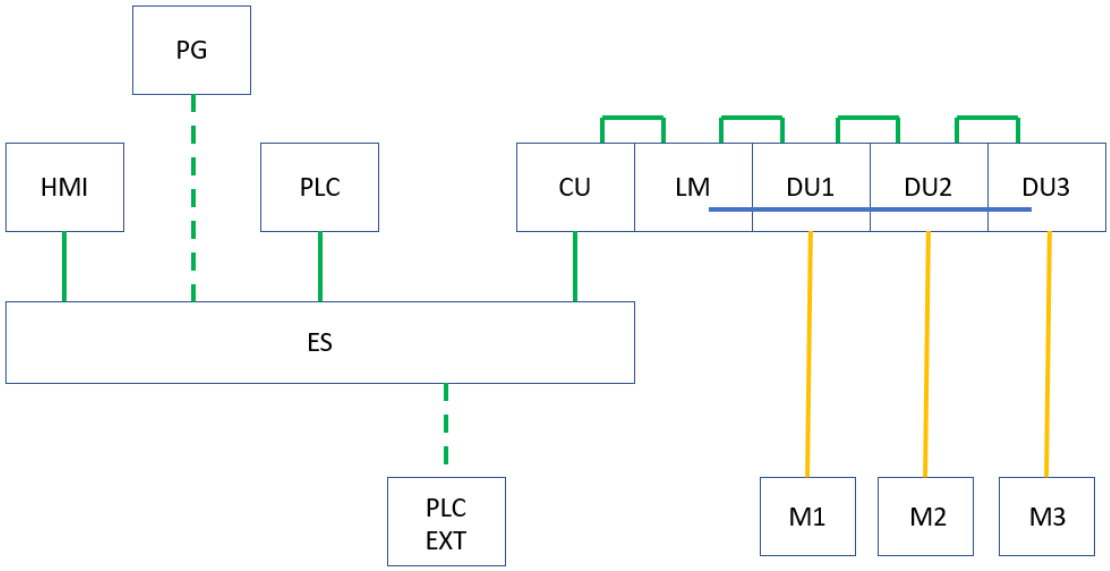
3.1 Servojärjestelmän pääkomponentit

Ohjausyksikkö (Central Unit) on servojärjestelmän aivot eli se prosessoi sille annetut ohjauskäskyt, jotka voivat tulla esimerkiksi ohjelmoitavalta logiikalta tai vaikka käsin ohjaten painonapeilta. Ohjausyksikkö lukee servojen oloarvoja ja huolehtii esimerkiksi siitä, että servon pyörintänopeus tai paikoitus on mahdollisimman lähelle haluttua.

Tehoyksikkö (Line Module) toimii tasasuuntaavana teholähteenä ja tuottaa varsinaisille servokäyttöyksiköille halutun syöttöjännitteen syöttökiskoston kautta. Kyseinen tasajännite on useiden satojen volttien suuruusluokkaa.

Servokäyttöyksikkö (Drive Unit) tuottaa servomoottorille syöttöjännitteen, -taajuuden ja -virran ohjausyksikön käskyjen perusteella.

Servomoottori muuttaa pyörivän magneettikentän avulla sähköenergian mekaaniseksi pyöriväksi liikkeeksi. Yleensä servomoottorin yhteyteen on kiinteästi asennettu moottorin paikan tilatiedosta kertova pulssianturi eli enkooderi. Pulssianturilla voidaan valvoa moottorin todellista pyörimisnopeutta eri kuormitustilanteissa sekä moottorin todellista etenemää paikoituskäytössä.



Kuva 3. Tyypillinen servojärjestelmä

Kuvassa 3 on esitettynä tyypillisen servojärjestelmän keskeisimmät pääkomponentit ja niiden liittyminen toisiinsa. Vihreällä viivalla ovat kuvattuna kiinteät kenttäväyläliitynnät ja vihreällä katkoviivalla usein tarvittavat kenttäväyläliitynnät. Tumman sininen viiva kuvaa servojen tehoyksikön tasajännitesyöttöä ja keltainen viiva servomoottoreille meneviä syöttö- ja ohjauskaapeleita. Kuvassa käytetyt lyhenteet ovat HMI – kosketuspaneeli, PG – kiinteä tai kannettava ohjelmointiasema, PLC – ohjelmoitavalogiikka, PLC EXT – ulkoinen ohjelmoitavalogiikka (esimerkiksi linjan päälogiikka), CU – servojen ohjausyksikkö, LM – tehoyksikkö, DU – servokäyttöyksikkö ja M – servomoottori.

3.2 Servojen ohjaustavat

3.2.1 Väyläohjaus

Nykyisin suurin osa liikeohjauksiin liittyvästä ohjauskommunikaatiosta tehdään erilaisten kenttäväyläohjausten avulla ja kenttäväylien käyttö on vähentänyt paljon analogia- ja I/O-viestien käyttöä teollisuusautomaatiossa. Pääsääntöisesti Siemens-laitteiden yhteydessä käytetään kahta erilaista kenttäväylätyyppiä: Profinet (tai Profinet IO) ja Profibus DP. Yleisesti Profibus DP -väylää kutsutaan pelkästään Profibus- tai DP-väyläksi. Muita teollisuuden kenttäväyliä ovat mm. Modbus eri variaatioineen, CAN, EtherCAT ja AS-i. Eri väylillä on myös eroja. Modbus on sarjaliikenneprotokolla, CAN on ajoneuvoteollisuuteen alun perin kehitetty kaksijohtiminen väylä. EtherCAT on Profinetin tavoin Ethernetiin pohjautuva. AS-i väylä on kaksijohtiminen ja kykenee syöttämään siihen kytkettyjen pienten laitteiden, kuten antureiden, tarvitseman syöttötehon.

Profinet pohjautuu toimistotietotekniikankin puolelta tuttuun Ethernet-lähiverkkoon ja usein Profinetin yhteydessä käytetäänkin standardoituja RJ45-tyyppisiä kaapeliliittimiä. Profinetin etuina ovat suuret tiedonsiirtonopeudet, suuri komponenttivalikoima, monimuotoinen verkkotopologia (laitteiden yhdistymistapa toisiinsa fyysisesti väyläverkossa) ja laiteverkon suuri mahdollinen osoitemäärä komponenteille, sillä verkon laitteet yksilöidään IP-osoitteen avulla. Monet laitekonfiguraatiot on mahdollista tehdä Profinet-väylän yli ilman, että tarvitsee erikseen käydä tekemässä joitain alkuasetuksia paikan päällä. Profinetin suurimmat erot ja edut verrattuna Ethernetiin ovat häiriöiden siedossa ja tosiaikaisessa teollisuusautomaation vaatimassa tiedonsiirtonopeudessa. (Profinet North America 2019)

Laitteiden suurin määrä Profinet-väylässä on lähes rajaton, mutta yleensä muut tekijät määrittävät laitteiden maksimimäärän, kuten esimerkiksi miten monen laitteen kanssa ohjelmoitava logiikka kykenee kommunikoimaan. Profibus-väylän suurin laitemäärä on 127. Käytännössä harvoin yhteen väylään kytketään maksimimäärää laitteita, sillä usein väylän fyysinen pituus kasvaisi jo niin pitkäksi,

että tarvittaisiin jopa useita signaalintoistimia vahvistamaan signaalia. Hyvin suuressa ohjauskeskuksessa voi olla lähes maksimimäärän verran Profibus-laitteita samassa väylässä, sillä silloin fyysiset etäisyydet ja ulkoiset häiriöt ovat pieniä. (Profinet North America 2019)

Profibus-väyläkaapelissa on kaksi johdinta, ja se on melko arka liitosten ylimenovastuksen kasvulle eli huonolle liitokselle. Huonot liitokset lyhentävät verkon fyysistä maksimipituutta ja lisäävät hieman häiriöalttiutta. Profibus-väylän topologia on väylämuotoinen eli laitteet ovat sarjassa ja ne yhdistyvät kaapeleilla toisiinsa yleensä jatkoliittimiä käyttäen. Väylän päissä olevien viimeisten laitteiden liittimissä kytketään sarjavastukset käyttöön.

Kenttäväylien tiedonsiirrot noudattavat tiettyjä tiedonsiirtoprotokollia. Tiedonsiirron kehykset sisältävät esimerkiksi tunnisteet, minkä verkossa olevan laitteen kanssa halutaan siirtää tietoa, mitä tietoa siirretään, sekä virheen tunnistuksia.

Kenttäväylissä on isäntälaitteita (Master) ja renkiä (Slave). Isännät kysyvät rengeiltä haluamiaan tietoja ja rengit vastaavat isännille. Tiedonsiirtoa voidaan suorittaa isäntienkin välillä.

3.2.2 Digitaali I/O -ohjaus

Servolaitteistoja voidaan ohjata myös jänniteohjauksena eli ns. digitaali I/O-ohjauksena (lyhyemmin pelkkä I/O). I/O tulee sanoista Input/Output, ja sillä viitataan siihen, että laitteistot lähettävät (Output) jännitesignaalin, joka on yleensä teollisuusautomaatiossa 24 volttia tasasähköä. Signaali vastaanotetaan (Input) toisessa laitteessa, joka tulkitsee sen päällä-tilatiedoksi, kun vastaanottavan tulokortin kynnysjännite ylittyy. Esimerkiksi Siemens 300 -logiikkasarjan digitaalitulokortin 6ES7321-1BH02-0AA0 jänniteväli päällä-tiedolle on 13 - 30 VDC. Vastaavasti kyseisen digitaalitulokortin jänniteväli poissa-tiedolle on -30 - +5 VDC. (Siemens 2019)

I/O-ohjaus soveltuu hyvin huonosti sellaisten servo-ohjausten kommunikointiin, joissa tarvitaan esimerkiksi välittää useita paikkatietoja tai nopeuksia logiikalta servo-ohjaimelle. I/O-ohjaus riittää, jos kiinteitä paikkoja on hyvin vähän eikä ohjaimelta tarvita esimerkiksi yksityiskohtaista vikadiagnostiikkaa logiikalle.

3.3 Anturit

Servosäätöjärjestelmiin liittyvien anturien eri tyyppimäärä voi olla suuri, sillä tarvittavat anturityypit riippuvat täysin sovelluksesta ja sen vaatimuksista. Tyypillisimmillään anturit ovat induktiivisia lähestymiskytkimiä, valokennoja ja erilaisia etäisyysantureita. Antureiden lähettämä signaali on joko jännitesignaali (PNP tai NPN), analoginen jännite- tai virtaviesti tai kenttäväyläsanoma.

PNP-signaali tarkoittaa sitä, että signaali on positiivinen tasajännite eli yleensä kaksikymmentäneljä volttia. NPN-signaali on nolla volttia tasajännitettä eli kuormaan tarvitaan silloin saman potentiaalin positiivinen tasajännite, jotta anturin tuottama signaali on tunnistettavissa. Myös positiivisella signaalilla tarvitaan kuorman toiselle puolelle saman potentiaalin nolla volttia potentiaalieron tunnistusta varten.

Tyypillisimmät analogiasignaalit ovat 0 – 10 VDC, 0 – 20 mA ja 4 – 20 mA. Tämä vaihteluväli kuvaa anturin tunnistamaa etäisyysväliä ja on yleensä lineaarinen.

Induktiiviset anturit tunnistavat metallisia kappaleita ja niistä varsinkin ferriittisille metalleille induktiiviset anturit soveltuvat parhaiten. Induktiivisten antureiden huonona puolena voidaan pitää suhteellisen lyhyttä tunnistusetäisyyttä, mutta vastaavasti hyvin lyhyen tunnistusetäisyyden anturit ovat myös hyvin tarkkoja. Ne soveltuvatkin hyvin esimerkiksi tarkkaan paikoituskäyttöön, jossa liikuteltavalla metallikappaleella on vain yksi positio anturia kohden. Paineilmasylintereiden rajakytkimet ovat yleensä pieniä induktiivisia antureita, millä tunnistetaan paineilmasylinterin alumiinisen kuoren läpi sylinterinmäntään kiinnitetty magneettinen kappale.

Valokennot ovat monikäyttöisiä ja niitä löytyy hyvin montaa eri tyyppiä, kuten kohteesta tunnistavia, peiliheijastavia ja kuituvalokennoja. Valokennoja on saatavilla myös melko pitkille tunnistusetäisyyksille. Tunnistusetäisyydet voivat parhaimmillaan olla kymmeniä metrejä. Valokennojen haittapuolina on niiden likaherkkyys, mutta monet nykyiset valokennot ovatkin jo melko kehittyneitä ja sietävät jonkin verran likaantumista, kuten pölyä.

3.4 Kosketuspaneeli

Nykyiset käyttöliittymät ovat enenevissä määrin kosketuspaneelityyppisiä kenttäväylään liittyviä laitteita. Ohjelmoitavuuden ja kenttäväylän takia niille ei tarvitse tuoda suurta määrää johdotuksia, kuten vastaavien ohjausten toteuttamiseen painonapeilla ja valoindikaattoreilla. Kosketuspaneelit ovat myös ohjelmoitavissa sovelluskohtaisesti kuhunkin käyttökohteeseen sopivaksi ja voidaan käyttää graafisesti kuvaavaa ulkoasua. Kosketuspaneeliin ohjelmoidut painikkeet, syöttökentät ja kuvaajat liitetään tageihin (TAG) eli nimettyihin muistialueisiin. Kun kosketuspaneeli ja ohjelmoitava logiikka kommunikoi keskenään, niin ne lukevat ja kirjoittavat tageja, jolloin niihin liitetyt paneeliohjaukset ohjaavat tai osoittavat halutusti.

3.5 Ohjelmoitavan logiikan pääkomponentit

3.5.1 Räkki

Siemensin 300-sarjan ohjelmoitavan logiikan moduulit kiinnitetään erityiseen kiinnityspohjalevyyn, jota kutsutaan räkiksi. Niitä on saatavilla eri mittaisina ja voidaan tarvittaessa katkaista sahaamalla haluttuun mittaan. Räkkiin kiinnitettävä logiikkamoduulit yhdistyvät toisiinsa moduuleiden pohjaan kiinnitettävien “siltaliittimien” välityksellä. Tämä liitin toimii väylänä ja siinä kulkee kaikki räkissä olevien moduulien välinen kommunikointi.

3.5.2 Teholähde

Teholähteellä tarkoitetaan verkkojännitteestä 110 – 230 VAC tasasuunnatun pienoisjännitteen 24 VDC tuottavaa jännitelähdettä. Laitetta kutsutaan usein nimityksellä “poweri”. Logiikan moduulit käyttävät kyseistä pienoisjännitettä apusähkönään eli tarvitsevat syötön. Siemensin oma 300-logiikkasarjan teholähde, kuten muutkin saman sarjan logiikkamoduulit käyvät edellä kuvattuun räkkiin.

3.5.3 Keskusyksikkö

Keskusyksikkö eli CPU (Central Processing Unit) toimii ohjelmoitavan logiikan aivoina. Se lukee logiikkaan syötetyt tiedot, suorittaa siihen syötetyn ohjelman

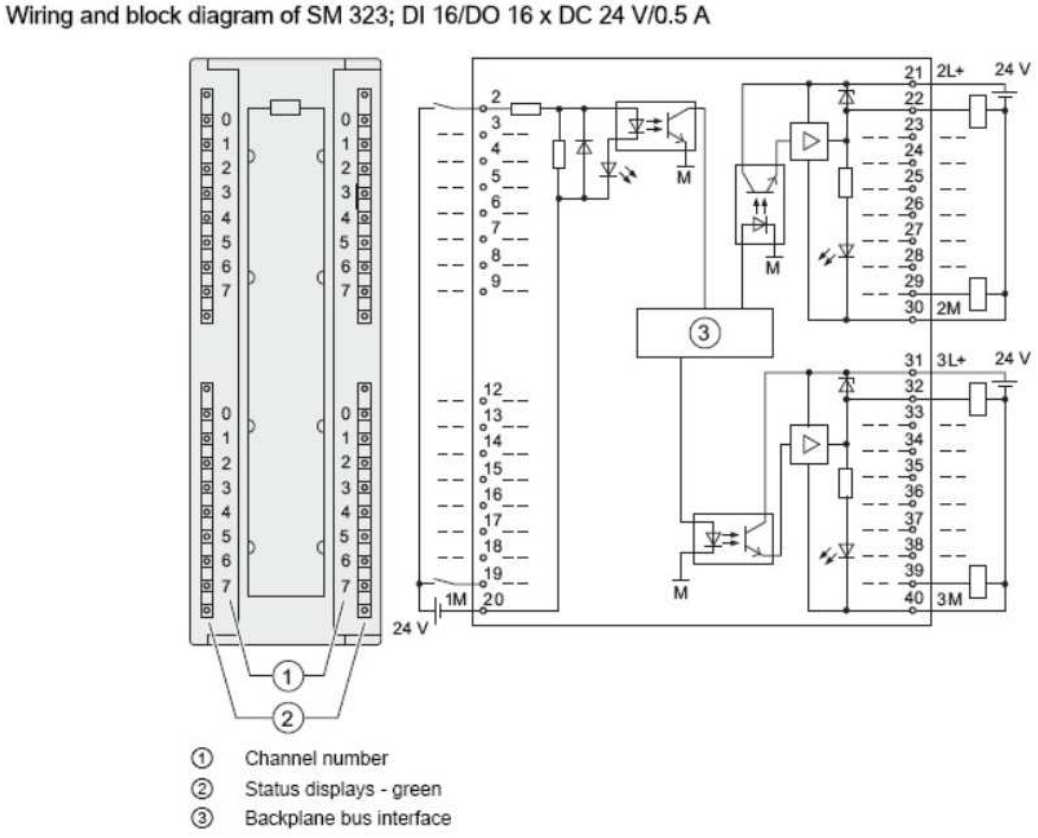
halutussa järjestyksessä ja tuottaa ohjelman mukaiset tulokset ja ohjaa halutut lähdöt.

3.5.4 Muistikortti

Keskusyksikköön, kuten joihinkin muihin moduuleihin ja ohjainlaitteisiin, voidaan liittää muistikortti. Muistikorttia yleensä käytetään tallentamaan jotain oleellisia pysyväistietoja ja varmuuskopioimaan ohjelmia. Muistikorttien kapasiteetit ovat melko pieniä suhteessa muussa tietotekniikassa käytettäviin muistikortteihin. Kapasiteetit ovat kymmenistä kilotavuista muutamiin megatavuihin.

3.5.5 Digitaalitulo ja -lähtökortti

Ohjelmoitava logiikka käsittelee siihen syötettyä tietoa ja antaa siihen laaditun ohjelman perusteella ulossyötteen. Tyypillisin ohjelmoitavan logiikan kokoonpano sisältää digitaaliset tulo- ja lähtökortit, varsinkin kappaletavaravan käsittelyssä. Voidaan myös käyttää yhteiskortteja, joissa on tietty määrä tuloja ja lähtöjä samassa kortissa. Kuvasta 4 nähdään yhdistelmäkortin kytkentäkaavio.



Kuva 4. Yhdistetty digitaalitulo- ja lähtökortti (Siemens)

Digitaalitulokorttiin, tai tulomoduuliin, syötetty kortille ominaisen kynnysjännitteen ylittävä potentiaaliero tulkitaan binäärisenä ykkösenä ja kynnysjännitteen alittava potentiaaliero tulkitaan binäärisenä nollana. Tällä tarkoitetaan, että logiikan tulkitsema tulotieto voi olla vain nolla (0) tai yksi (1). Usein myös tiloja kutsutaan nimityksillä “pois” ja “päällä”.

Digitaalilähtökortti, tai lähtömoduuli, on nykyään yleisimmin transistorilähtöinen. Joskus myös käytetään relelähtöjä, jos halutaan loppukuormaa ohjata suoraan lähtökortilta. Transistorilähdöllä usein ohjataan ohjausreleitä, joilla taas ohjataan haluttua loppukuormaa. Jos keskustilanpuute tai budjetti ei ole esteenä, transistorilähtö on mieluisampi valinta käyttöiän takia. Transistorilähtö on keskimäärin pitkäikäisempi kuin relelähtö. Digitaalilähdön tila voi olla nolla tai yksi, kuten tulokortissakin. Tulo- ja lähtökortit jaetaan vielä nopeisiin ja “normaaleihin” kortteihin. Yleensä kappaletavarateollisuuden ohjauksissa ei tarvita nopeita kortteja, mutta joissakin sovelluksissa nekin ovat tarpeen. Täytyy kuitenkin ensin miettiä riittääkö logiikan ohjelmankiertoaika hyödyntämään korttien nopeutta. Tarpeen mukaan voidaan hyvin nopeissa sovelluksissa käyttää myös laitteiston keskeytystuloja.

Ohjelmoitava logiikka lukee ohjelmankierrossa ensin tulot, sitten suorittaa ohjelman ja lopuksi kirjoittaa lähdöt ohjelman tulosten perusteella. Tämän jälkeen kierto alkaa alusta.

3.5.6 Analogiatulo ja -lähtökortti

Analogisia signaaleja käytetään silloin kun jonkin asian tilasta kertovaksi tiedoksi ei riitä pelkkä päällä- tai pois-tieto. Analogiasignaalit ovat jännite- tai virtatietoja. Tyypillisimmät käytetyt signaalit ovat 0 – 10 VDC, 0 – 20 mA ja 4 – 20 mA.

Vaikka saman analogiatiedon välittämiseen voidaan käyttää jännite- ja virtasignaaleita, niin niiden käytännön erot ovat suuret. Jännitetieto on hyvin kriittinen signaalin lähettävän laitteen ja sen lukevan eli vastaanottavan laitteen välimatkasta, liitosten ylimenoresistanssista, signaalikaapelin resistanssista ja ympäristön häiriöistä. Pahimmillaan virhettä voi tulla monia prosenttiyksiköitä ja signaalintila vaihtelee ympäristön häiriöistä johtuen. Virran kulkuun vaaditaan suljettu silmukka ja signaalin syöttävä laite sekä signaalin lukeva laite sijaitsevat samassa silmukassa. Sen vuoksi signaalintaso ei voi laskea matkalla syöttävältä

laitteelta vastaanottavalle laitteelle. Virtasignaalin tapauksessakin täytyy kuitenkin huomioida samat asiat kuin jännitesignaalissa eli kaapelin pituus, matkalla olevat vastukset ja häiriöt. Syöttävällä laitteella on omat rajoitteensa siinä, miten paljon se pystyy syöttämään ja ylläpitämään tehoa eli kuormalla on vaikutusta.

Analogiasignaali muutetaan analogiatulokortissa numeeriseksi lukuarvoksi. Siemens käyttää lukuväliä 0 – 27648, nollan vastatessa käytetyn signaalin nollakohtaa ja 27648 signaalin huippuarvoa. Kortteja on saatavilla eri resoluutioilla, mikä tarkoittaa miten tarkkaan luettu signaali voidaan muuttaa lukuarvoksi. Pienemmällä resoluutiolla tarkkuus on huonompi kuin isommalla resoluutiolla ja lukuarvo menee jyrkemmin portain signaalin tason vaihtuessa. Lähtökortti toimii päinvastoin kuin tulokortti eli logiikan ohjelman tuottama lukuarvo (0 – 27648) muutetaan halutuksi jännite- tai virtatiedoksi. Tältä lukuarvoväliltä poikkeavia lukuja voidaan käyttää vikadiagnostiikassa.

3.5.7 Muita moduuleita

Tulo- ja lähtökortteihin kuulumattomia moduuleita ovat tyypillisimmin enkooderi- ja kommunikaatiomoduulit.

Enkooderimoduuli eli pulssianturikortti on nopean digitaalitulokortin kaltainen, mutta pulssien luentaan tarkoitettu moduuli. Pulssiantureita on monen tyyppisiä ja yleensä pulssianturikortit voidaan konfiguroida jokaista pulssianturimallia lukevaksi. Pulssianturityypeillä lähinnä määritellään anturin syöttämä jännitetaso, transistorilähdön tyyppi (NPN tai PNP) ja pulssimäärä eli monta pulssia anturi antaa yhdellä anturin kierroksella. Laitteistokonfiguraatiossa määritellään käytetty pulssianturityyppi edellä mainituin perustein. Joskus voidaan tarvita myös itse enkooderimoduulin konfigurointia sopivaksi valintakytkimillä asettamalla.

Keskusyksikkö sisältää erilaisia kommunikointiin käytettäviä portteja mallista riippuen, ja jos nämä eivät riitä, logiikkaan voidaan liittää erillisiä kommunikaatiomoduuleita. Portit ovat pääsääntöisesti RJ45-tyyppiä (Profinet) ja 9-pinnisiä D-sub-tyyppisiä (MPI, Profibus, sarjaliikenne -protokollat).

3.6 Ohjelmointikielet

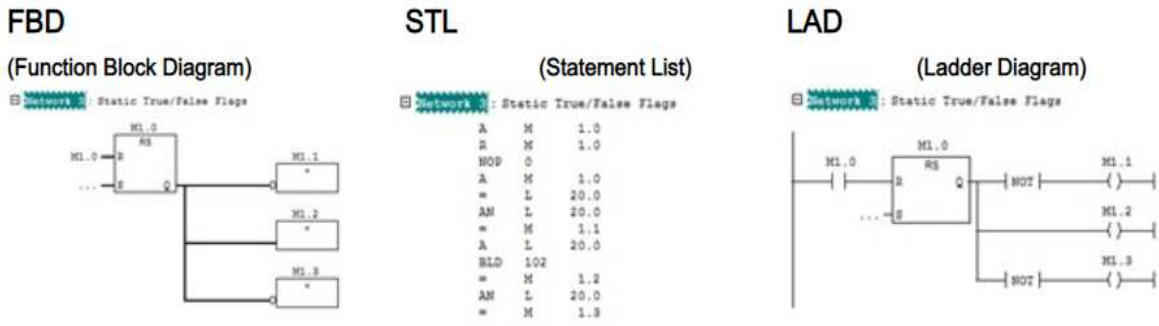
Ohjelmoitavalogiikka tarvitsee toimintaohjeet eli logiikkaohjelman suorittaakseen halutut tehtävät. Siemensin ohjelmoitavissa logiikoissa on käytettävissä neljä tai viisi erilaista ohjelmointikieltä riippuen siitä, miten määritellään ohjelmointikieli. Ohjelmointikielet ovat STL/ST, FBD, LAD, SCL ja S7-Graph. Näistä jokaisella on omat hyvät ja huonot puolensa suhteessa toisiin.

STL (Structured Text List) tai ST (Statement List) on käskylistamuotoinen hyvin paljon tietotekniikasta tutun Pascal-ohjelmointikielen tyyppinen kieli. Siinä ohjelma kirjoitetaan ylhäältä alaspäin ja luetaan ohjelmankierrossa samassa järjestyksessä. Ylinnä olevat käskyrivit luetaan ensin ja alimmat viimeisenä. STL on tiivis mutta hidaslukuinen suhteessa graafisiin ohjelmointikieliin. Lukuarvojen muutokset toisiinsa (konversio) on hieman helpommin toteutettavissa kuin graafisilla kielillä. STL:ssa voidaan hyödyntää myös keskusyksikön muistiakkuja. Niiden määrä on kaksi tai neljä riippuen siitä, mikä keskusyksikön tyyppi on käytössä. STL on hyvä yksinkertaisissa komennoissa ja matemaattisissa toiminnoissa. STL on hieman monimutkainen käyttää loogisissa ohjauksissa eli esimerkiksi sekvenssiohjauksessa.

FBD (Function Block Diagram) on graafinen hieman loogisten operandien piirrosmerkkimuotoinen ohjelmointikieli koostuen “laatikoista”, joihin syötetään halutut tulot ja jotka tuottavat loogisen lähdön. Se kehitettiin vanhemman käskylistamuotoisen ohjelmointikielen rinnalle, tai ehkä jopa korvaamaan se, ohjelmoitavissa logiikoissa. Sen etuina on merkittävästi helppolukuisempi loogisten ohjausten toteutus ja luku kuin käskylistamuotoisessa kielessä. Tämä lyhentää esimerkiksi vianetsintäaikaa teollisuuden sovelluksissa. FBD:ssa, kuten muissakin graafisissa kielissä, matemaattisten toimintojen teko vaatii hieman enemmän muistia ja tilaa kuin käskylistamuodossa. Aikaisemmin mainitut lukuarvomuutokset tarvitsevat välitallennuksen halutulle muistipaikalle, mihin yleisimmin käytetään väliaikaisia, yhden ohjelmankierron ajan tilansa säilyttäviä TEMP-muistialueita. FBD soveltuu melko hyvin esimerkiksi sekvenssiohjaukseen, ollen kuitenkin tässä hieman LADia vaikeampi lukuisempi. FBD on hyvä perusohjelmointikieli, joka on jokaisen logiikkaa ohjelmoivien syytä osata johtuen loogisten operandien piirrosmerkkien samankaltaisuudesta.

LAD (Ladder) on graafinen ohjelmointikieli, mikä muistuttaa hyvin paljon sähköpiirustuksista tuttua piirikaaviota. LAD-kielessä esimerkiksi loogiset JA/TAI-funktiot tehdään yhdistelemällä piirikaavioiden kosketin-piirrosmerkin kaltaisia symboleita halutulla tavalla. Invertointi eli tilan kääntö toteutetaan kosketinpiirrosmerkkien normaalisti avoin- ja normaalisti suljettu -tyyppejä käyttämällä. LAD poikkeaa graafisesti FBD:stä, mutta on ohjelmoitavuudeltaan melko samankaltainen, vaikkakin esimerkiksi sekvenssien tekeminen on hieman suoraviivaisempaa ja siten mahdollisesti hieman nopeampaa.

Kuvassa 5 on esitetty sama ohjelma kolmella yleisimmällä ohjelmointikielellä.



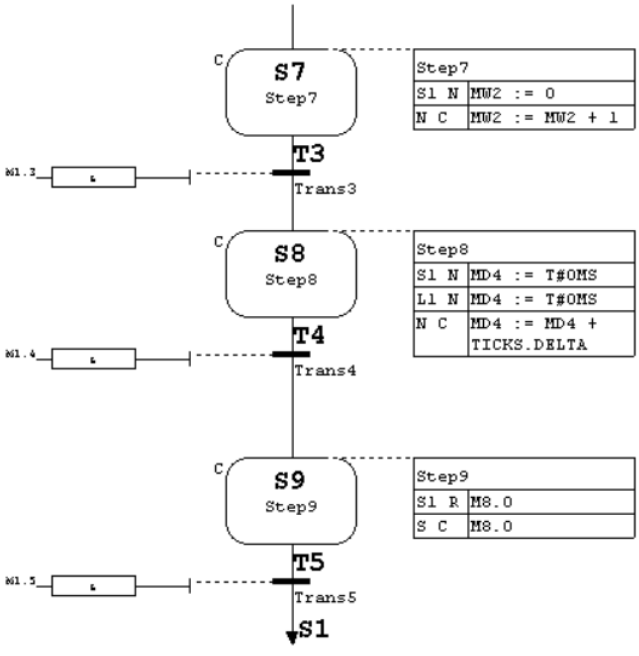
Kuva 5. Kolmen yleisimmän ohjelmointikielen vertailu (Lothar Kern, European Southern Observatory)

SCL (Structured Control Language) muistuttaa vielä enemmän Pascal-ohjelmointikieltä kuin STL. SCL mahdollistaa helpoimman tavan tehdä matemaattisia laskutoimituksia ohjelmoitavassa logiikassa, mutta vastaavasti esimerkiksi kappaletavaran ohjaussekvenssien toteuttaminen on hankalampaa kuin graafisissa kielissä. Yksi merkittävä vaikkakin harvoin tarvittu hyöty SCL:ssä on verrattuna graafisiin. Sitä voidaan melko suoraan kopioida eri logiikoiden välillä, mahdollisesti lähes ilman muokkaustarpeita. Kuvassa 6 on esimerkki SCL-muotoisesta ohjelmasta, jossa tuodaan kaksi sanan mittaista kokonaislukua toimilohkoon. Lohkoa kutsuttaessa se tuottaa kaksi sanan mittaista kokonaislukua, joista toinen on kahden tuodun luvun summa ja toinen on tuotujen lukujen erotus.

```
FUNCTION_BLOCK FB34

VAR_INPUT
    IN1: INT; //Input variable
    IN2: INT;
END_VAR

VAR_OUTPUT
    OUT1: INT; //Output variable
    OUT2: INT;
END_VAR

OUT1:= IN1 + IN2;
OUT2:= IN1 - IN2;

END_FUNCTION_BLOCK

DATA_BLOCK DB34 FB34 //Instance DB of FB34

BEGIN
END_DATA_BLOCK
```

Kuva 6. Esimerkki SCL muotoisesta ohjelmasta (Siemens)

S7-Graph on graafisin ohjelmointikieli ja muistuttaa visuaalisesti hyvin paljon vuokaaviota, kuten kuvasta 7 voidaan nähdä. Siinä haluttu toiminto toteutetaan antamalla prosessin etenemisjärjestyksessä ohjaustietoja ja ehtoja, sekä määrittämällä esimerkiksi odotusaikoja eri toimintojen välillä. S7-Graph soveltuu oikein hyvin lähes kaikkeen sekvenssiohjaukseen.



Kuva 7. Esimerkki S7-Graph muotoisesta ohjelmasta (Siemens 2019)

Yhteenvetona voidaan todeta, että kaikilla ohjelmointikielillä voidaan tehdä kaikki halutut asiat mutta tekstipohjaiset ovat hieman parempia lukuarvojen käsittelyssä

ja graafiset muissa ohjauksissa. Eri ohjelmointikieliä yhdistämällä päästään tehokkaimpaan lopputulokseen, mutta se ei välttämättä ole järkevintä turhan monimutkaisuuden takia.

3.7 Muuttujat

Muuttujilla tarkoitetaan erilaisia ohjelmassa käsiteltäviä arvoja ja tiloja, jotka nimensä mukaisesti voivat muuttua tai joiden arvoon ja tilaan voidaan vaikuttaa. Muuttujien sijoittamiseen voidaan käyttää logiikassa eri muistialueita ja muistityyppejä, kuten esimerkiksi pysyvät ja väliaikaiset muistit.

3.8 Muisti ja muistialueet

Eri ohjelmoitavissalogiikkamalleissa on eri määrä muistia käytössä, ja tarvittavat kapasiteetit liikkuvat kilo- ja megatavuluokassa, eli tarve on hyvin paljon pienempi kuin mihin tietotekniikassa on totuttu. Muisteja on erilaisia, joita voidaan hyödyntää tarpeeseen sopivasti. Pysyvät muistit säilyttävät tilansa ohjelmankierrosta riippumatta, ellei niiden tilaa erikseen muuteta, mutta väliaikaiset muistit nollautuvat ohjelmankierron jälkeen.

3.9 Kommunikointi servojen kanssa

Siemens käyttää omia Telegramiksi nimeämiään väyläohjaussanomia ja ohjelmoijan vapaampaan parametrointiin soveltuvia BICO-sanomia. Erityyppiset Telegram-sanomat soveltuvat eri sovelluksiin ja esimerkiksi rekistereiden säädössä tarvitaan paikoitusajoa nollakohtaan ajossa, jolloin ohjaussanomaksi valitaan Telegram 111. Kyseisessä sanomassa on kaksitoista tulosanaa ja kaksitoista lähtösanaa (HMK Wiki 2013). Nämä tarkoittavat servon ohjaussanoja (Control Word) sekä tilasanoja (Status Word). Kuvasta 8 on nähtävissä tulo ja lähtösanojen sisältö.

Receive data

PZD Number	Reference	Description	Units
1	STW1	Main control word	BOOL
2	SATZANW	Traversing block selection	BOOL
3	POS_STW	EPOS control word	BOOL
4	STW2	Control word 2	BOOL
5	OVERRIDE	EPOS velocity override	Percentage - 16384 DEC or 4000 HEX = 100% of velocity setpoint
6 7	MDI_TARPOS	Position setpoint	LU
8 9	MDI_VELOCITY	Velocity setpoint	1000 LU/min
10	MDI_ACC	Acceleration override	Percentage - 16384 DEC or 4000 HEX = 100% of p2572
11	MDI_DEC	Deceleration override	Percentage - 16384 DEC or 4000 HEX = 100% of p2573
12	USER	User definanble	INT/WORD

Send data

PZD Number	Reference	Description	Units
1	ZSW1	Main status word	BOOL
2	POS_ZSW1	EPOS status word	BOOL
3	POS_ZSW2	EPOS status word 2	BOOL
4	ZSW2	Status word 2	BOOL
5	MELDW	Drive status	BOOL
6 7	XIST_A	Actual position	LU
8 9	NIST_B	Actual speed	Percentage - 40000000 HEX or 1073741824 DEC = 100% of p2000
10	FAULT_CODE	Fault code	INT
11	WARN_CODE	Alarm code	INT
12	USER	User definanble	INT/WORD

Kuva 8. Telegram 111 (HMK Wiki 2013)

Yleistäen voidaan sanoa, että ohjaussanoilla liikutetaan servoa ja tilasanalla tiedetään esimerkiksi servon asema, nopeus ja muut tiedot. Servojen ohjaus- ja tilasanojen lisäksi tarvitaan servojen ohjausyksikölle oma sanomansa. Siihen valitaan Telegram 2, joka on kahden tulo- ja kahden lähtösanan mittainen. Ohjaus-

yksikön erikseen ohjaaminen ei ole edes täysin välttämätöntä, mutta ohjausyksiköltä voidaan esimerkiksi lukea yksikön vikakoodit ja muita tilatietoja. Ohjausyksikkö ja servo-ohjaimet kommunikoivat keskenään sisäisellä väyläliikenteellä, jolloin esimerkiksi servo-ohjaimien vikatiedot ilmenevät suoraan niiden omista tilasanoistaan.

Ohjelmoitavalla logiikalla on oma IP-osoite tai vaihtoehtoisesti oma Profibus-osoite, jos kyseistä väylää käytettäisiin kommunikointiin. Servojen puolella servojen ohjausyksikölle asetetaan oma IP-osoitteensa, mutta servo-ohjaimet eivät omaa osoitetta tarvitse. Kommunikaatio hoituu servolaitteiden sisäisenä.

On huomioitava, että ohjelmoitavan logiikan osoitekonfiguraatio vastaa servojen osoitekonfiguraatiota, jolloin sanomaliikenne on mahdollista kyseisten laitteiden välillä.

4 Sivurekisterin automaattisen säädön toteutus

4.1 Lähtötiedot

Tarvittavan rekistereiden sivuttaissuuntaustiedon antaa BST Eltromatin Registar -laitteisto. Jokaista painoyksikköä kohden Registar antaa digitaalilähtötiedot, jotka luetaan sivurekisteriä ohjaavassa logiikassa digitaalitulokortin avulla. 24VDC pienoisjännitetasoisia tietoja tulee kaksi yhtä painoyksikköä kohti, joista toinen vastaa rekisterin siirtotarvetta linjan suunnassa vasemmalle (OP - OPerator side - operaattori-/hoitopuoli) ja toinen linjan suunnassa oikealle (GR - GeaR side - käyttöpuoli). Muu rekisterinohjaukseen liittyvä ohjaus tehdään rekistereiden ohjaukseen suunnitellussa ohjauskeskuksessa siihen liittyvine kenttälaitteineen.

4.2 Sähkö- automaatiosuunnitelma

Järjestelmää varten suunniteltiin oma ohjauskeskus sisältäen Siemens 315 -sarjan ohjelmoitavan logiikan keskusyksikön (CPU Central Processing Unit), jossa on kaksi PN (Profinet) -väyläpaikkaa ja yksi MPI/DP-väyläpaikka. Logiikan sisäisesti kumpikin Profinet -väyläpaikka on samassa IP-osoitteessa eli logiikan portit toimivat Ethernet-kytkimenä ja siksi logiikan kautta voidaan esimerkiksi jatkaa väylää eteenpäin tai olla muutoin yhteydessä kahteen eri väyläpisteeseen.

DP/MPI-väyläportti voidaan konfiguroida joko Profibus DP -väyläksi tai hitaammaksi MPI-väyläksi. Tässä käytössä kaikki väyläkommunikointi hoidetaan Profinetin kautta, jolloin DP/MPI-väyläportti jää mahdollisiin tulevaisuuden laajennuksiin. Kyseiseen keskusyksikkötyyppiin päädyttiin tehtaan varaosavaraston optimoinnin vuoksi. Keskusyksikkötyyppi on käytössä myös tehtaan kahdessa muussa linjalaitteessa, joten se löytyy myös valmiiksi varaosavarastonimikkeenä. Keskusyksikkö on myös riittävän tehokas suorittamaan sivurekisteriohjelman halutulla ohjelmankiertonopeudella.

Ohjauskeskus sisältää ohjelmoitavan logiikan lisäksi keskuksen sisäisen sähkönjakelun johdonsuoja-automaatteineen, pienoisjännitelähteen, servo-ohjaimet, ohjausreleistyksen, servo-ohjaimien syöttökontaktorit ja tarvittavat riviliittimet. Keskuksen pääkojeena toimii 63A-kytkinvaroke, johon laitteiston tarvitseman virran ja syötön selektiivisyyden vuoksi valittiin neljänkymmenen ampeerin kahvasulakkeet.

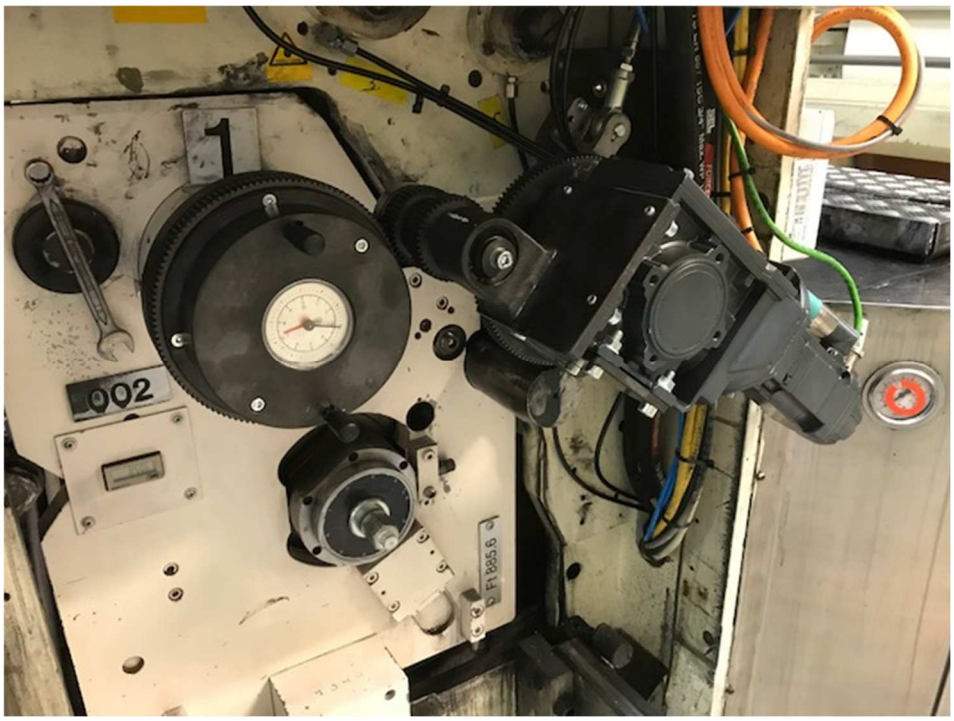
Käyttöliittymäksi valittiin Siemensin KTP 1200 PN -kosketuspaneeli ja osa toiminnoista voidaan suorittaa painonapeilla painoyksiköiden etureunasta.

4.3 Mekaaninen toteutus

Painokasetin painopellin rummussa on kuuden millimetrin sivusuuntainen säätöalue, jota voidaan säätää käsipyörällä. Pyörässä on analoginen asennonosoituskello. Tämän käsiohjauksen rinnalle toteutettiin kuvien 9 ja 10 mukainen servolta mekaanisesti hammaspyörin välitetty voima säätämään painorumpua sivusuunnassaan. Kuvasta 9 nähdään painoyksiköstä ulos otettu painokasetti säätömekanismeineen, jossa ovat hammaspyörä ja servo ympyröitynä punaisella.

Kuva 9. Painokasetti ulkona painoyksiköstä (Tetra Pak Production Oy)

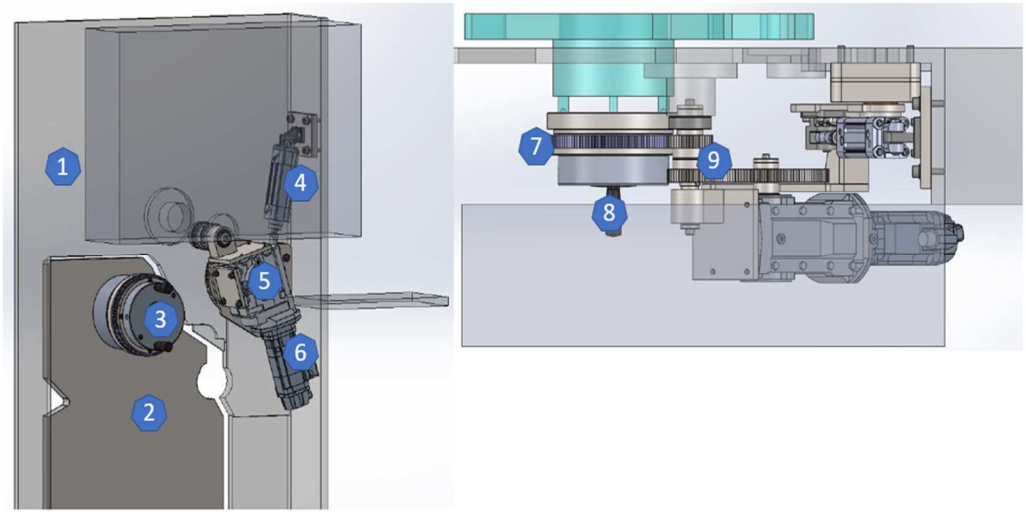
Kuvasta 10 nähdään painoyksikön sisällä oleva painokasetti, missä on säätö-servo ajoasennossaan.



Kuva 10. Painokasetti sisällä painoyksikössä (Tetra Pak Production Oy)

4.4 Säätölaitteiston mekaaniset pääkomponentit

Servolla toteutettu automaattinen säätö jakautuu mekaanisesti kahteen kokonaisuuteen: painokasetin puoleisiin komponentteihin ja painoyksikön puoleisiin komponentteihin. Painokasetti otetaan ulos painoyksiköstä painopeltien vaihdon ja muun huollon, sekä tuotekoon vaihdon ajaksi. Tämän vuoksi painokasetin ja painoyksikön välillä pitää olla mahdollisimman vähän irrotettavia tai liitettäviä komponentteja. Laitteiston mekaaniset pääosat ovat esitetty kuvassa 11.



Kuva 11. Laitteiston mekaaniset pääosat (Tetra Pak Production Oy).

Kuvassa 11 olevat osat ovat: 1-painoyksikkö, 2-painokasetti, 3-painokasetin sivuttaissäätölaitteisto, 4-servon nostosylinteri, 5-servomoottorin vaihteisto, 6-servomoottori, 7-painokasetin säätölaitteiston hammasratas, 8-painokasetin säätölaitteiston käsikahvat ja 9-sivurekisterin säädön mekaaninen välitys.

4.5 Ongelmakohdat

Mekaanisen voiman välittävät hammaspyörät liikkuvat hampailleen aina painokasettia siirrettäessä, joten hammaspyörien hampailleen meno täytyi varmistaa. Tämä toteutettiin pyörittämällä painoyksikön puoleista voiman välittävää hammasratasta hitaasti laskuvaiheessa, jotta hammaspyörien hampaat asemoituvat oikein painokasetin hammasrattaan kohdatessaan.

5 Ohjelma ja ohjaus

5.1 Toiminnalliset vaatimukset

Sivurekistereiden säätölaitteiston tulee vastata Registarin tuottamaan ohjaussignaaliin mahdollisimman nopeasti, mutta kuitenkin siten, ettei tarpeettoman nopeasti ohjata. Tällöin on vaarana, että ajetaan halutusta kohdistusetäisyydestä yli, johon Registar reagoi ja yliohjaa palaamaan edelliseen suuntaan ja säätö alkaa värähtelemään. Servojen kiihdytysaika, nopeus ja hidastusaika pitää tuotantoajotestien perusteella säätää optimaaliseksi, jolloin säätö on mahdollisimman nopea ilman värähtelyn mahdollisuutta.

Sivurekistereille tarvitaan referenssipiste, johon ajamalla voidaan optimoida kullekin yksikölle mahdollisimman tarkka nollakohta vastaamaan alkutilannetta aina ennen tuotannon aloittamista. Tällä toiminnolla varmistutaan siitä, että painopellit ja -kasetit ovat mahdollisimman samassa linjassa ja tuotannon alettua ensimmäisen säädön tarve on mahdollisimman pieni, ja jolloin myös tuotannonhävikki minimoidaan. On huomioitava, että linjaa käyttävien operaattoreiden suorittama painopeltien kasettiin asemointi on hyvin standardoitu, joten oikealla nollakohdalla on merkitystä. Jokaisen yksikön kaseteilla on hieman erisuuruisia mekaanisia välyksiä toisiinsa nähden, jolloin nollakohdatkaan eivät täysin vastaa toisiaan, vaan ovat yksilöllisiä. Pyrkimys on kuitenkin löytää jokaiselle yksikölle oma optimaalinen nollakohtansa tuotannon aloitusta varten.

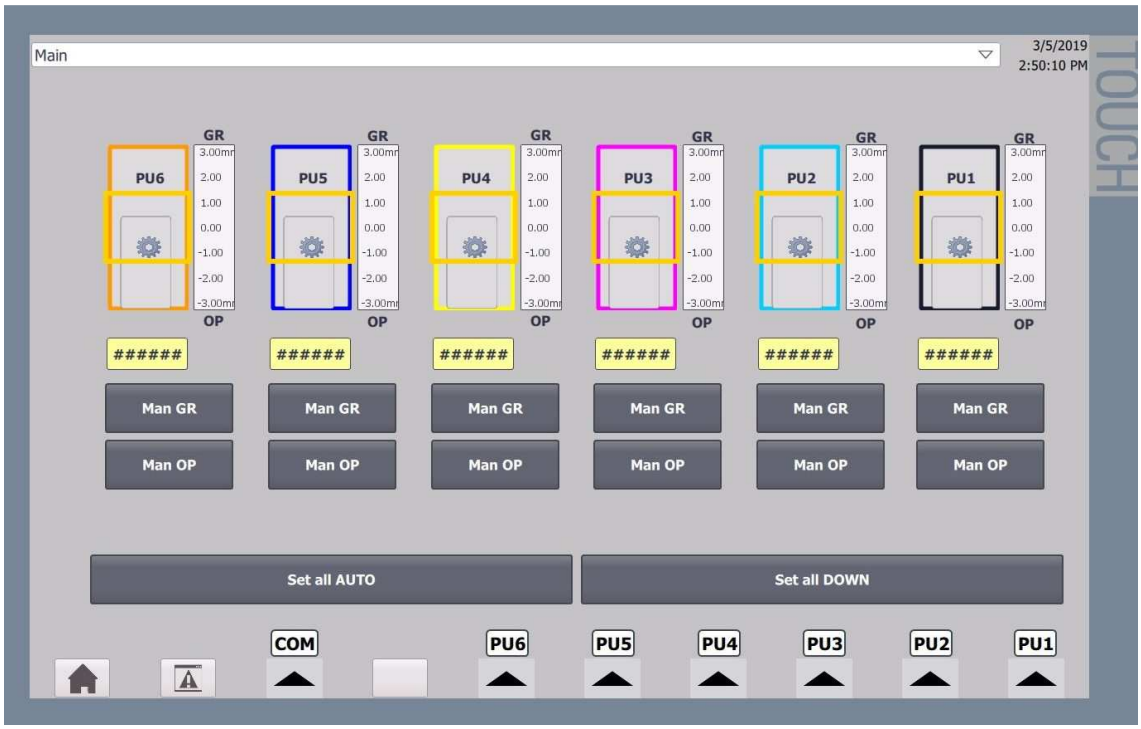
Laitteiston pitää olla myös säädettävissä käsin ajamalla painoyksiköltä painonapeilla sekä myös pääkäyttöpaikalla sijaitsevalta kosketuspaneelilta. Automaattista säätöä ja käsiohjaustilaa ei voida käyttää yhtä aikaa, joten tarvitaan myös automaatti/käsi-valintamahdollisuus painoyksiköille ja kosketuspaneelille. Jokaiselle yksikölle tehdään omat valintansa, jolloin on mahdollista säätää osaa yksiköistä manuaalisesti ja osaa automaattisesti. Tämän lisäksi tarvitaan täysin manuaalinen säätö sähköisensäädön rikkoutumisen varalle, jolloin voidaan painokasetin käsipyörästä kääntämällä suorittaa rekisterinsäätö kuten aiemminkin.

Rekisterisäädössä yhtä painoyksikköä sanotaan Keycoloriksi eli kyseinen yksikkö pysyy paikallaan, mutta muut yksiköt säädetään sen suhteen. Jos kaikkia yksiköitä säädettäisiin itsenäisesti, niin säädöstä tulisi lähes mahdoton toteuttaa

käytännössä. Tällöin järjestelmä alkaisi värähtelemään, ellei säätöä toteutettaisi kohdistamalla referenssi johonkin virtuaalilinjaan eli esimerkiksi säätöalueen laskennalliseen keskikohtaan. Säädön värähtelyllä tarkoitetaan tässä tapauksessa sitä, ettei haluttua kohdistusta ikinä saavuteta, koska haluttu referenssikohta muuttuisi jatkuvasti. Keycolor-yksikkö toimii referenssikohtana ja muut yksiköt kohdistetaan sen mukaisesti. Painolinjan ensimmäinen painoyksikkö on Keycolor ja laitteistoa on säädettävä sen suhteen. On huomioitava kuitenkin, että mekaanisten välysten ja epätarkkuuksien vuoksi Keycolor ei voi olla täysin säätämättömissä. Jos yksikin muu painoyksikkö lähestyy mekaanista rajaa, niin silloin Keycoloria täytyy vastaavasti pystyä siirtämään vastakkaiseen suuntaan, jolloin saadaan luotua lisää säätövaraa kyseiselle yksikölle. Keycolorin säätäminen kesken ajon pitää toteuttaa mahdollisimman hitaasti, jolloin muut yksiköt voivat seurata sitä painolaadun kuitenkin huonontumatta Keycolorin säädön ollessa käynnissä.

5.2 Käyttöliittymät

Käyttöliittymiä on kaksi, Siemens KTP1200 Basic PN -kosketuspaneeli ja painoyksiköissä sijaitsevat painonapit. Kosketuspaneelista voidaan tarkkailla suojaverkkojen asentoa, servojen paikkaa, laitteiston käyttötilaa ja rekisterin säädön asemaa. Lisäksi paneelista voidaan asettaa kullekin painoyksikölle oma nollakohtansa. Kosketuspaneelin pääsivu on nähtävissä kuvasta 12.



Kuva 12. Kosketuspaneelin pääsivu

Painoyksiköiden painonapeilla voidaan suorittaa oleellisimmat toiminnot, kuten käyttötilan valinta ja servon nosto sekä lasku.

5.3 Kommunikointi

5.3.1 Digitaalitulot ja -lähdöt

Jokaiselle yksikölle Registar lähettää 24VDC digitaalisignaalit, joilla automaattisäätö tapahtuu. Digitaalituloihin liittyvät myös painoyksiköiden hallintalaitteet eli painonapit. Painoyksiköissä on painonapit servon nostamiselle ja laskemiselle, automaatti- ja käsiajovalinnalle, nollapisteeseen ajamiselle ja sivurekisterin käsiajoille. Linjan päälogiikka liittyy digitaalituloilla- ja lähdöillä rekisterinsäätölogiikkaan. Päälogiikalta tuodaan linja käynnissä-, tuotteen kokovalinta- ja painoyksiköiden suojaverkon asemoiden tiedot rekisterilogiikalle. Servoja liikuttelevien paineilmasylintereiden sylinterirajatiedot tuodaan myös rekisterilogiikalle. Jokaisen painoyksikön sylinterissä on kolme Reed-tyyppistä sylinterirajaa, jotka ilmaisevat servon yläasennon eli kotiaseman, sekä kahden eri tuotekoon painokasetin ohjausasennot.

Ohjauskeskuksen sisäisistä tilatiedoista logiikalle tuodaan servo-ohjainten syöttökontaktoreiden tilatieto. Digitaalilähdöillä ohjataan ohjausreleiden kautta servojen aseman paineilmasylintereitä sekä painonappien valoindikointeja.

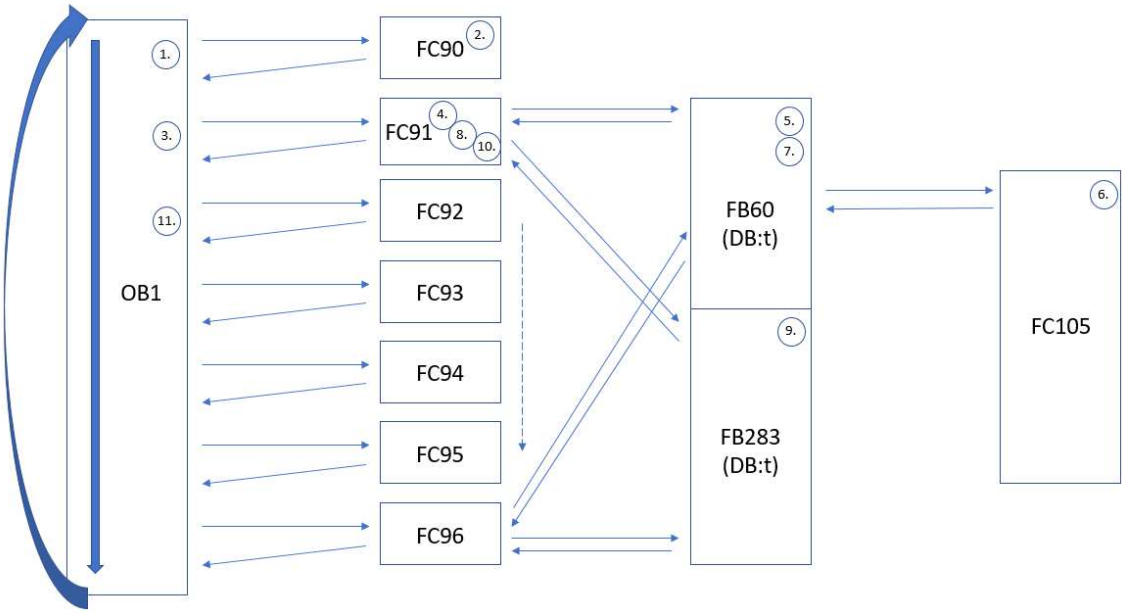
5.3.2 Väylätulot ja -lähdöt

Profinet-väylän kautta logiikalta servoille lähetetään ohjaussanoissa servojen sallinta-, käynnistys-, jog-, nopeus-, kiihdytysnopeus-, hidastusnopeus-, positio- ja vikojen kuittaustiedot. Servoilta logiikalle tuodaan servon todellinen positio, positio saavutettu, vika ja valmiustiedot.

5.4 Ohjelman rakenne

5.4.1 Ohjelmankierto

Pääohjelmankierrossa (OB1) kutsutaan jokaisen painoyksikön ohjaukseen luotuja toimintoja (FC), joiden sisällä suoritetaan kuvan 13 mukaisesti järjestyksessään ohjauksien tarvitsemat toiminnot.



Kuva 13. Pääohjelmankierto

Kuvasta 13 nähdään, miten ylhäältä alas luettaessa organisaatiolohko OB1 kutsuu toiminnot, jotka kutsuvat niiden sisälle määritellyt toimilohkot ja toiminnot.

Kuvaan ympyröidyt järjestysnumerot kuvaavat ohjelmansuoritusjärjestystä, mikä toistuu kaikkien painoyksiköiden ohjausten (FC91-FC96) kesken samanlaisesti.

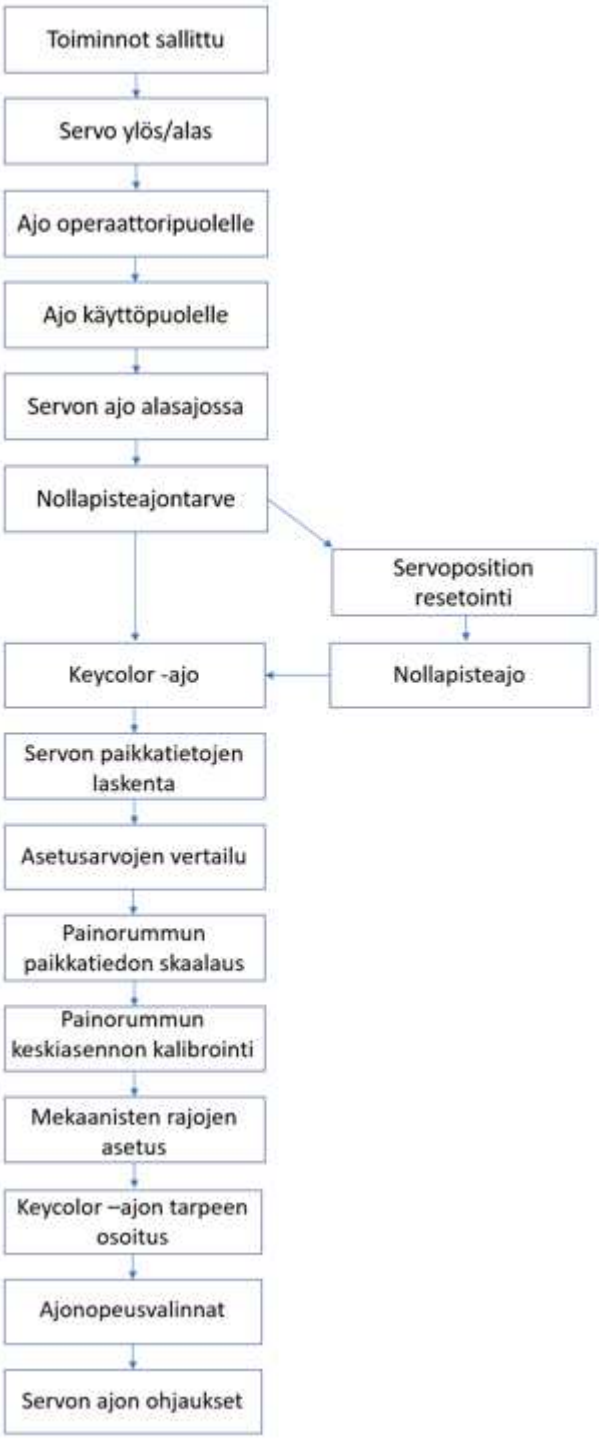
Tämän lisäksi erillään pääohjelmankierrosta otettiin käyttöön organisaatiolohko OB32, jolla kutsutaan tuhannen millisekunnin välein jokaisen yksikön nollapisteen määrittelevää toimintoa, ja organisaatiolohko OB34, jolla kutsutaan kahdensadan millisekunnin välein painokasetin todellisen paikan anturitiedon keskiarvolaskentaa. Nämä toiminnot eivät ole niin aikakriittisiä kuin pääohjelmankierrossa OB1 oleva servojen ohjaus, joten ne voidaan kutsua huomattavasti harvemmin. Tämän lisäksi voidaan erikseen suorittaa jokainen yksikkö omassa aikakutsukierrossaan, mikä osaltaan vähentää pääohjelmankiertoon tulevaa viivettä kutsuttaessa aikaperusteista organisaatiolohkoa. Tietyn ajan välein suoritettavat organisaatiolohkot keskeyttävät pääohjelmankierron, ja suorituksen jälkeen palataan jälleen pääohjelmankiertoon siihen kohtaan, joka keskeytyi.

Näillä toimin pyritään pääsemään ohjelmankiertoajassa alle kahteen millisekuntiin, mikä on hyvä tähän ohjaukseen. Ilman erillisiä aikaperusteisia nollapisteen ja keskiarvolaskennan määrittämiskutsuja, ohjelmankiertoaika hidastuu arviolta noin kaksi millisekuntia.

Ohjelmankiertoon ja suorittamiseen osallistuvien organisaatiolohkojen lisäksi ohjelmaan lisättiin Siemensin omat vikaorganisaatiolohkot: OB80 – ohjelmankiertoajan ylitys, OB82 – diagnoosikeskeytys, OB85 – prioriteettiluokkavirhe, OB86 – asemavika, OB87 – kommunikaatiovika ja OB121 - ohjelmointivirhe. Jos jokin edellä kuvatuista virheistä tai vioista tapahtuu, niin kyseiselle vialle tarkoitettu OB estää logiikan menemisen STOP-tilaan. Lisäksi voidaan halutessa ohjelmoida jokin haluttu toiminto, kun kyseinen vika havaitaan.

5.4.2 FB60 - Servo-ohjauksien toimilohko

Keskeisin sivurekisterin säätämiseen tehty toimintalohko on FB60. Se on tehty asettamaan halutuilla ehdoilla servo-ohjaus mahdolliseksi (sallinta), ohjaamaan servomekaniikan liikuttelua kotiaseman ja kasetinsäädön välillä, ajamaan servoa nolla-asemaan, suorittamaan Keycolor-ajo, sekä asettamaan tarvittavat ohjausbitit servokommunikoinnin ohjaussanaa varten. Kuvaan 14 on koottu yksinkertaistettuna keskeisimmät toimilohko FB60:n toiminnot.

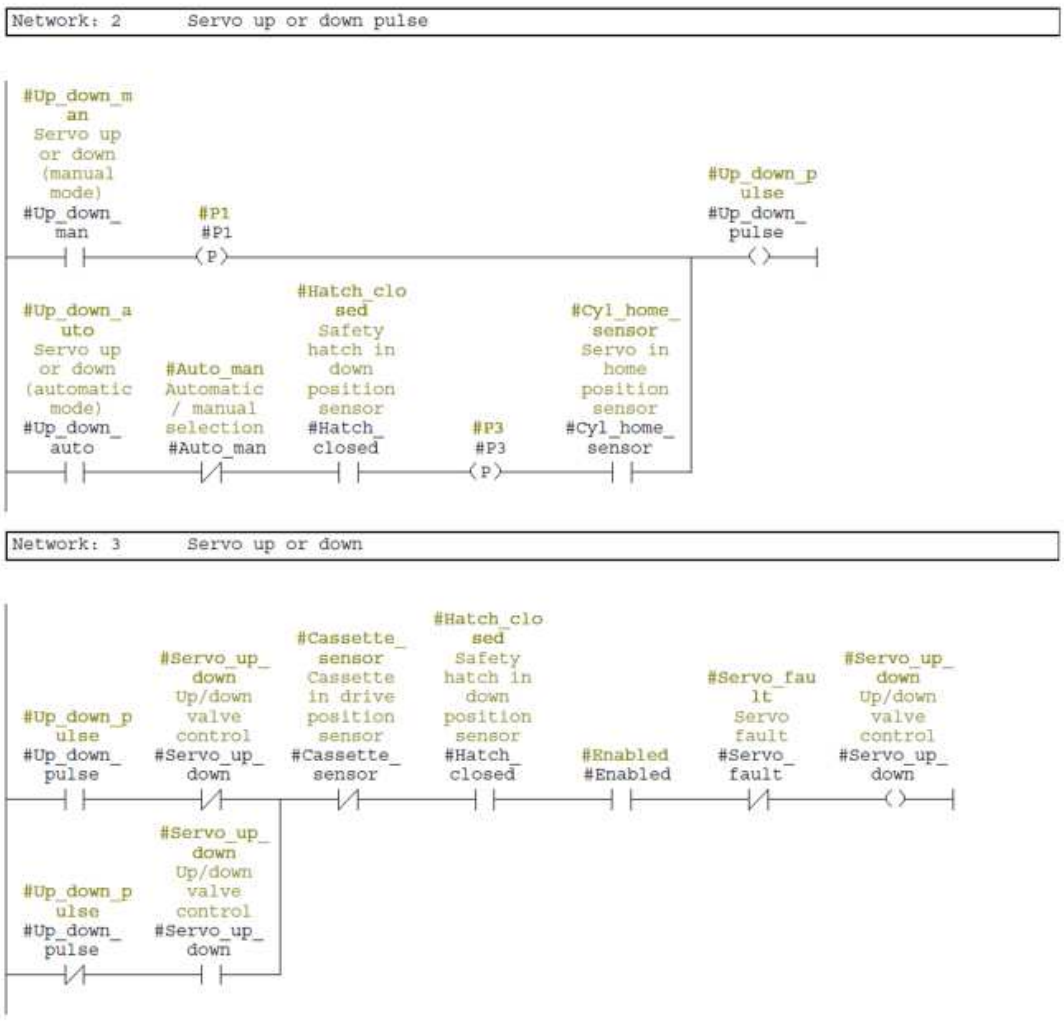


Kuva 14. Toimilohkon FB60 toiminnot

Toimilohkon tulorajapintaan tuodaan ohjauksen sallintaa varten servon turvakytkimen tilatieto ja yleinen päällä-tieto, jossa käytetään linjan turvapiirin tilatietoa. Toimilohkossa nämä käsitellään loogisena JA-operandina eli kummatkin tiedot

päällä ollessaan asettavat toimilohkon sisäisen staattisen muistibitin päällä -tilaan. Tätä bittiä käytetään toimilohkon sisällä muissa ohjauksissa yhtenä sallintaehtona.

Servomekaniikan liikuttaminen paineilmasylinterillä kotiaseman ja ohjausaseman välillä on mahdollista painoyksiköiden painonapeilla tai automaattisesti, kun automaatti-tila on valittuna kosketuspaneelista tai painoyksikön painonapista. Servomekaniikan koti- ja säätöasemaan liikuttelua varten tehtiin kahdesta piiristä koostuva kiikku-toiminto. Ensimmäisen piiriin tuodaan halutut ehdot, joiden nousevan reunan tunnistus ohjaa väliaikaisen muistibitin yhden ohjelmankierron ajaksi päälle-tilaan. Tämä väliaikainen TEMP-muistibitti tuodaan kahtena rinnakkaisena tulona seuraavaan piiriin, jossa piirin lähdölle asetettu staattinen muistibitti liitetään kyseisten tulojen jälkeen sarjaan siten, että toisen tulon tila on invertoitu. Myös toinen edellisestä piiristä tuotu tulo täytyy invertoida. Kun ensimmäisen piirin ehdot täyttyvät ja muistibitti asettuu yhden ohjelmankierron ajaksi päälle, niin sen tila luetaan seuraavassa piirissä. Kyseisen piirin lähdön tilan ollessa pois päältä, täyttyy lähdön tilan invertoidulta puolelta ehdot, jolloin lähdön staattinen muistibitti asettuu päälle-tilaan ja piiri jää omaan pitoon. Seuraavalla ensimmäisen piirin yhden ohjelmankierron päällä-tilalla katkeaa toisen piirin omapito ja kyseisen piirin staattinen muistibitti asettuu pois-tilaan. Tämä toiminto on mahdollinen, kun huomioidaan ohjelman suoritusjärjestys ylhäältä alas kuvan 15 esittämän kahden ohjelmapiirin avulla. Toisen piirin staattista muistibittiä käytetään paineilmasylinterin ohjauksen lähtöosoitteen tilan ehtona.



Kuva 15. Servon noston ja laskun kiikku -toiminnon logiikkapiirit ohjelmasta (Tetra Pak Production Oy)

Rekisterin kohdistus eli servon liikuttaminen tapahtuu ryömittämällä servoa (JOG). Ryömittämisellä tarkoitetaan koneen hidasta ajoa. Usein ryömittämistä käytetään myös koneiden huoltotoimenpiteiden yhteydessä, jos koneen liikuttaminen toimenpiteiden aikana on välttämätöntä. Servo on parametroitu paikoitusohjaukseen soveltuvaksi nollapisteajon vuoksi, joten muuhun kuin paikoitusliikuttamiseen soveltuu hyvin ryömittäminen. Servon erisuuntiin ryömittämistä varten tehtiin kaksi piiriä, joihin asetettiin halutut ehdot ja piirin lähtötila on staattisena muistina. Sinamics servo-ohjain on tarkka missä järjestyksessä ohjaussanan bitit asetetaan, joten ohjelmaan asetettiin kymmenen millisekunnin mittainen viive viivästyttämään ohjaussanan bitin DBX173.6 asettumista järjestyksessään viimeisenä ja varmistamaan haluttu toiminta ilman servon virheilmoituksia. Kyseinen

bittijärjestys voitaisiin tehdä myös hyödyntäen ohjelmankiertojärjestystä, mutta väyläviiveet ja mahdolliset ohjelmointivirheet huomioiden, päädyttiin kyseiseen kymmenen millisekunnin viiveeseen. Toimilohkossa ei ohjausbiteistä muodostettu ohjaussanaa servon väylälähtöä varten, vaan ohjausbitit eriteltiin ja käsiteltiin yksittäisinä, koska Sinamicsin parametrointiin käytettävässä Starter-ohjelmassa voidaan ohjausbittien tiloja tarkkailla yksittäin tarkoitukseen soveltuvassa käyttöönoton vianetsintätoiminnossa. Tämä helpottaa löytämään mahdolliset puuttuvat ohjausbitit ja varmistumaan, että kaikki ohjataan oikeassa järjestyksessä. Ohjaus- ja tulosanojen lukuarvot voidaan nähdä esimerkiksi monitoroimalla halutun servon ohjaussanaan käytettävän muistilohkon sanoja kokonaisuutena.

Servomekaniikkaa alaspäin laskettaessa täytyy varmistua siitä, että mekaniikan hammasrattaat kohdistuvat oikein. Tämä ratkaistiin ryömittämällä servoa hitaasti laskuvaiheessa aikaperusteisesti kolmen sekunnin ajan. Aikaperusteiseen pysäytykseen päädyttiin, että vältetään sylinterirajasta riippuvainen ryöminnän pysäytys. Jos servomekaniikan ala-asennon tunnistava sylinteriraja on rikki tai jostain mekaanisesta syystä rajaa ei saavuteta, niin on vaarana ryömittää servoa niin kauan, että rekisterinsäädön mekaaniset rajat saavutetaan. Tällöin syntyy jumitilanne ja mahdollinen mekaaninen vaurio.

Nollapisteeseen ajon sallintaehdoksi asetettiin painokasetin ala-asennossa käynti ja turvaverkon nosto. Näin ollen voidaan päätellä, että kasettia on mahdollisesti liikuteltu manuaalisesti ja edellinen tuotantoajon aikainen asema ei enää ole kohdallaan. Vaihtoehtoisesti voitaisiin nollapisteajo tehdä aina kun esimerkiksi linja käynnistetään uudelleen, mutta tällöin todennäköisesti tehdään tarpeettoman pieniä nollapisteajoja, jos kasettia ei olla manuaalisesti liikuteltu välillä. Kun sallintaehto on aktiivinen, painokasetti yläasennossa, verkko alhaalla ja servo lasketaan säätöasentoon, niin silloin täyttyvät nollapisteeseen ajon ehdot. Ajon aluksi nollataan servo-ohjaimen paikkatieto käyttämällä ohjausbitti DBX177.1 päällä. Näin saadaan servon referenssipiste halutuksi arvoksi, eli nollaksi, ja nollapisteajo on loogisempi toteuttaa nollasta alkaen. Tämän jälkeen siirrytään itse paikoitusajoon, jossa aluksi servolle kerrotaan haluttu paikka pituus-

yksikköinä LU (Lenght Unit) ohjauskaksoissanalla DBD182. Servo-ohjaimelle esimerkiksi paikoitusetäisyydeksi ilmoitettaessa lukuarvo kaksituhatta, tarkoittaa se sitä, että haluttu positio on kaksi millimetriä servon nollakohdasta positiiviseen suuntaan. Negatiivinen arvo tarkoittaa vastakkaista suuntaa. Siemensin Sinamic-sissa yksi pituusyksikkö on yksi mikrometri. Näin tarkkaa paikoitusta yleensä ei tarvita normaaleissa teollisuussovelluksissa, mutta pituusyksiköt ovat mielekästä ilmoittaa mikrometreissä, koska näin vältytään käyttämästä desimaaleja. Alle mikrometrin todelliseen tarkkuuteen ei teollisuussservomekaniikalla ole mahdollista päästä. Tarvittava paikoitusarvo määriteltiin halutun nollapistearvon ja todellisen paikan erotuksena, ottaen huomioon myös suunnan. Paikoitusajo poikkeaa ryömityksestä asetettavien ohjausbittien osalta siten, että servon valmiuteen asettamisen (enablointi) jälkeen asetetaan ensin ohjausbitit DBX173.4, DBX173.5, DBX174.0, DBX174.4 ja DBX174.7, jonka jälkeen viimeisenä paikoitusajon aloittava ohjausbitti DBX173.6. On huomioitava, että kesken ajon, jos valmiusohjausbitit poistuvat, niin servo menee häiriötilaan.

Painokasettien mekaaninen keskiasento haluttiin vastaamaan pituusarvoa nolla millimetriä, jolloin mekaanisesti täyden säätöetäisyyden kasettien ääriarvot ovat plus-miinus kolme millimetriä säätöalueen ollessa kuusi millimetriä. Positiivinen arvo haluttiin linjan suunnassa oikealle ja negatiivinen arvo linjan suunnassa vasemmalle. Servon mekaniikka suhteessa servon pyörimissuuntaan oli päinvastainen eli asetettava paikoitusarvo täytyi ohjelmassa kääntää päinvastaiseksi. Servon todellisen paikan ilmoittavasta tilakaksoissanasta DBD222 on tehtävä myös muunnos takaisinpäin vastaamaan plus-miinus kolmen millimetrin säätöaluetta. Tämä arvo on nähtävissä kosketuspaneelista.

Paikoitusajon saavuttua haluttuun positioon, täytyy ohjausbitit asettaa pois päältä. Servon tilatiedoissa on tilabitti DBX212.2, mikä menee päälle saavutetussa paikassa. Ohjausohjelmassa kuitenkin päädyttiin vertailemaan servon ilmoittamaa todellista paikkaa suhteessa haluttuun ja käyttämään näiden eroa ohjaamaan pysäytys. Näin mahdollistetaan helposti halutun paikkojen erotustoleranssin säätö, vaikkakin oletusarvoisesti haluttu erotus on mahdollisimman lähellä nollaa. Todellisen paikan LU-arvosta vähennetään halutun paikan LU-arvo

ja tästä erotuksesta otetaan itseisarvo. Itseisarvon ollessa pienempi tai samansuuruinen kuin vertailuun asetettu toleranssi, asetetaan paikoituksen pysäytykseen tarkoitettu staattinen muistibitti päälle. Tämä mahdollistaa ohjelmallisesti sen, että servoa ei edes yritetä ohjata todellisen ja halutun paikan ollessa riittävän lähelle saman suuruisia.

Lohkoon ohjelmoitiin myös kiinteät raja-arvot pysäyttämään säätö mekaanisten rajojen lähestyessä. Jokainen yksikkö on hieman erilainen mekaanisilta ominaisuuksiltaan, jolloin päädyttiin asettamaan kyseiset raja-arvot mekaanisesti huonoimman painokasetin mukaiseksi ja raja-arvot ovat tällöin +/- 2.5mm. Mekaanisen raja-arvon täyttyessä säätö estetään.

Lisäksi lohkoon ohjelmoitiin Keycolor-raja-arvot, jolloin ennakoidaan mekaanisen rajan lähestymistä. Kun Keycolor-raja-arvo täyttyy, niin kyseisen yksikön toimilohkon lähtö menee päälle ja lähdössä olevan muistibitin avulla ilmoitetaan Keycolor-yksikölle, että Keycoloria on säädettävä vastakkaiseen suuntaan. Koska jokaisessa kuudessa yksikössä käytetään samaa toimilohkoa servojen säätöön, niin lohkoon tehtiin tulo, jonka avulla ilmoitetaan, onko kyseinen painoyksikkö Keycolor vai jokin muu jäljellä olevasta viidestä yksiköstä. Keycolor-säätöä varten asetettiin oma, hiljainen säätövauhti, jotta muut yksiköt ehtivät seuraamaan säätöajossa olevaa Keycoloria painolaadun heikentymättä. Muita yksiköitä ei kuitenkaan ohjelmoitu seuraamaan Keycolor-säätöä suoraan toimilohkon kautta, vaan Registar-järjestelmän ohjaamana, jolloin voidaan varmistua todellista painolaadusta säädön aikana.

Servon ohjaukseen tehdystä toimilohkosta saatiin toimiva kokonaisuus, joka soveltuu jokaisen painoyksikön sivurekisterin ohjaukseen, kunhan lohkoon tuodut lähtötiedot ovat yksilöllisesti oikein määritelty esimerkiksi Keycolor-yksikönvalinnan suhteen. Tästä koitui ongelmaksi myös se, että suurenmäärän toimintoja ohjelmoiminen yhteen toimilohkoon, monimutkaistaa toimintojen hallintaa jonkin verran. Toimintoja optimoitaessa pitäisi arvioida joidenkin toimintojen eriyttämistä omiin lohkoihinsa, jotta kokonaisuuden hallinta saattaisi olla helpompaa. Tällöin kuitenkin lisääntyy vaikeus lohkojen keskinäisen suoritusjärjestyksen suhteen eli millä toiminnolla on korkeampi prioriteetti.

5.4.3 FB70 - Keskiarvon laskennan toimilohko

Alkuperäisen suunnitelman mukaan oli tarkoitus käyttää induktiivista referenssianturia osoittamaan kasetin sivusuuntainen ääriasento, josta kasetti voidaan paikoittaa haluttuun nolla-asemaan. Ensimmäisten koeajojen jälkeen huomattiin, että tämä ei onnistu, sillä kasettia ei voida nopeasti ajaa asennosta toiseen painatuksen ollessa päällä. Ajonopeuden pitää olla melko pieni, ettei painopelti liu'u pahoin painotelan pinnassa. Hitaasti ajettaessa mahdollinen hävikin säästö tuotannon aloituksessa menetetään nollapisteajon tarvitsemasta suhteellisen pitkästä ajasta johtuen. Kasettia ei myöskään voida nollapisteajaa, ennen kuin kasetti on yläasennossaan eli valmiina painatukseen.

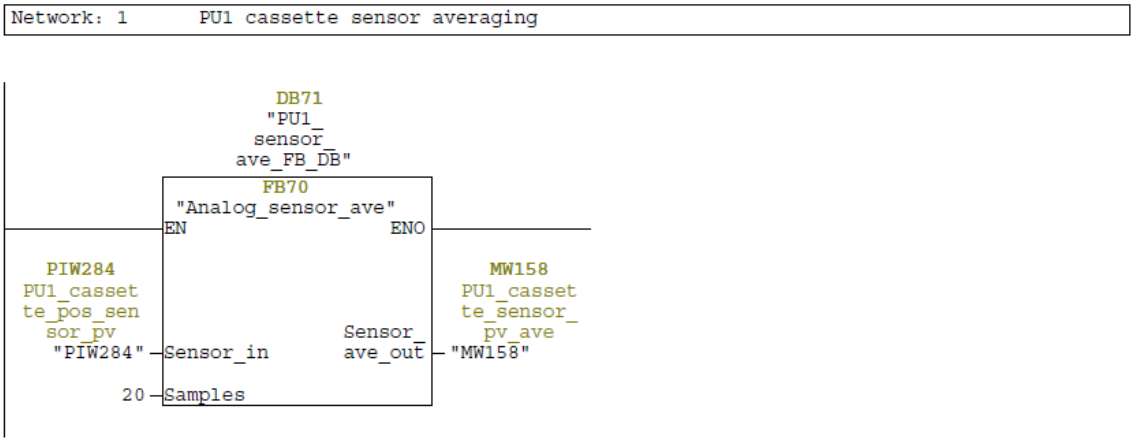
Ongelma ratkaistiin vaihtamalla induktiiviset referenssianturit induktiivisiin etäisyysantureihin, jotka lähettävät analogisen virtasignaalin 4 – 20 mA muodossa logiikan analogiatulokortille. Kyseisissä antureissa mittausalue on 0,8 – 8 mm eli 4 mA signaali vastaa 0,8 mm etäisyyttä ja vastaavasti 20 mA signaali vastaa 8 mm etäisyyttä. Logiikalla analogiatulokanavaa luettaessa kyseinen signaalialue vastaa lukuarvoa 0 – 27648, joka mahtuu yhden sanan suuruiseen muistiin.

Alkuperäinen automaatiosuunnitelma ei sisältänyt analogiatulokortteja, joten sellainen piti logiikkaan lisätä. Alkuperäisen suunnitelman logiikkakonfiguraatiossa oli jäljellä yhden moduulin tila logiikan pohjalevyssä. Tämä yhden moduulin tila olisi riittänyt, mutta varauduttiin mahdollisiin lisämoduulitarpeisiin ja logiikkaan asennettiin hajautusyksikkö Profinet-väylään. Fyysisesti kyseinen hajautusyksikkö asennettiin ohjauskeskukseen logiikan alapuolelle. Analogiasignaalin luentaan valittiin Siemensin High Feature -analogiatulokortti, jolloin signaalin tarkkuus, resoluutio, saatiin mahdollisimman suureksi. Resoluutiolla tarkoitetaan tässä mahdollisimman tasaisen lineaarista nousua tulosignaalia muunnettaessa lukuarvoksi.

Ensimmäisissä testiajoissa uudella anturityypillä huomattiin signaalin suuri heilahtelu johtuen painokasetin anturilla luettavan pinnan sisältämistä hammastuksista ja pultin syvennyksistä. Anturia ei voitu sijoittaa kasetissa tasaisempaa kohtaan, vaan signaali piti logiikassa muokata tasaisemmaksi. Tätä varten ohjelmoitiin toimilohko FB70 signaalin keskiarvon laskentaa varten.

Toimilohkoon tuodaan analogiasignaalin tulo ja haluttu näytemäärä. Kummatkin ovat yhden sanan mittaisia kokonaislukuja. Toimilohkon lähdöksi tulee näytemäärän mukainen keskiarvo yhden sanan mittaisena kokonaislukuna. Tämä keskiarvo käytetään toimilohko FB60:ssä ja muunnetaan esimerkiksi kasetin todelliseksi paikaksi.

Toimilohko toimii siten, että se kutsutaan organisaatiolohko OB34:ssa, joka on asetettu kahdensadan millisekunnin ohjelmankierrolle. Näytetaajuudeksi testien perusteella asetettiin kaksikymmentä näytettä. Nämä kertomalla saadaan keskiarvon päivitysväliksi neljä sekuntia. Toimilohko toimii yksinkertaisena keskiarvon laskentana eli näytteiden arvot lasketaan yhteen ja jaetaan näytteiden lukumäärällä. Toimilohkoon sisälle tuotu yhden sanan mittainen kokonaisluku täytyi aluksi muuntaa kahden sanan mittaiseksi kokonaisluvuksi, että yhteen lasketut lukuarvot mahtuvat toimilohkon staattiseen muistiin. Toimilohkossa saatu keskiarvo muunnetaan jälleen yhden sanan mittaiseksi kokonaisluvuksi ennen sen ulosvientiä. Kuvasta 16 nähdään kutsutun toimilohkon rajapinnat eli tulot ja lähdöt.



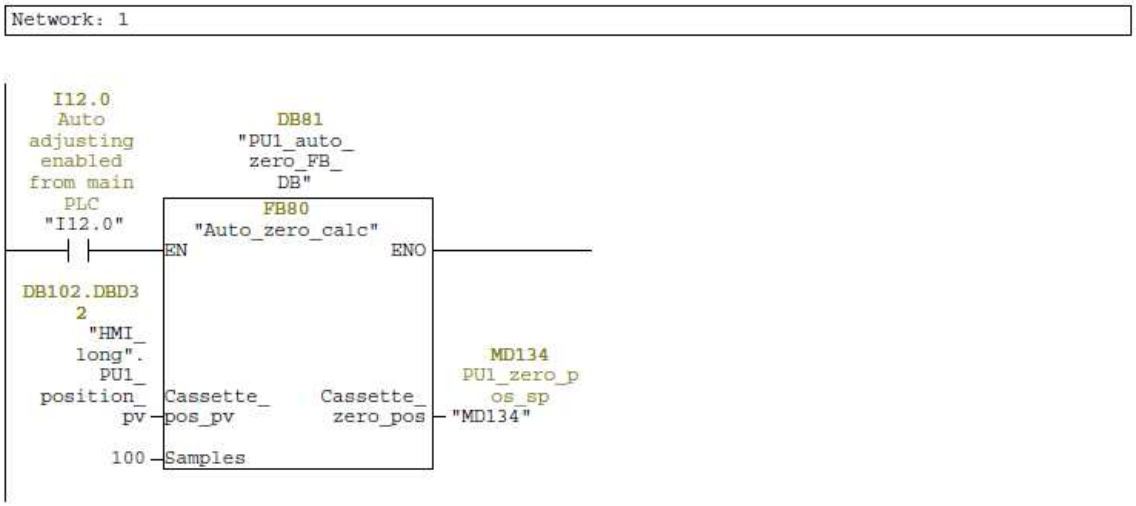
Kuva 16. Organisaatiolohkossa OB34 kutsuttu painoyksikkö yhden keskiarvolaskenta.

Tietyn mittaista ohjelmankiertoa käytettäessä tämän kaltaisessa laskennassa, vältytään käyttämästä erillisiä pulsseja näytetaajuuden hallintaan, sekä ohjelman rakenne pysyy myös melko selkeänä. Lisäksi saadaan optimoitua pääohjelmankiertoaika, eli organisaatiolohko OB1:n kutsumat toiminnot pysyvät pienempänä.

5.4.4 FB80 - Nollapisteen määrityksen toimilohko

Ideaalitilanteessa painopellit asetetaan kasetin mittamerkkiin täsmällisesti ja jokainen kasetti on mekaanisesti täysin identtinen. Tällöin jokaisessa painokasetissa on täysin sama nollapiste ja asetuksen jälkeen sivuttaissuuntaista rekisterin säätöä ei juurikaan edes tarvita. Todellisuudessa painolinjan kuudessa yksikössä on kuusi mekaanisesti hieman erilaista painokasettia. Tämän lisäksi tuotteita on kahta eri kokoa, jolloin painokasetteja tarvitaan kaksi painoyksikköä kohden. Säädön kannalta on tärkeää, että säädön lähtötilanne on mahdollisimman hyvin keskitetty linjan ja kasetin säätöalueen keskikohdan suhteen. Tähän tavoitteeseen päästään, kun painopellit asetetaan aina saman työohjeen mukaisesti ja täsmällisesti. Tämän lisäksi pitää määrittää jokaiselle kasetille ominainen nollakohta, mihin säätö ajetaan ennen tuotannon aloittamista. Näin sivurekisterin säädöntarve on mahdollisimman vähäistä ja tuotannon hävikki minimoidaan tuotannon aloituksessa. Jokaiselle kasetille ominainen nollapiste voidaan määrittää kahdella tavalla, joko operaattoreiden käsin asettamalla kosketuspaneelin avulla tai automaattisesti logiikkaohjelmalla.

Automaattiseen nollapisteen määrittelyyn ohjelmoitiin toimilohko FB80. Tämä lohko kutsutaan organisaatiolohkossa OB32, joka on tuhannen millisekunnin ohjelmankierrossa. Toimilohkoon tuodaan kasetin todellinen paikkatieto toimilohko FB60:stä kahden sanan mittaisena liukulukuna (Real). Liukuluku on desimaalimuotoinen eli paikkatietona voidaan käyttää millimetriä ja sen desimaaliosia. Millimetrin kymmenesosien tarkkuus on tässä sovelluksessa täysin riittävä. Toinen toimilohkoon tuotava tieto on näytetaajuus yhden sanan mittaisena kokonaislukuna. Kuvasta 17 nähdään kutsutun toimilohkon rajapinnat eli tulot ja lähdöt.



Kuva 17. Organisaatiolohkossa OB32 kutsuttu painoyksikkö yhden nollapistelaskenta.

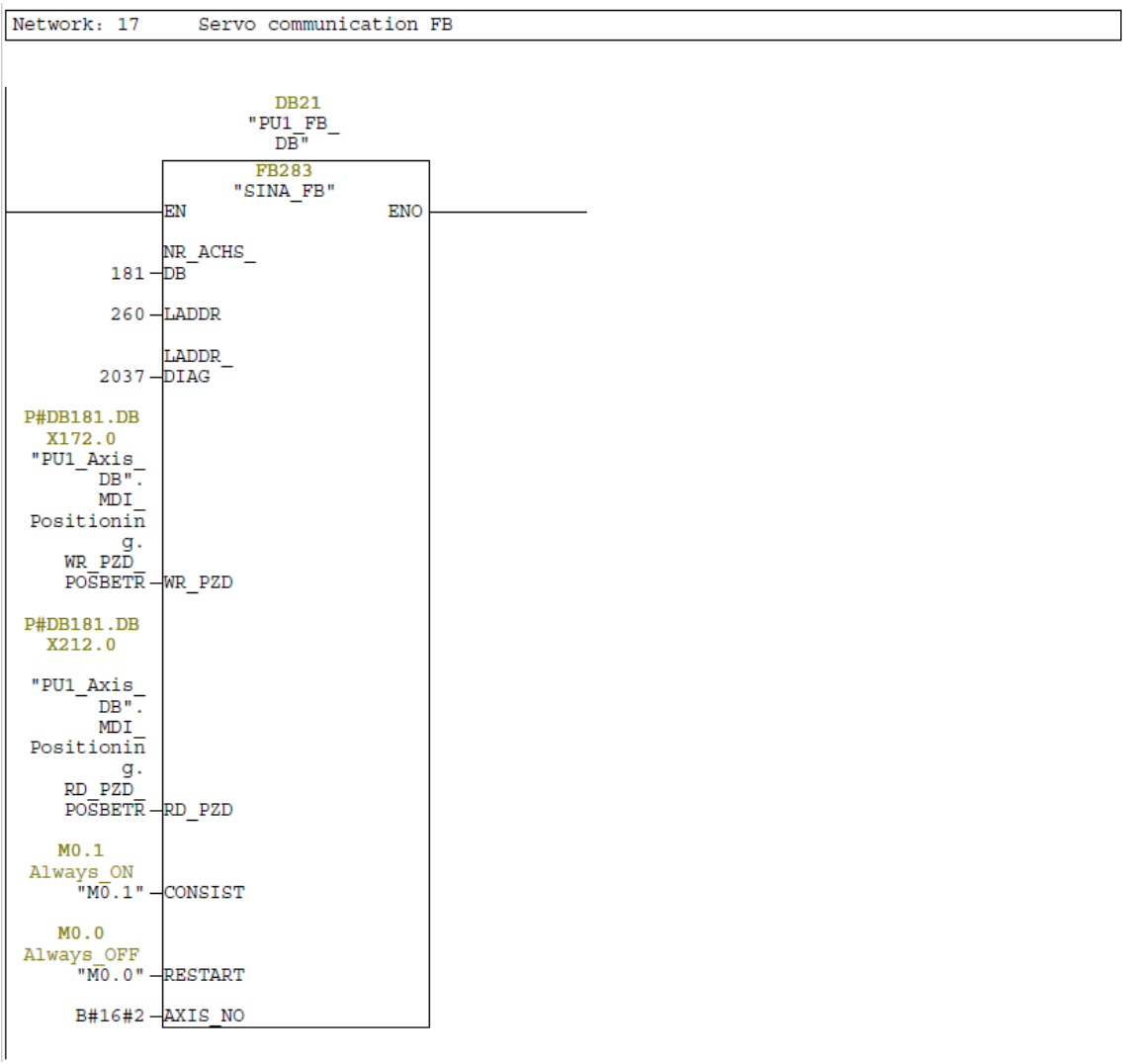
Toimilohkossa vertaillaan uusimman näytteen arvoa edelliseen näytteeseen. Jos näytteiden ero on alle 0,2 millimetriä, kyseinen arvo hyväksytään toimilohkon suorituksessa eteenpäin. Ohjelmallisesti tämä vertailu toteutettiin siten, että edellisestä hyväksytystä näytteestä vähennetään uuden näytteen arvo ja saatu erotus siirretään toimilohkon staattiseen muistiin. Tämän jälkeen kyseisestä erotuksesta otetaan itseisarvo, jolloin ei ole merkitystä oliko erotus positiivinen vai negatiivinen. Saatu itseisarvo vertaillaan LE-real (Less or Equal) vertailussa ja ehdon täyttyessä, uusin näyte siirretään staattiseen muistiin vertailtavaksi seuraavan näytteen kanssa. Toimilohkoon asetettiin näytetaajuudeksi sata kappaletta ja jokainen hyväksytty näyte kasvatti yhteenlaskulla ohjelmoitua laskuria yhdellä. Tämä on logiikassa helppo tehdä käyttämällä yhteenlaskuominaisuutta, mistä saatu tulos siirretään staattiseen muistiin ja seuraavassa kierrossa lisätään itsellään. Näin ollen, kun muistissa olevaan lukuarvoon lisätään luku yksi, niin jokaisessa kierrossa muistissa oleva luku kasvaa yhdellä. Tässäkin toiminnossa hyödynnetään logiikan ohjelmansuoritusjärjestystä. Myös logiikan omia laskureita voitaisiin käyttää, mutta monesti tämän kaltainen tehtävää varten ohjelmoitu yhteen- tai vähennyslaskuri on selkeämpi käyttöinen. Kun hyväksyttyjen näytteiden määrä saavuttaa asetetun näytemäärän, niin laskuri nollataan kirjoittamalla kokonaisluku nolla kyseiseen staattiseen muistiin ja siirtämällä viimeisin hyväksytty näyte staattiseen muistiin, josta se siirretään toimilohkon lähtöön ohjelman lopussa.

5.4.5 FB283 – Servokommunikaation toimilohko

Servo-ohjaimen ja logiikan välinen kommunikointi voidaan tehdä suoraan laitekonfiguraatiossa annettujen kommunikaatio-osoitteiden avulla. Monesti laitevalmistajat ovat kuitenkin ohjelmoineet valmiit toimilohkot kommunikaatiota varten, jolloin voidaan helposti hyödyntää halutessaan esimerkiksi kommunikaatiodiagnostiikkaa.

Siemens käyttää S7-300- ja S7-400 -logiikkasarjojen yhteydessä FB283:ksi nimettyä valmista toimilohkoa. Tähän toimilohkoon annetaan tuloiksi käytetyn datablokin numero, laitekonfiguraatiosta ensimmäinen tulosana, diagnostiikkasana, käytetyn datablokin ohjaussanan ensimmäinen bitti sekä ohjaussanan pituus ja käytetyn datablokin tilasanan ensimmäinen bitti sekä tilasanan pituus. Myös ohjatun laitteen numero eli moniakselikäytössä ohjattavan akselin numero annetaan. Tämä tieto saadaan parhaiten servoille tarkoitetusta konfiguraatio-ohjelmasta. On huomioitava, että FB283 käytettäessä on ohjaus- ja tilasanojen oltava numeroltaan samat, sillä kyseisessä toimilohkossa niitä ei anneta erillisillä suureilla. Siemensin ohjelmisto oletusarvoisesti määrittelee ne konfiguraatiossa aina samaksi.

Toimilohko FB283 kutsutaan pääohjelmankierrossa eli organisaatiolohko OB1:n ohjaamana. Näin ollen servokommunikaatio päivittyy niin usein kuin logiikan keskusyksikkö kykenee sen suorittamaan. Kutsujärjestysvuorossa oleva toimilohko lukee sille osoitetusta datalohkosta ohjausbitit ja -sanat, siirtää ne servo-ohjaimelle, lukee servo-ohjaimelta tilabitit ja tilasanat sekä kirjoittaa ne osoitettuun datalohkoon. Tilatiedot voidaan hyödyntää halutulla tavalla logiikkaohjelmassa. Kuvasta 18 voidaan nähdä painoyksikkö yhden servokommunikaatioon käytetyn toimilohko FB283:n tulorajapinta eli mitä tuloja sille on tuotu.



Kuva 18. Painoyksikkö yhden servokommunikaation toimilohko FB283.

5.4.6 FC – Käytetyt toiminnot

Jokaiselle kuudelle rekisterinohjaukselle ohjelmoitiin toiminnot eli funktiot (FC), jotka kutsutaan pääohjelmankierrossa. Nämä toiminnot sisältävät perusohjauksia ja kutsut servokommunikaation toimilohkoille.

Perustoimintoja ovat esimerkiksi eri ohjauspaikoista tehtävien käskyjen rinnakkain asettaminen yhdeksi ohjaukseksi eli esimerkiksi painoyksikön painonapista käsiohjausta painettaessa, tapahtuu sama ohjaus kuin vastaavasta kosketuspaneelissa olevasta painikkeesta. Muita perusohjauksia ovat esimerkiksi painonappien valoindikaattoreiden ohjaukset.

5.4.7 DB – Käytetyt Datalohkot

Kosketuspaneelin ja logiikan väliseen ohjaukseen ja tiedonsiirtoon tehtiin neljä eri datalohkoa. Yksi datalohko sisälsi kaikki yhden bitin mittaiset ohjaukset, yksi datalohko sisälsi sanan mittaiset lukuarvot, yksi datalohko sisälsi kahden sanan mittaiset lukuarvot ja yksi sisälsi hälytyslistaukset. Nämä kaikki jaettiin eri datalohkoihin, jolloin uusien toimintojen ja arvojen lisääminen myöhemmin pitää ohjaukseen tarkoitetut datalohkot selkeinä ja järjestelmällisinä. Voitaisiin käyttää myös yhtä datalohkoa kaikkeen kosketuspaneelin ja logiikan väliseen tiedonsiirtoon, mutta tällöin erimittaiset ohjaukset menisivät sekalaiseen järjestykseen ja rakenne tulisi vaikeampi selkoiseksi. Toteutettu lohkojaottelu myös auttaa uusien ohjausten lisäämisen myöhemmin ja lohkon logiikkaan lataukseen, sillä samalla ei vaikuteta kaikkiin arvoihin ja mahdollisesti palauteta niitä oletusarvoiksi aina ladattaessa. Datalohkoja hyödynnettiin myös pysyväismuistina esimerkiksi säilyttämään rekistereiden nollapistearvoja.

Joskus ohjelmoijat käyttävät kosketuspaneeleiden ja logiikan välisiin ohjauksiin suoraan logiikan omia M-muistialueita, mutta tämä ei ole kovin käytännöllinen tapa, sillä näin varataan mahdollisesti rajallista muistia ja tehdään ohjelmasta vaikeampi selkoinen, kun ei voida olla ensinäkemältä varmoja, onko kyseinen muistialue logiikkaohjelman kirjoittama vai kosketuspaneelista ohjattu. Datalohko voidaan nimetä esimerkiksi “HMI”-alkuiseksi, jolloin ohjelmasta nähdään heti, että kyseessä on esimerkiksi kosketuspaneelin ja logiikan välinen tiedonsiirto tai ohjaus.

Ohjelmaa varten tehdyt toimilohkot tarvitsevat omat datalohkonsa, yksi lohko jokaista käytettyä toimilohkoa varten. Nämä datalohkot syntyvät automaattisesti, kun toimilohko otetaan käyttöön ja numeroidaan yksilöllisesti.

5.4.8 Ohjelmankiertoaika

Koska ohjelmassa käytettiin ajoitettuja kutsuja eri toiminnoille, kuten keskiarvonlaskennalle, niin silloin ohjelmankiertoaika ei voi pysyä täysin samana kaikilla kierroilla. Ohjelmankiertoaika kasvaa hieman kutsuttaessa pääohjelmankierron ulkopuolisia toimintoja ja silloin ohjelmankiertoajan kasvun suuruus riippuu kut-

sutuiden toimintojen pituudesta. Muuttuva ohjelmankiertoaika voi haitata esimerkiksi hyvin tarkkojen säätöpiirien toimintaa, jolloin on otettava huomioon kyseinen kiertoajan muutos. Sivurekisterin säätöohjelmassa hieman muuttuvalla kiertoajalla ei ole merkitystä.

Ohjelmankiertoaika ohjelmalla jäi kahteen millisekuntiin ja tuotantoajossa voidaan todeta sen olevan täysin riittävä ohjaukselle, sillä säädössä ei voida havaita mitään muutoksia tai puutteita verrattuna laitteistolta vaadittavaan toiminnallisuuteen.

5.5 Käyttöönotto

Ensimmäisenä laitteistossa asennettiin ohjauskeskus ja siihen tuleva sähkönsyöttö. Tämän jälkeen aloitettiin painoyksikkö viiteen tuleva kenttäkaapelointi ja asennettiin paineilmojen syötöt. Kyseinen painoyksikkö valittiin pilottikohteeksi, jotta voitiin testata aluksi yhden rekisterin säätömekaniikan haluttu toiminta ja vasta sen jälkeen valmistuttaa mekaniikat loppuihin viiteen painoyksikköön. Tällainen projektitoiminta on erittäin suotavaa riskien hallinnan kannalta, jos täysin vastaavanlaisia kohteita on useita. Näin minimoidaan riskit niin taloudellisesti kuin tuotannollisesti. Suunniteltu mekaniikka toimi halutusti ja loput viisi voitiin valmistuttaa ilman muutoksia.

Mekaanisten ja sähköisten asennusten valmistuttua pilottiyksikköön, kytkettiin järjestelmään sähköt ja tehtiin I/O-testi. Kyseisellä testillä varmistutaan siitä, että esimerkiksi antureilta ja painonapeilta tulevat signaalit oikeisiin logiikan osoitteisiin, eikä kaapeleiden kytkennöissä ole sattunut asennusvirheitä. Tässä testauksen vaiheessa on myös hyvä säätää antureiden suuntaus, etäisyys ja herkkyys kohdalleen. Lisäksi voidaan testata toimilaitteet eli ajaa liikkeet läpi ja todeta ohjausten toiminta.

Kun pilottiyksikön todettiin olevan valmis testiajoon asennusten puolesta, asetettiin logiikalle, käyttäjäpäätteelle ja servo-ohjaimille IP-osoitteet. Osoitteiden tulee olla yrityksen linjauksen mukaiset, jolloin vain IP-osoitteen viimeiset numerot ovat vapaasti valittavissa juoksevan järjestysnumeroinnin mukaisesti, muu osa osoitteesta määräytyy esimerkiksi tuotantolinjan mukaisesti. Siemensin käyttäjäpäätteisiin voidaan helpoiten asettaa manuaalisesti IP-osoite valitsemalla paneelin

käynnistyksen yhteydessä asetusvalikosta yhteyksien konfigurointi. Siellä voidaan paneelille asettaa IP-osoite, aliverkon peite ja tarvittaessa oletusyhdyskäytävän osoite. Logiikalle ja servo-ohjaimille osoitteen antaminen tehtiin Siemensin ohjelmointiohjelman Step 7 V5.5 avulla. Kyseisestä ohjelmointiohjelmasta löytyy Ethernet-osoitteen muuttamiseen erillinen osio, jonka avulla voidaan hakea kaikki lähiverkkoon liitetyt laitteet. Ohjelma listaa löydettyjen laitteiden verkkonimen, IP-osoitteen ja MAC-osoitteen. Jos listatuista laitteista ei olla varma, niin ne voidaan paikallistaa valitsemalla listalta löydetty laite ja asettamalla testitila päälle. Tässä testitilassa laite vilkuttaa etupaneelin ledejään, jolloin voidaan olla varma siitä, mikä laite on valittuna. Tämän jälkeen valitulle laitteelle asetetaan IP-osoite, aliverkon peite ja tarvittaessa oletusyhdyskäytävä. Laitteita etsittäessä näin on huomioitava, että ohjelmointiasemaan on asetettu automaattinen IP-osoite eli DHCP-asetus päälle. Tätä ei tarvitse tehdä, jos tiedetään, että etsittävä laite kuuluu jo IP-osoitteeltaan samaan osoiteavaruuteen kuin ohjelmointiasema. Yleensä Siemens-logiikoiden keskusyksiköiden IP:t ovat oletusasetukseltaan 0.0.0.0. Kun laitteille IP-osoitteet ovat asetetut, niin on muistettava asettaa ohjelmointiaseman IP-osoite jälleen samaan osoiteavaruuteen kuin mikä laitteille asetettiin.

Osoitteiden asetuksen jälkeen voitiin logiikkaan ladata valmisteltu logiikkaohjelma, käyttäjäpaneelin ohjelma, parametroida servo-ohjain ja tehdä logiikan ja servo-ohjaimen välisen kommunikoinnin mahdollistava konfiguraatio. Tässä konfiguraatiossa määritellään halutut sanomien pituudet lähetykseen ja vastaanottoon sekä sanomien osoitteiden numerointi. Sekä logiikassa että servo-ohjaimessa täytyy olla yhtenevät sanomien pituudet ja osoitteet.

Servon liikkuminen eli kommunikoinnin toiminta testattiin siten, ettei servo ollut vielä mekaanisesti asennettu laitteistoon. Näin vältyttiin mahdollisilta laitteiston vaurioilta ja vaaratilanteilta, jos ohjauksessa on sattunut virhe ja servo pyörii esimerkiksi aivan liian suurella nopeudella. Kun servon ohjauksen huomattiin toimivan toivotusti, se asennettiin mekaanisesti paikalleen. Tässä vaiheessa pilottiyksikkö oli manuaalista testiajoa varten valmis, mikä tarkoitti liikkeiden ajamista painonapeista. Kun toiminnan todettiin olevan haluttu, niin tuotanto sai ohjeistuksen jälkeen luvan aloittaa normaali tuotantoajo ja säätää pilottiyksikön sivurekisteriä painonapeista aikaisemmin manuaalisen käsipyöräsäädön sijaan.

Loput viisi painoyksikköä asennettiin samalla tavoin kuin pilottiyksikkö, jolloin tuotantohenkilökunta pystyi käyttämään sivurekistereiden säätöön painonappeja. Pilottiyksikön mekaanisesta asennuksesta saaduin kokemuksin pystyttiin arvioimaan tarkemmin asennukseen menevä aika, jolloin hankittiin ulkopuolisia asennusresursseja ja asennettiin kaksi yksikköä kerrallaan. Myös asennusten esivalmisteltavat kohteet tehtiin ennen käyttöönottoasennusta. Näin onnistuttiin puolittamaan asennukseen mennyt aika verrattuna pilottiyksikön asennukseen.

BST Eltromatin käyttöönottoinsinööri saapui tehtaalle automaattitoiminnon käyttöönottoon noin kuukausi sen jälkeen, kun kaikki yksiköt olivat manuaalisesti ajettavissa painonapeista. Alkuun tehtiin I/O-testaaminen järjestelmien välillä, jolloin varmistettiin kaikkien signaalien saapumisesta oikeisiin osoitteisiin. Tämän jälkeen aloitettiin normaali tuotantoajo testimateriaalilla. BST Eltromatin käyttöönottoinsinööri sääti Registar-järjestelmästä säätönopeudet mahdollisimman lähelle haluttua seuraamalla tuotantoajossa laitteiston reagointinopeutta säätötarpeeseen. Järjestelmään tehtiin hienosäätöä samalla, kun tuotantoajo oli käynnissä. Kun asetuksiin ja sivurekisterinsäädön toimintaan oltiin riittävän tyytyväisiä linjan tuotantohenkilökunnan puolesta, niin jäätiin seuraamaan laitteiston toimintaa pidemmän aikavälin tuotannossa. Tässä vaiheessa tuotantohenkilökunnalle oli jo ohjeistettu laitteiston peruskäyttämiseen liittyvät toiminnot ja voitiin seurata myös heidän toimintaa laitteistoa käytettäessä.

6 Yhteenveto ja pohdinta

6.1 Projektin tulokset

Projektin tuloksia voidaan arvioida monesta eri näkökulmasta. Linjan käynnistyksen ja hyvän painolaadun saavuttamisen välissä joudutaan poistamaan kaikki tuotettu materiaali, joten tämä hävikki haluttiin minimoida siltä osin, kuin se on automaattisen sivurekisterin säädön osalta mahdollista. Kyseisen hävikin säästön arvioiminen on melko hankalaa, sillä samaan aikaan säädetään myös prosessin muita kohtia, kuten pituussuuntaista rekisteriä. Käytössä voidaan kuitenkin havaita, että sivurekisterin kohdistus saavutetaan ennen pituussuuntaista kohdistusta, jolloin tulosten voidaan todeta tältä osin olevan hyviä.

Sivurekisterin säädöllä on merkittävä tehtävä, kun kartonkiradan jatkoskohta kulkee painoprosessin läpi. Tällöin radassa syntyy heilahtelua, joka näkyy painojäljessä sivurekisteriheittona. Tältä osin tulokset viittaavat noin viidenkymmenen prosentin vähennykseen, mutta painolinjan ratavakausongelmien vuoksi lukema on suuntaa antava arvio. Automatisoitu sivurekisterin säätö vapautti prosessinhoitajien aikaa muuhun prosessin valvontaan, kuten painatuksen sävyjen hallintaan liittyvään työhön. Säädön hallinta on kokoaikaista, kun manuaalisesti tehtävänä sitä jouduttaisiin valvomaan jatkuvasti laadunvalvonnassa. (Kankkunen 2019)

Tuloksia voidaan verrata myös toteutuksen kustannuksiin ja ajankäyttöön, jolloin voidaan arvioida investoinnille takaisinmaksuaika, kun lasketaan saavutettavat hyödyt. Itsetehtynä projekti tuli maksamaan noin viidesosan painolinjan alkuperäisen toimittajan samansisältöiseen kilpailevaan tarjoukseen verrattuna. Käytännössä olisi ollut mahdotonta toteuttaa kilpailevan tarjouksen sisällön mukaisesti projektia, sillä se olisi vaatinut painokasettien lähettämistä Ranskaan pitkäksi aikaa. Tämä olisi ollut täysin mahdotonta.

Projektin tuloksista voidaan päätellä projektin onnistuneen tavoitteiden mukaisesti.

6.2 Pohdinta

Kyseinen projekti ja opinnäytetyö olivat hyvin mielenkiintoisia, sillä sille oli selkeä todellinen tarve olemassa. Tuotantoprosessissa oli ongelma ja se täytyi ratkaista. Tetra Pak käyttää World Class Manufacturing (WCM) ajatusmallia ja metodiikkaa kaikessa toiminnassaan ja sen keskeiset periaatteet ovat hävikin vähentäminen ja jatkuva kehitys. Tämä projekti opinnäytetöineen täyttää kummatkin edellä mainitut periaatteet.

Projekti oli todellisuudessa hyvin monipuolinen sisältäen mekaniikkaa, sähkö-automaatiota ja ohjelmointia. Vaikkakin opinnäytetyö käsitteli pääasiassa projektin automaation osuutta, erityisesti logiikkaohjelmointia, niin se opetti myös laaja-alaisesti projektinhallintaa, sisältäen esimerkiksi budjetointia ja aikataulutusta.

Hävikkiä voidaan laitteistolla vähentää vielä lisää tulevaisuudessa, kun kartonkirata saadaan kulkemaan vakaammin myös jatkoskohdassa. Tähän ei vielä opinnäytetyön valmistumisen aikaan ole löytynyt ratkaisua.

6.3 Jatkokehitysmahdollisuudet

Laitteiston automatiikan osuutta voidaan lisätä jatkossa ja esimerkiksi optimoida liikenopeuksia. Säädön nopeuden nostolla ei kuitenkaan enää ole suurta merkitystä, sillä tuotannonaloituksessa sivurekisterin hävikin määrä on jo pienentynyt niin paljon, että se sulautuu muuhun tuotannonaloitushävikkiin. Tällä tarkoitetaan sitä, että haluttu sivurekisterin suuntainen säätö saavutetaan ennen kuin muut laadulliset säädöt ovat kohdallaan. Automatiikan nostolla voitaisiin kuitenkin saavuttaa laitteiston käyttämisen nopeutusta, jolloin linjaoperaattoreiden aika voi siltäkin osin kohdistua muuhun prosessinhallintaan. Yhtenä esimerkkinä voisi olla käyttöliittymien kehittäminen.

Lähteet

Graafinen 2015. painomenetelmät.
https://www.graafinen.com/tietopankki/painomenetelmat/ Luettu 31.12.2018.

HMK Wiki 2013. Telegram 111.
http://wiki.hmkdirect.com/mediawiki/index.php/Telegram_111 Luettu 2.1.2019.

Kankkunen A. 2019. Painoprosessivastaava. Tetra Pak Production Oy. Imatra. Sähköpostikysely 12.3.2019

Profinet North America 2019.
https://us.profinet.com/technology/profinet/ Luettu 15.3.2019.

Siemens 2019. Industry mall.
https://mall.industry.siemens.com/mall/en/WW/Catalog/Product/6ES7321-1BH02-0AA0 Luettu 2.1.2019.

Tetra Pak.
https://www.tetrapak.com/about/facts-figures Luettu 28.4.2019.

OB1 - <offline>

Liite 1
1(1)

"Main_Cycle"

Name: **Family:**
Author: **Version:** 0.1
Block version: 2
Time stamp Code: 03/05/2019 02:08:24 PM
Interface: 02/15/1996 04:51:12 PM
Lengths (block/logic/data): 00222 00108 00022

Name	Data Type	Address	Comment
TEMP		0.0	
OB1_EV_CLASS	Byte	0.0	Bits 0-3 = 1 (Coming event), Bits 4-7 = 1 (Event class 1)
OB1_SCAN_1	Byte	1.0	1 (Cold restart scan 1 of OB 1), 3 (Scan 2-n of OB 1)
OB1_PRIORITY	Byte	2.0	Priority of OB Execution
OB1_OB_NUMBR	Byte	3.0	1 (Organization block 1, OB1)
OB1_RESERVED_1	Byte	4.0	Reserved for system
OB1_RESERVED_2	Byte	5.0	Reserved for system
OB1_PREV_CYCLE	Int	6.0	Cycle time of previous OB1 scan (milliseconds)
OB1_MIN_CYCLE	Int	8.0	Minimum cycle time of OB1 (milliseconds)
OB1_MAX_CYCLE	Int	10.0	Maximum cycle time of OB1 (milliseconds)
OB1_DATE_TIME	Date_And_Time	12.0	Date and time OB1 started

Block: OB1 "Main Program Sweep (Cycle)"

Tetra Pak / T. Pinomäki 5.3.2019

Network: 1 Always ON / OFF

```
AN    "M0.0"  M0.0              -- Always_OFF
O     "M0.1"  M0.1              -- Always_ON
S     "M0.1"  M0.1              -- Always_ON
R     "M0.0"  M0.0              -- Always_OFF
```

Network: 2 Program main cycle calls

```
CALL  "PU_common"       FC90
CALL  "PU1_Control_FC"  FC91
CALL  "PU2_Control_FC"  FC92
CALL  "PU3_Control_FC"  FC93
CALL  "PU4_Control_FC"  FC94
CALL  "PU5_Control_FC"  FC95
CALL  "PU6_Control_FC"  FC96
```

Liite 2
1(4)

OB32 - <offline>

"1000ms_Cycle"

Name: **Family:**
Author: **Version:** 0.1
Block version: 2
Time stamp Code: 03/05/2019 02:10:15 PM
Interface: 02/15/1996 04:51:05 PM
Lengths (block/logic/data): 00556 00422 00026

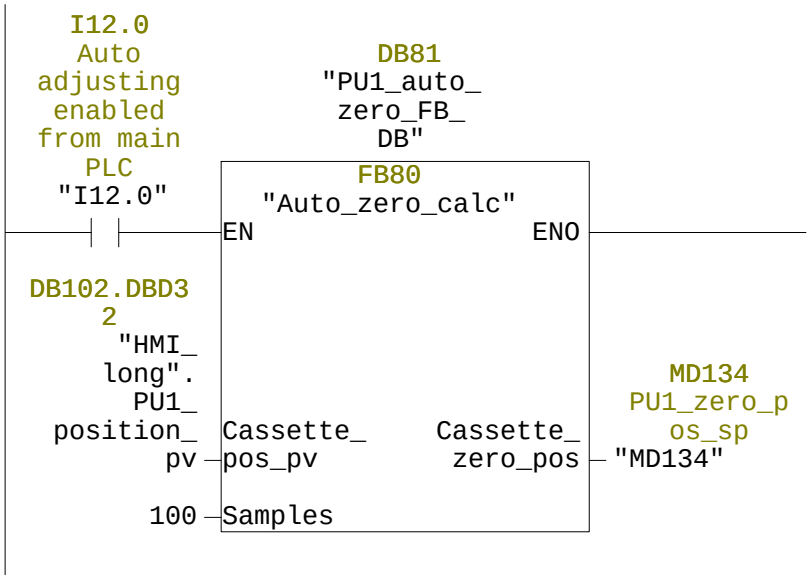
Name	Data Type	Address	Comment
TEMP		0.0	
OB32_EV_CLASS	Byte	0.0	Bits 0-3 = 1 (Coming event), Bits 4-7 = 1 (Event class 1)
OB32_STRT_INF	Byte	1.0	16#33 (OB 32 has started)
OB32_PRIORITY	Byte	2.0	Priority of OB Execution
OB32_OB_NUMBR	Byte	3.0	32 (Organization block 32, OB32)
OB32_RESERVED_1	Byte	4.0	Reserved for system
OB32_RESERVED_2	Byte	5.0	Reserved for system
OB32_PHS_OFFSET	Int	6.0	Phase offset (integer, milliseconds)
OB32_RESERVED_3	Int	8.0	Reserved for system
OB32_EXC_FREQ	Int	10.0	Frequency of execution (msec)
OB32_DATE_TIME	Date_And_Time	12.0	Date and time OB32 started

Block: OB32 "Cyclic Interrupt"

Tetra Pak / T. Pinomäki 5.3.2019

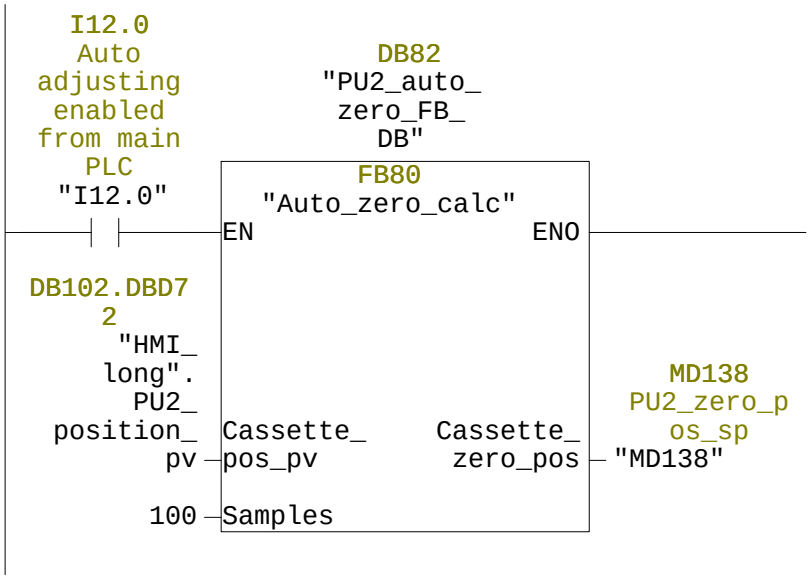
1000ms cycle

Network: 1

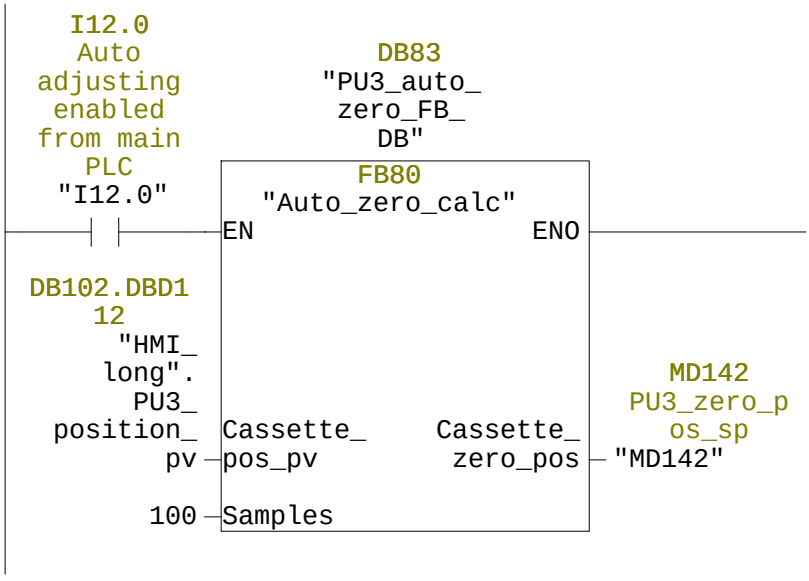


Liite 2
2(4)

Network: 2

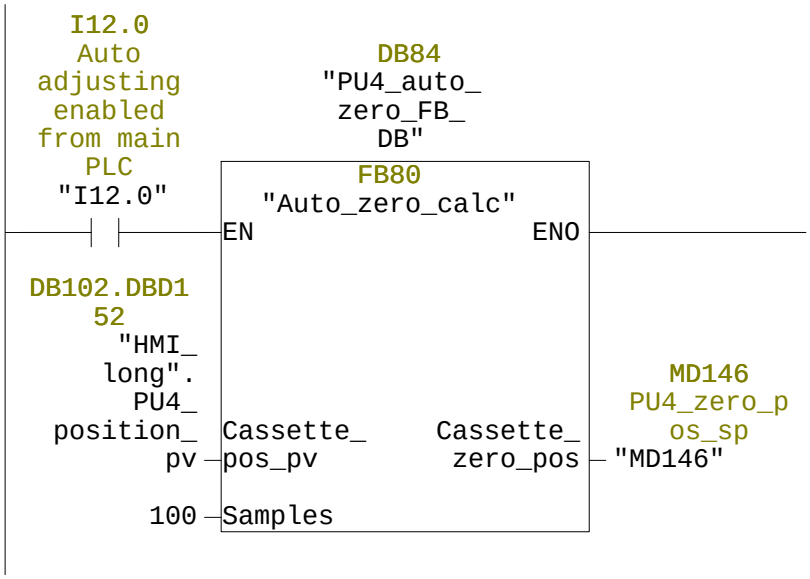


Network: 3

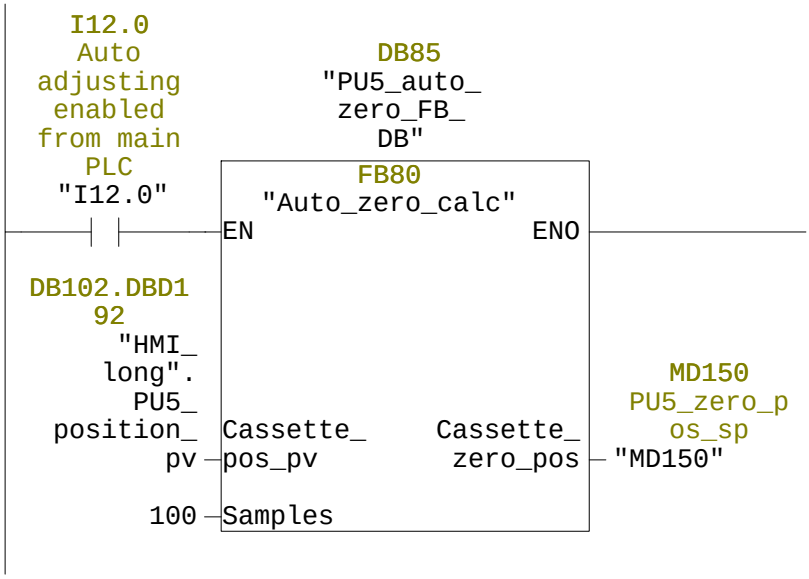


Liite 2
3(4)

Network: 4

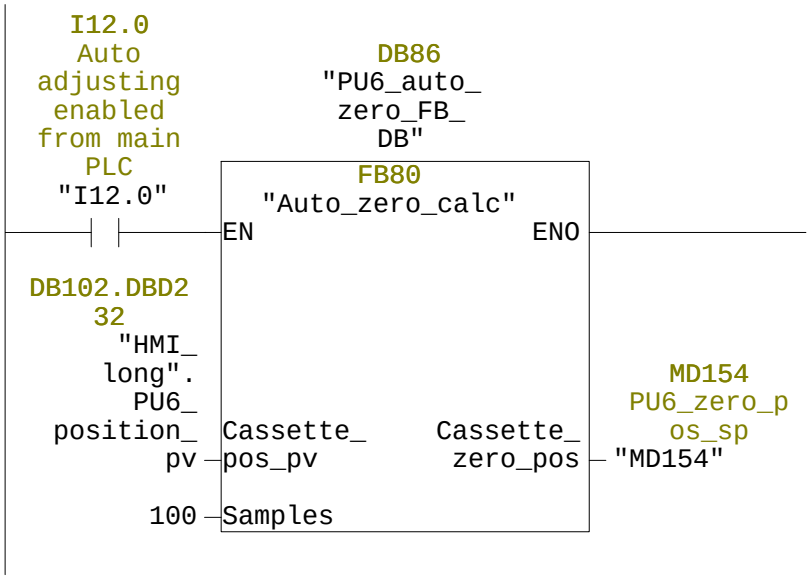


Network: 5



Network: 6

Liite 2
4(4)



Liite 3
1(3)

OB34 - <offline>

"200ms_Cycle"

Name: **Family:**
Author: **Version:** 0.1
Block version: 2
Time stamp Code: 03/05/2019 02:10:44 PM
Interface: 02/15/1996 04:51:06 PM
Lengths (block/logic/data): 00460 00338 00026

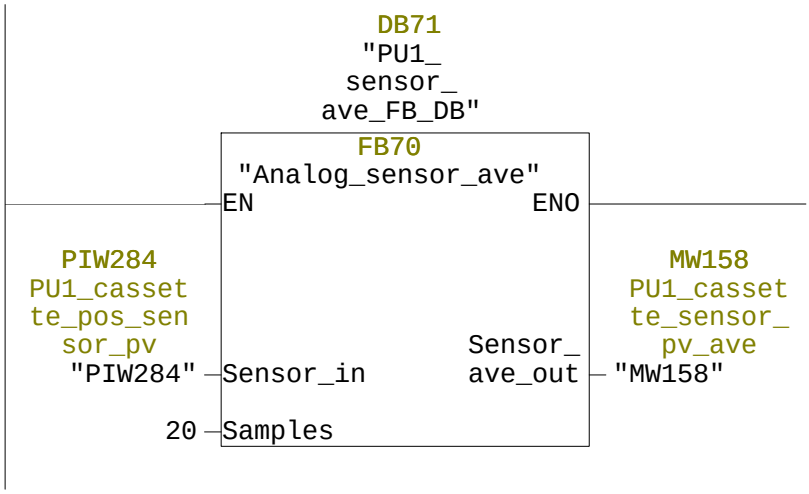
Name	Data Type	Address	Comment
TEMP		0.0	
OB34_EV_CLASS	Byte	0.0	Bits 0-3 = 1 (Coming event), Bits 4-7 = 1 (Event class 1)
OB34_STRT_INF	Byte	1.0	16#35 (OB 34 has started)
OB34_PRIORITY	Byte	2.0	Priority of OB Execution
OB34_OB_NUMBR	Byte	3.0	34 (Organization block 34, OB34)
OB34_RESERVED_1	Byte	4.0	Reserved for system
OB34_RESERVED_2	Byte	5.0	Reserved for system
OB34_PHS_OFFSET	Int	6.0	Phase offset (integer, milliseconds)
OB34_RESERVED_3	Int	8.0	Reserved for system
OB34_EXC_FREQ	Int	10.0	Frequency of execution (msec)
OB34_DATE_TIME	Date_And_Time	12.0	Date and time OB34 started

Block: OB34 "Cyclic Interrupt"

Tetra Pak / T. Pinomäki 5.3.2019

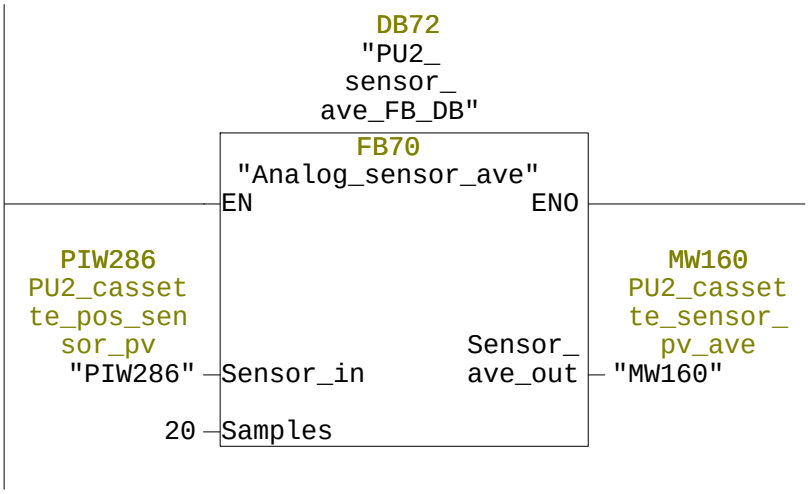
200ms cycle

Network: 1 PU1 cassette sensor averaging

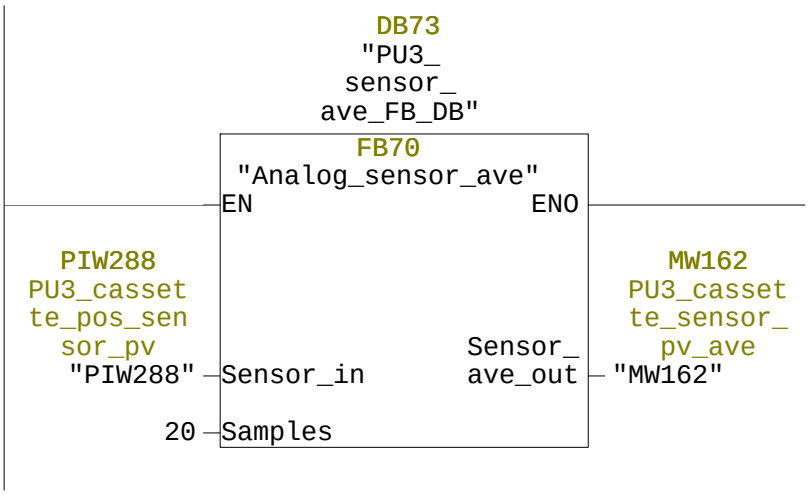


Liite 3
2(3)

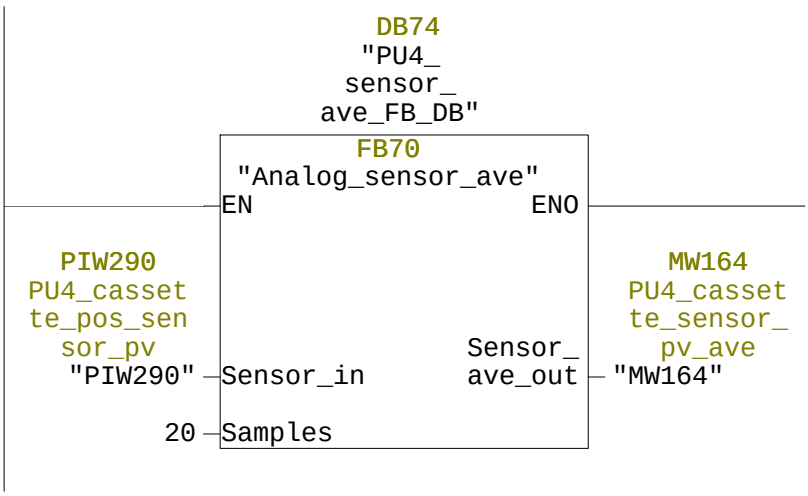
Network: 2 PU2 cassette sensor averaging



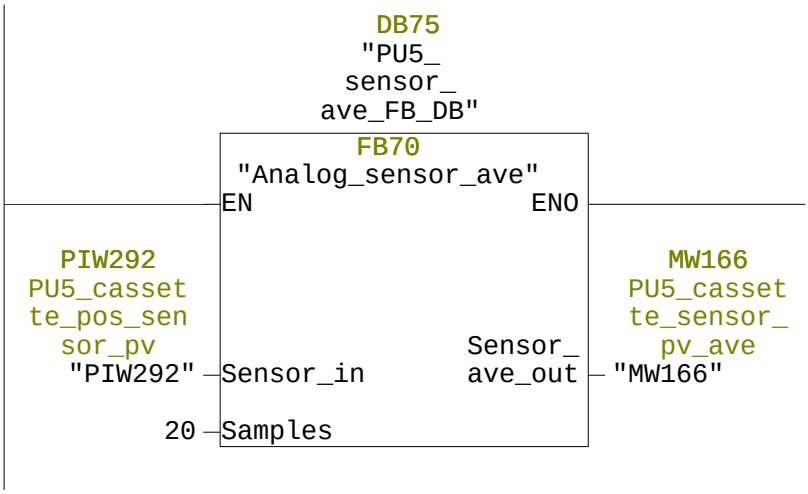
Network: 3 PU3 cassette sensor averaging



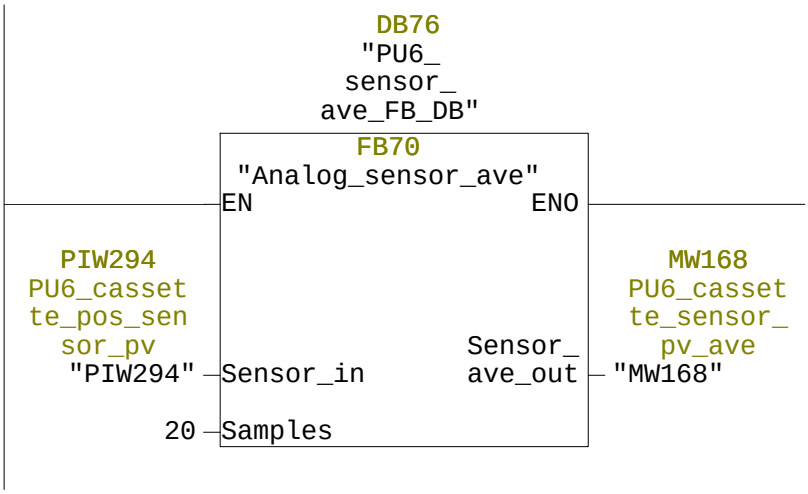
Network: 4 PU4 cassette sensor averaging



Network: 5 PU5 cassette sensor averaging



Network: 6 PU6 cassette sensor averaging



Liite 4
1(15)

FC91 - <offline>

"PU1_Control_FC"

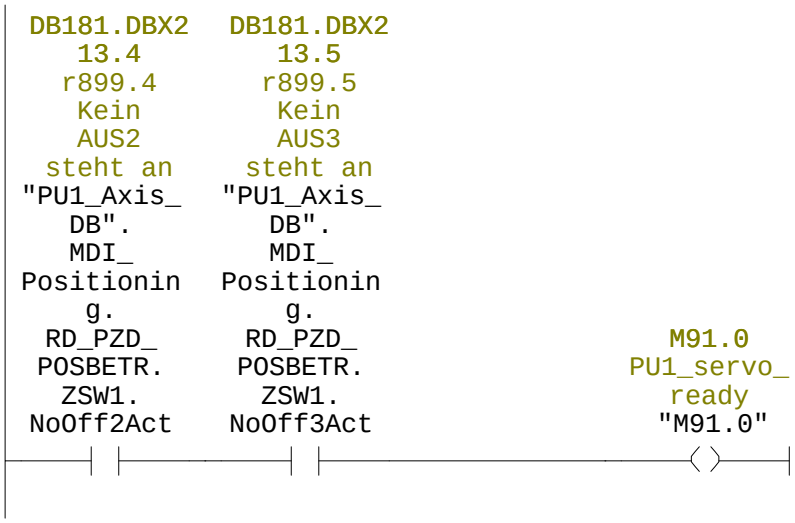
Name: **Family:**
Author: **Version:** 0.1
Block version: 2
Time stamp Code: 03/05/2019 02:15:06 PM
Interface: 07/21/2015 05:49:01 PM
Lengths (block/logic/data): 01424 01264 00020

Name	Data Type	Address	Comment
IN		0.0	
OUT		0.0	
IN_OUT		0.0	
TEMP		0.0	
Speed_override	DInt	0.0	
Servo_fault	Bool	4.0	
servo_pos_Pv_out	Real	6.0	
TEMP_FAULT_ACK	Bool	10.0	
Move_GR_man	Bool	10.1	
Move_OP_man	Bool	10.2	
Servo_up_down	Bool	10.3	
RETURN		0.0	
RET_VAL		0.0	

Block: FC91 PU1 -control

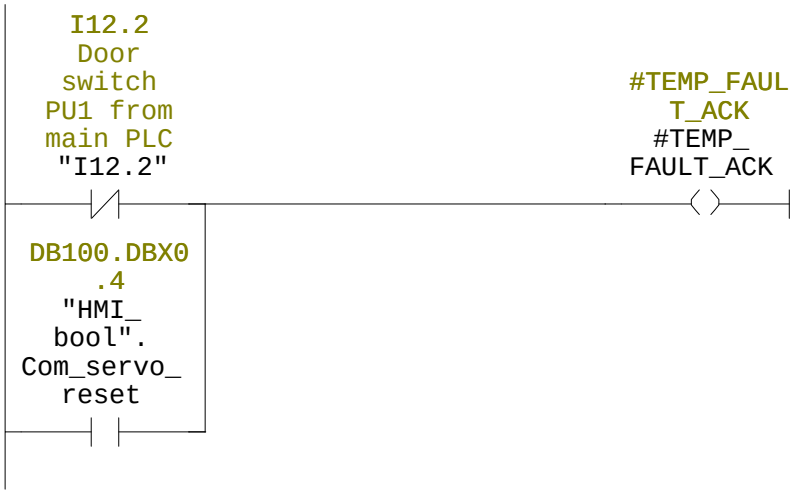
Tetra Pak / T. Pinomäki 5.3.2019

Network: 1 Servo ready

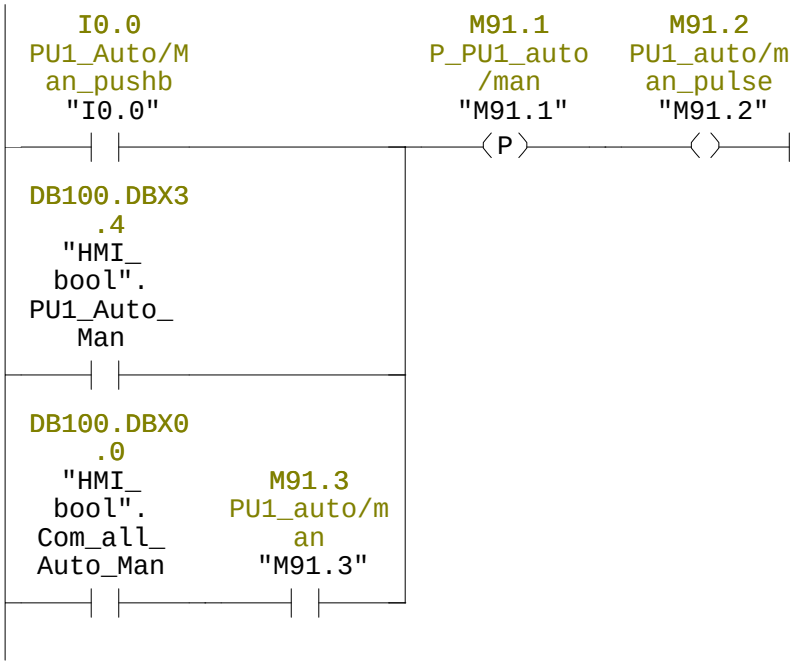


Network: 2 Fault ack

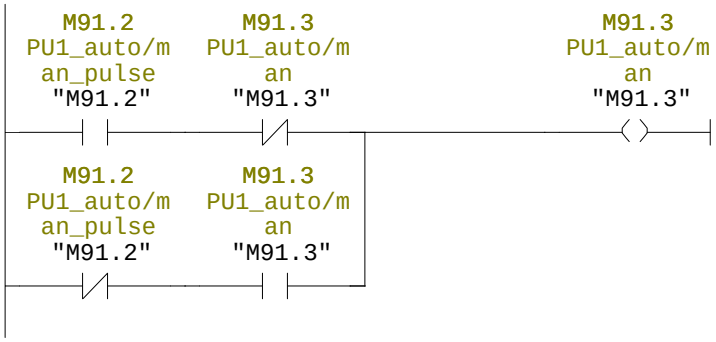
Liite 4
2(15)



Network: 3 HMI / PB - Auto/man pulse

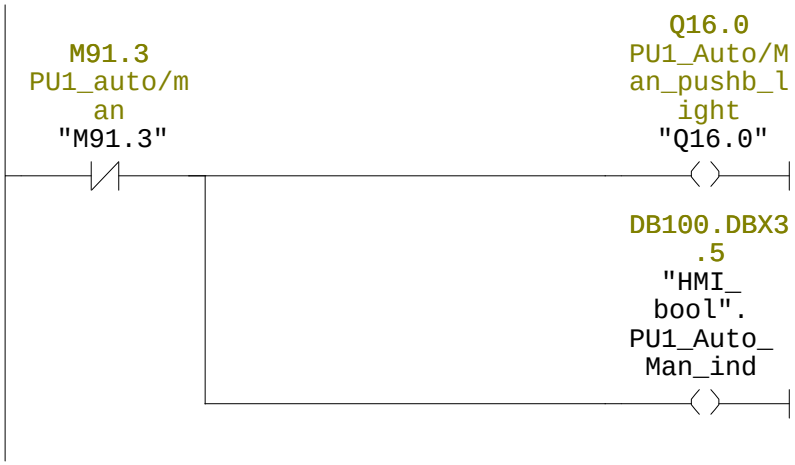


Network: 4 HMI / PB - Auto/man



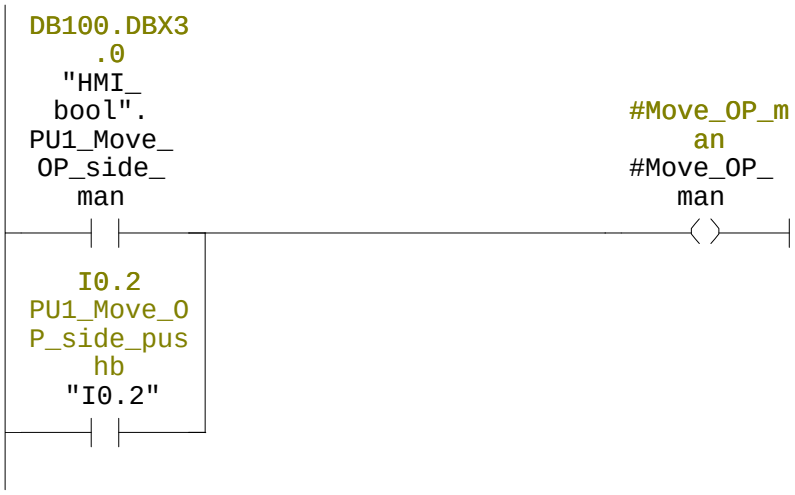
Liite 4
3(15)

Network: 5 PB light - Auto/man indicator



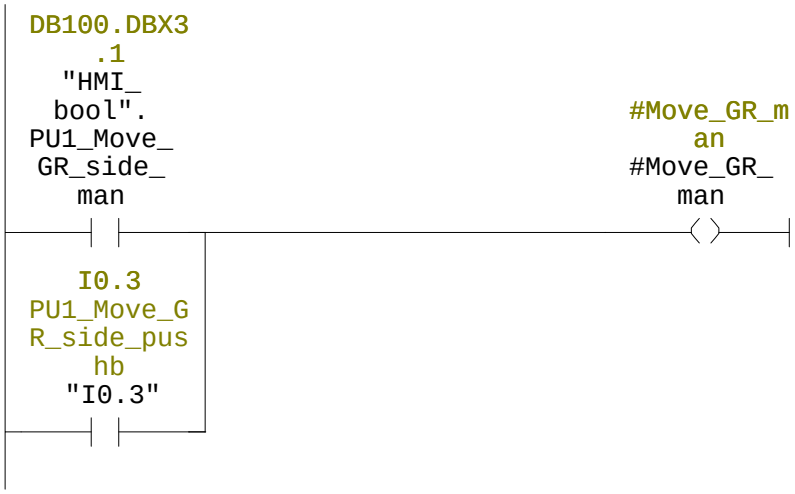
Network: 6 PB light - Servo up/down indicator

Network: 7 HMI / PB - move OP manual

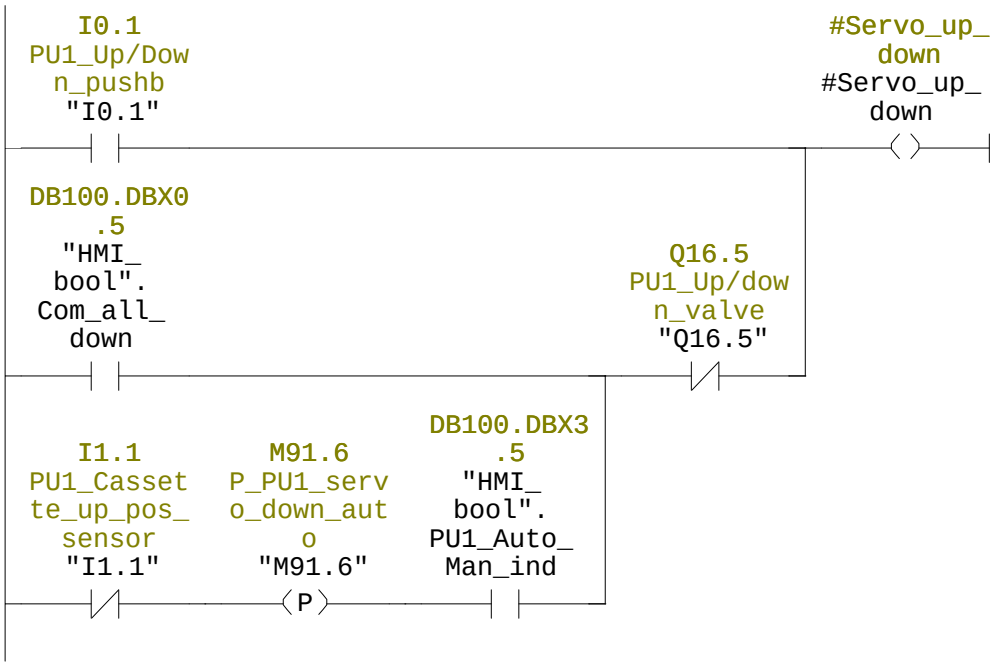


Liite 4
4(15)

Network: 8 HMI / PB - move GR manual



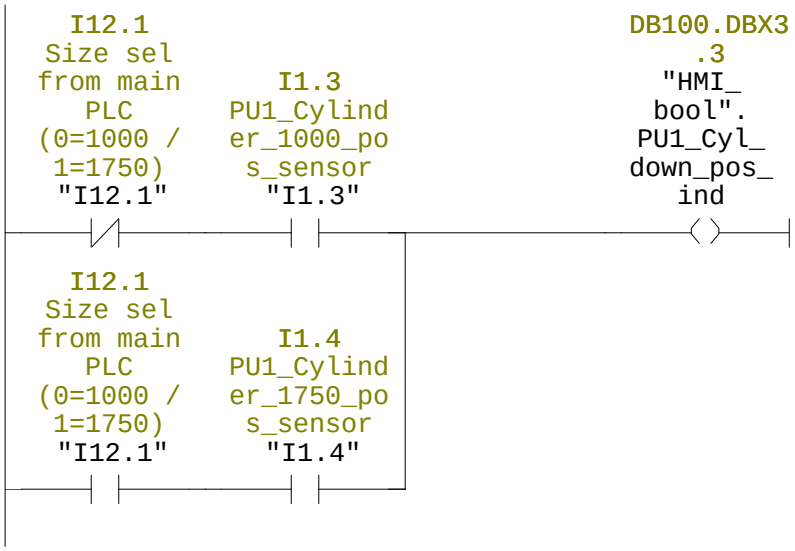
Network: 9 HMI / PB - Servo up/down



Network: 10 HMI - Servo up

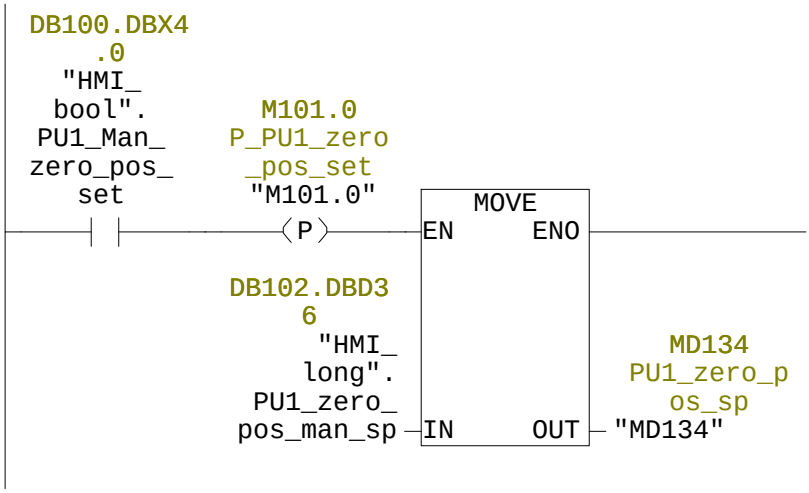
Liite 4
5(15)

Network: 11 HMI - Servo down

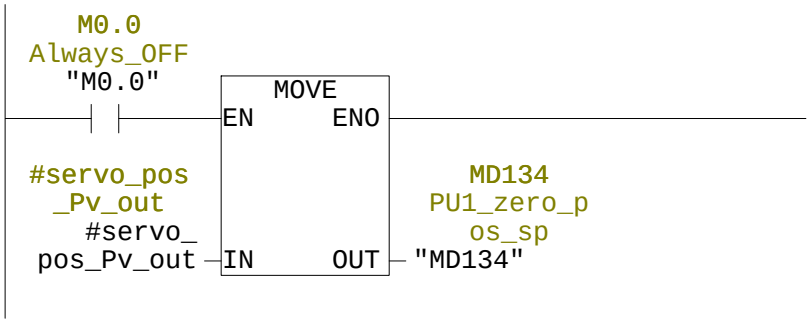


Network: 12 HMI - Hatch up

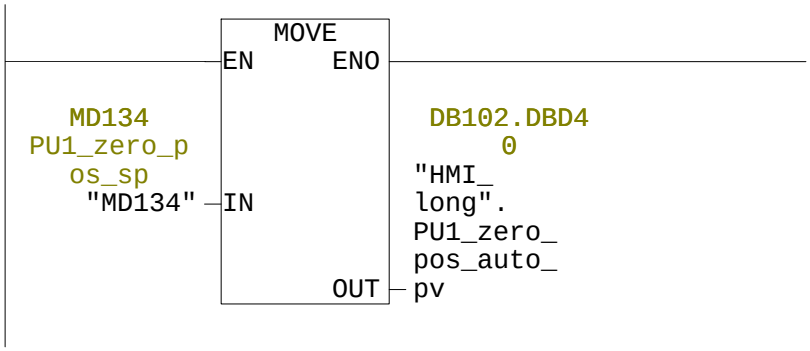
Network: 13 HMI - zero pos set man



Network: 14 HMI - zero pos set auto

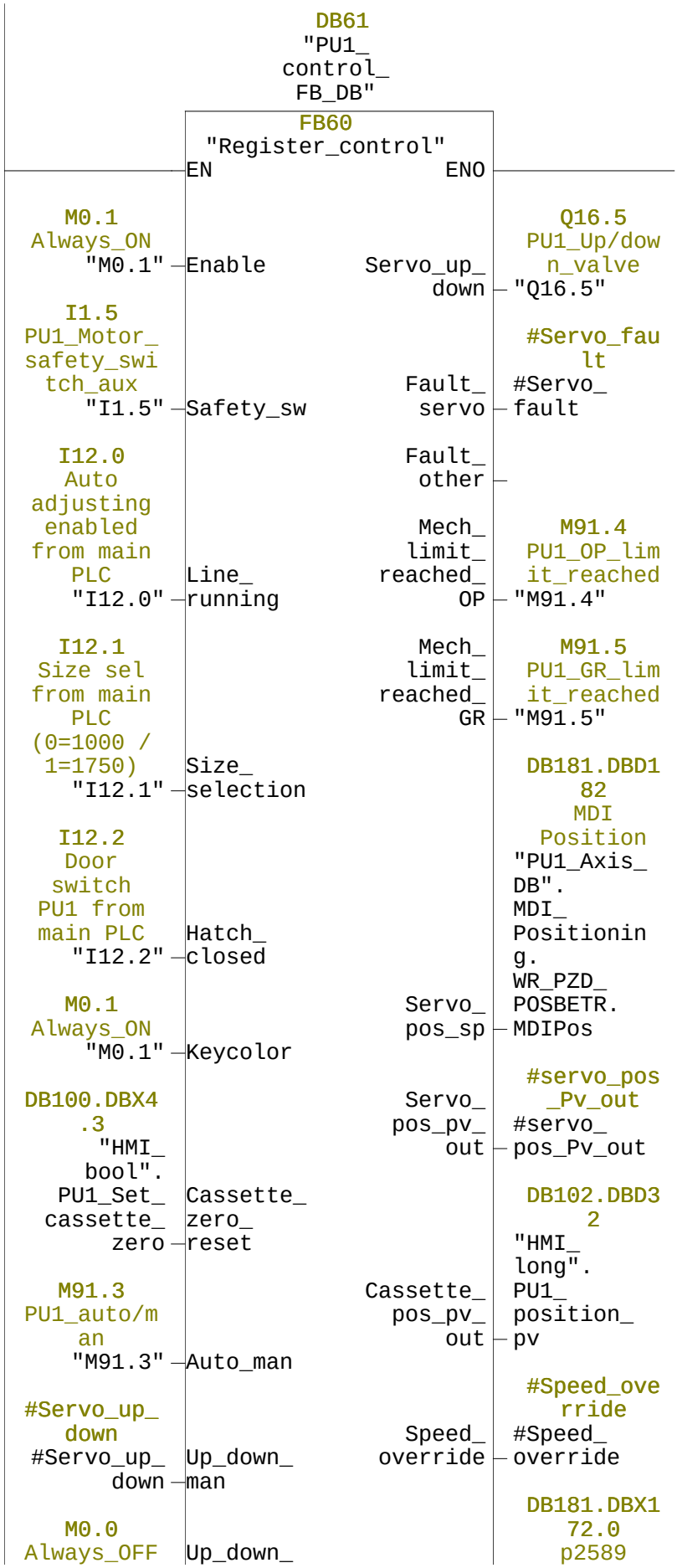


Network: 15 HMI - zero pos actual

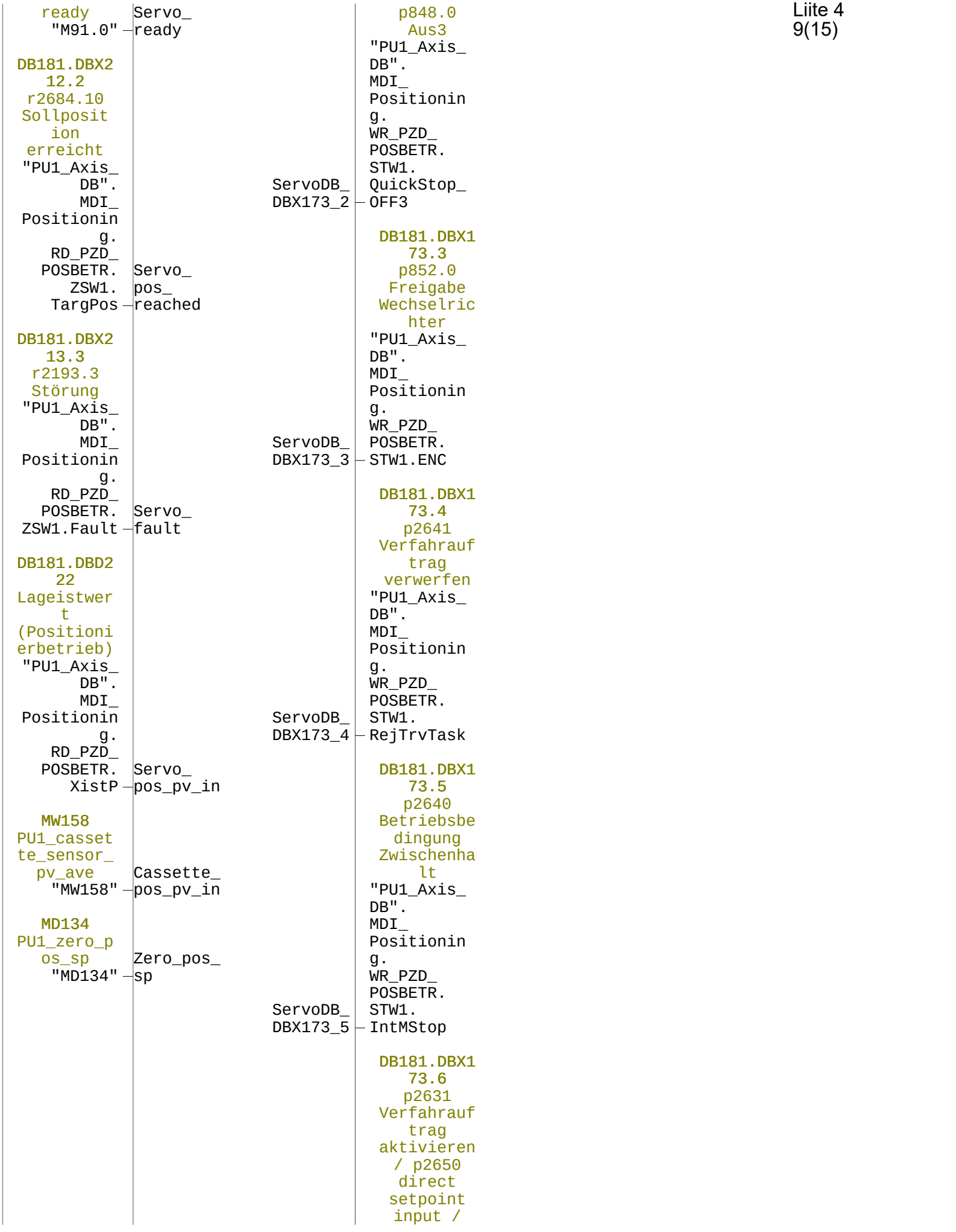


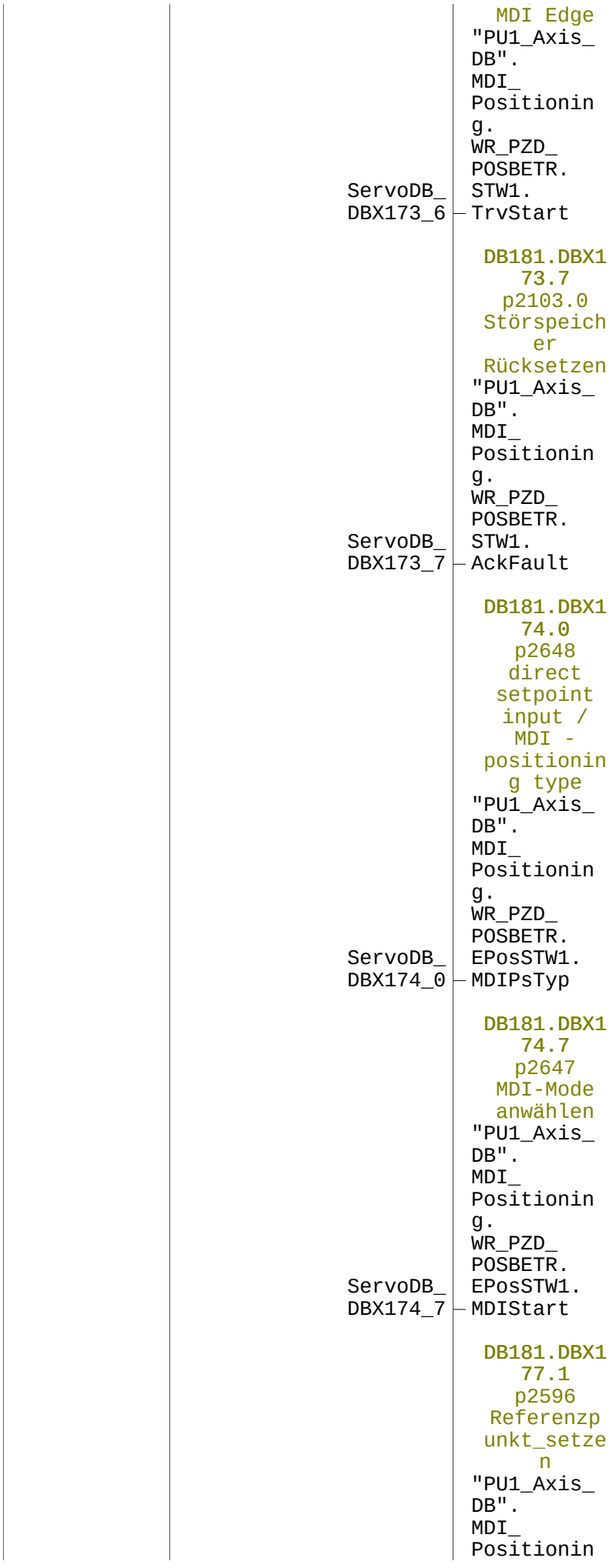
Network: 16 Servo control FB

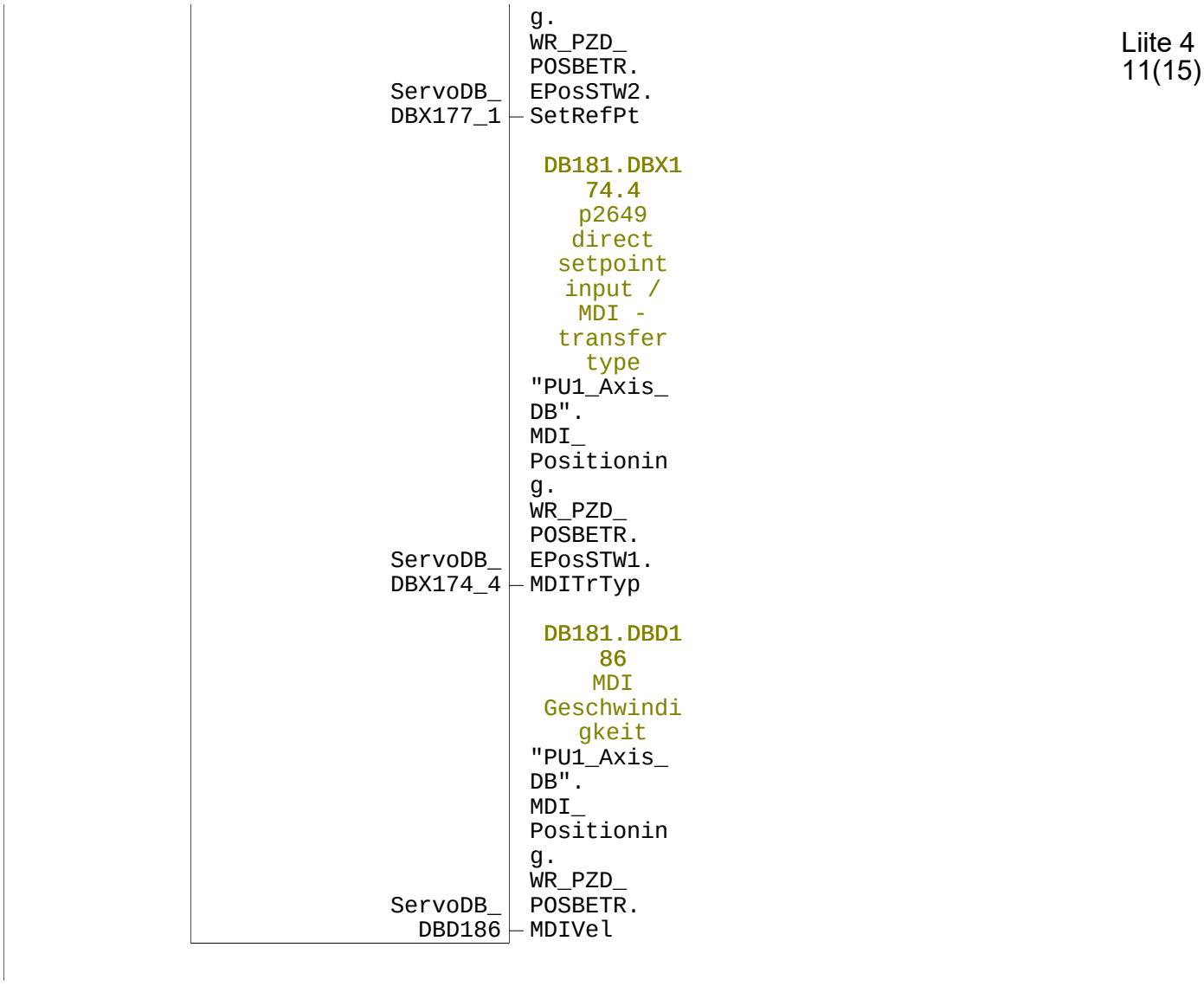
Liite 4
7(15)



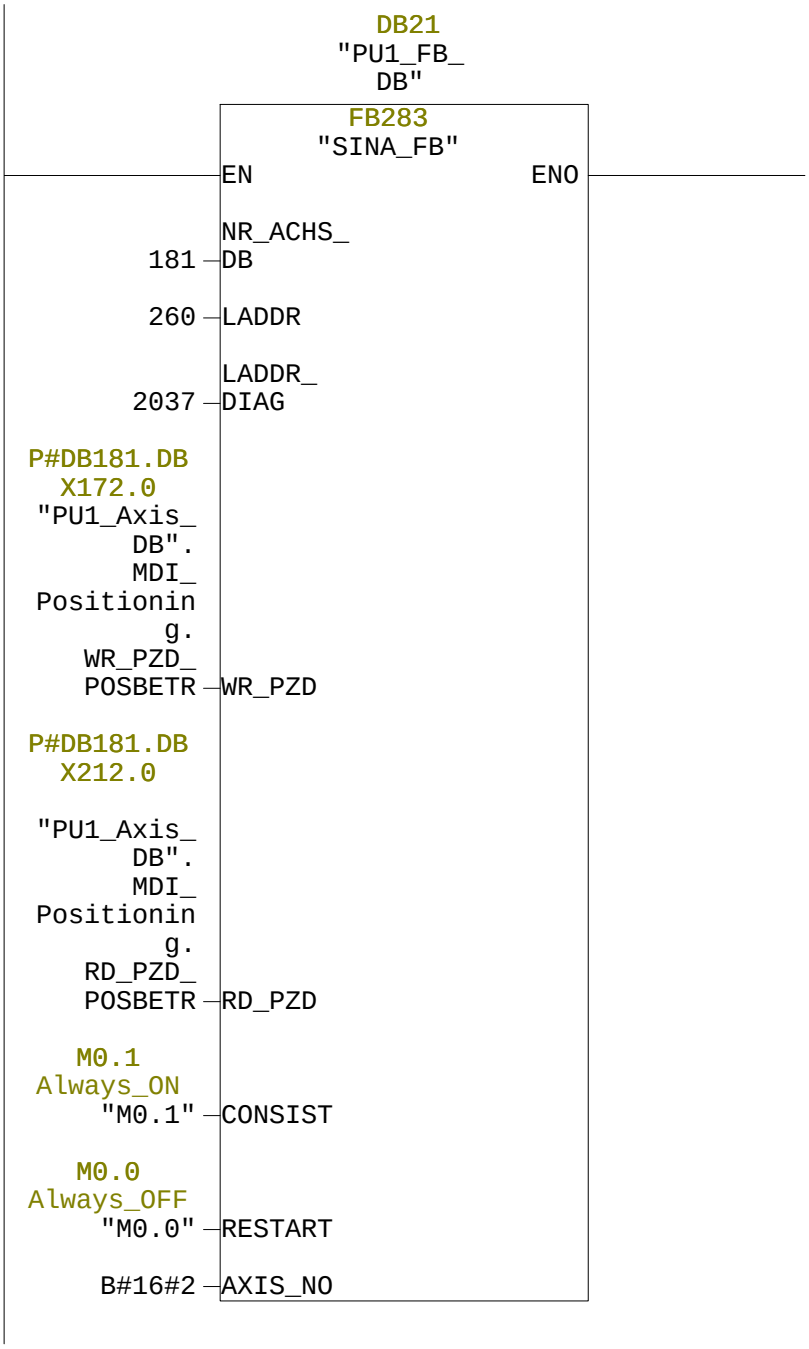
"M0.0"	auto		Tippen Signalquelle 1 ein
I14.2 PU1_move_OP_From_Regist	Move_OP_ext		"PU1_Axis_DB". MDI_ Positioning.
"I14.2"			
I14.3 PU1_move_GR_From_Regist	Move_GR_ext	ServoDB_ DBX172_0	WR_PZD_ POSBETR. STW1.Jog_1
"I14.3"			DB181.DBX172.1
#Move_OP_man			p2590 Tippen Signalquelle 2 ein
#Move_OP_man	Move_OP_int		"PU1_Axis_DB". MDI_ Positioning.
#Move_GR_man			
#Move_GR_man	Move_GR_int		WR_PZD_ POSBETR.
M90.1 PU_keycolor_to_OP	Keycolor_move_OP	ServoDB_ DBX172_1	STW1.Jog_2
"M90.1"			DB181.DBX172.2
M90.2 PU_keycolor_to_GR	Keycolor_move_GR		p854.0 Führung gefordert
"M90.2"			"PU1_Axis_DB". MDI_ Positioning.
I0.4 PU1_Move_to_zero_pos_pushb	Move_zero_man		WR_PZD_ POSBETR.
"I0.4"		ServoDB_ DBX172_2	STW1.LB
I1.1 PU1_Cassette_up_pos_sensor	Cassette_sensor		DB181.DBX173.0 p840.0 Aus1
"I1.1"			
I1.2 PU1_Cylinder_home_pos_sensor	Cyl_home_sensor		"PU1_Axis_DB". MDI_ Positioning.
"I1.2"			WR_PZD_ POSBETR.
I1.3 PU1_Cylinder_1000_pos_sensor	Cyl_MLl000_sensor	ServoDB_ DBX173_0	STW1.OFF1
"I1.3"			DB181.DBX173.1 p844.0 Aus2
I1.4 PU1_Cylinder_1750_pos_sensor	Cyl_MLl750_sensor		"PU1_Axis_DB". MDI_ Positioning.
"I1.4"			
#TEMP_FAULT_ACK			WR_PZD_ POSBETR. STW1.
#TEMP_FAULT_ACK	Servo_fault_ack	ServoDB_ DBX173_1	CoastStop_OFF2
M91.0 PU1_servo_			DB181.DBX173.2


ready
"M91.0"
DB181.DBX2 12.2 r2684.10 Sollposition erreicht
"PU1_Axis_DB".MDI_Positioning.RD_PZD_POSBETR.ZSW1.TargPos
DB181.DBX2 13.3 r2193.3 Störung
"PU1_Axis_DB".MDI_Positioning.RD_PZD_POSBETR.ZSW1.Fault
DB181.DBD22 Lageistwert (Positionierbetrieb)
"PU1_Axis_DB".MDI_Positioning.RD_PZD_POSBETR.XistP
MW158 PU1_cassette_sensor_pv_ave
"MW158"
MD134 PU1_zero_pos_sp
"MD134"
Servo_ready
Servo_pos_reached
Servo_fault
Cassette_pos_pv_in
Zero_pos_sp
Servo_pos_pv_in
ServoDB_DBX173_2
ServoDB_DBX173_3
ServoDB_DBX173_4
ServoDB_DBX173_5
p848.0 Aus3
"PU1_Axis_DB".MDI_Positioning.WR_PZD_POSBETR.STW1.QuickStop_OFF3
DB181.DBX173.3 p852.0 Freigabe Wechselrichter
"PU1_Axis_DB".MDI_Positioning.WR_PZD_POSBETR.STW1.ENC
DB181.DBX173.4 p2641 Verfahrauftrag verwerfen
"PU1_Axis_DB".MDI_Positioning.WR_PZD_POSBETR.STW1.RejTrvTask
DB181.DBX173.5 p2640 Betriebsbedingung Zwischenhalt
"PU1_Axis_DB".MDI_Positioning.WR_PZD_POSBETR.STW1.IntMStop
DB181.DBX173.6 p2631 Verfahrauftrag aktivieren / p2650 direct setpoint input /


MDI Edge
"PU1_Axis_
DB".
MDI_
Positionin
g.
WR_PZD_
POSBETR.
ServoDB_
STW1.
DBX173_6
TrvStart
DB181.DBX1
73.7
p2103.0
Störspeich
er
Rücksetzen
"PU1_Axis_
DB".
MDI_
Positionin
g.
WR_PZD_
POSBETR.
ServoDB_
STW1.
DBX173_7
AckFault
DB181.DBX1
74.0
p2648
direct
setpoint
input /
MDI -
positionin
g type
"PU1_Axis_
DB".
MDI_
Positionin
g.
WR_PZD_
POSBETR.
ServoDB_
EPosSTW1.
DBX174_0
MDIPsTyp
DB181.DBX1
74.7
p2647
MDI-Mode
anwählen
"PU1_Axis_
DB".
MDI_
Positionin
g.
WR_PZD_
POSBETR.
ServoDB_
EPosSTW1.
DBX174_7
MDIStart
DB181.DBX1
77.1
p2596
Referenzp
unkt_setze
n
"PU1_Axis_
DB".
MDI_
Positionin


g.
WR_PZD_
POSBETR.
ServoDB_
EPosSTW2.
DBX177_1
SetRefPt
DB181.DBX1
74.4
p2649
direct
setpoint
input /
MDI -
transfer
type
"PU1_Axis_
DB".
MDI_
Positionin
g.
WR_PZD_
POSBETR.
ServoDB_
EPosSTW1.
DBX174_4
MDITrTyp
DB181.DBD1
86
MDI
Geschwindi
gkeit
"PU1_Axis_
DB".
MDI_
Positionin
g.
WR_PZD_
ServoDB_
POSBETR.
DBD186
MDIVel

Network: 17 Servo communication FB

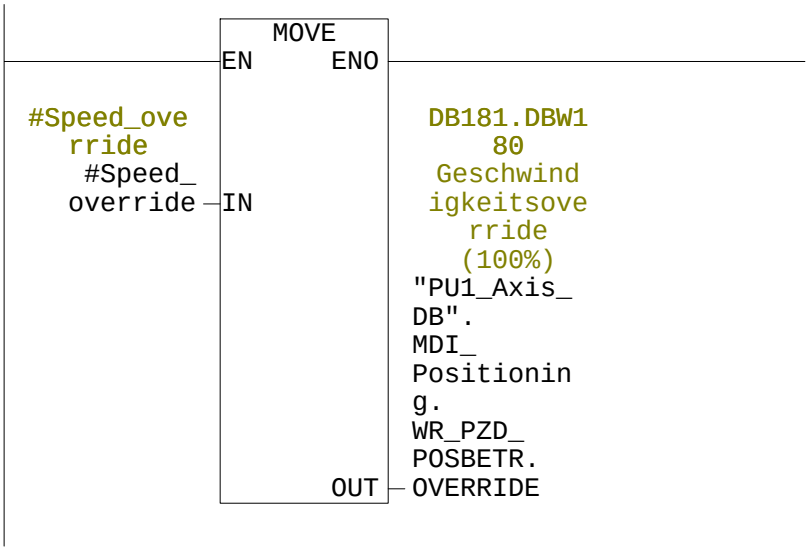


Liite 4
12(15)

Network: 18 Speed

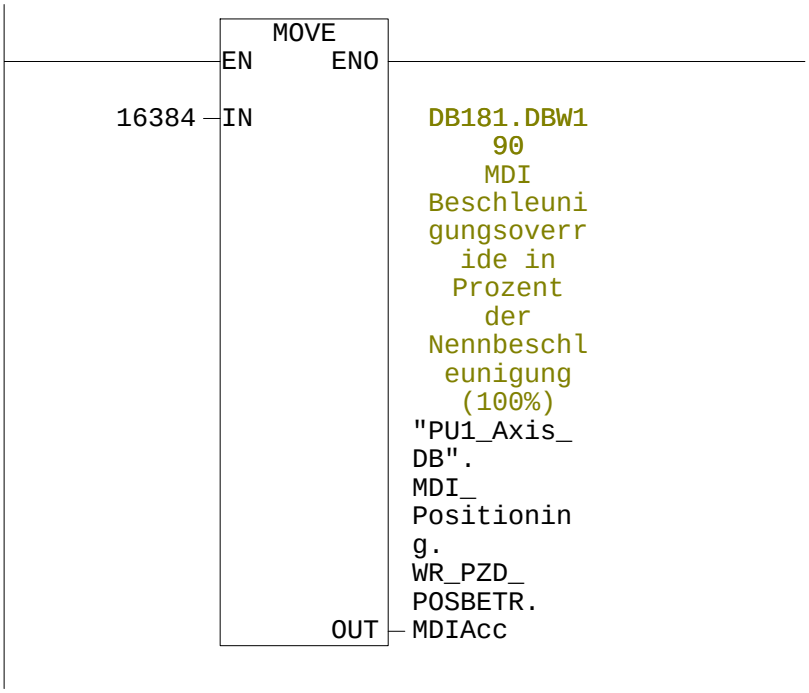
0-100% <--> 0-16384

Jog=8000
MDI=16384

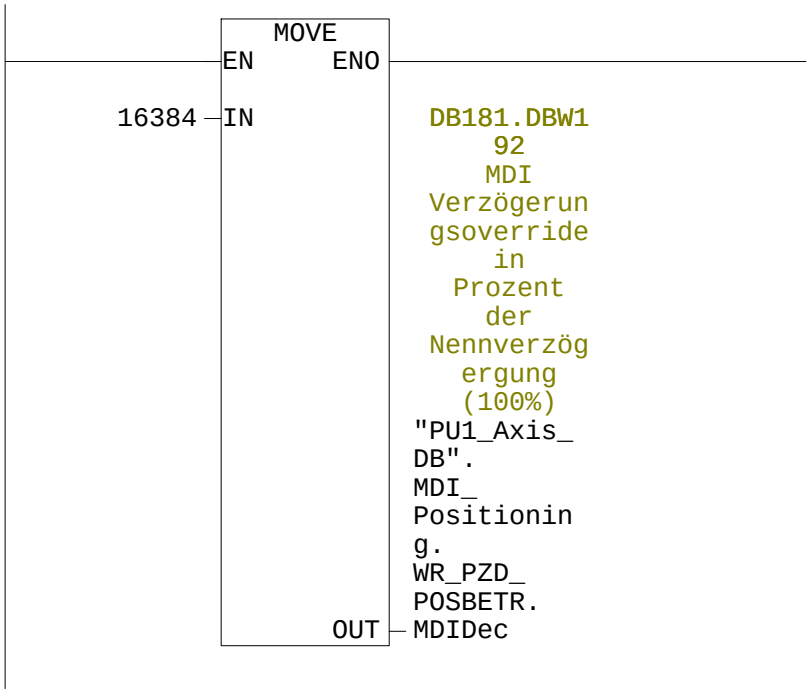


Liite 4
13(15)

Network: 19 Acceleration

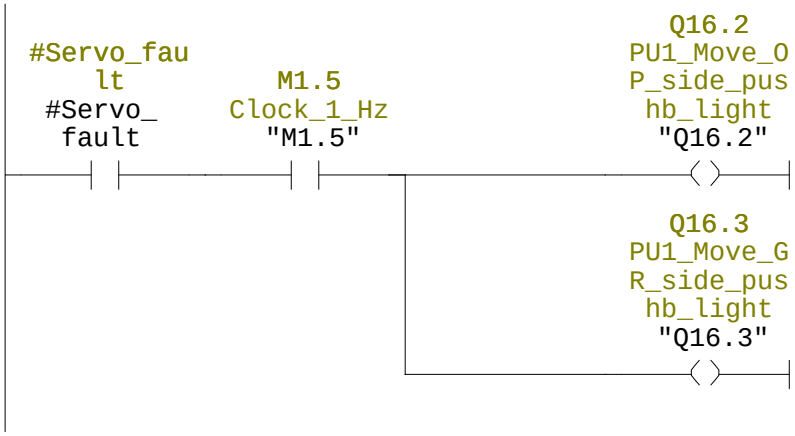


Network: 20 Deceleration



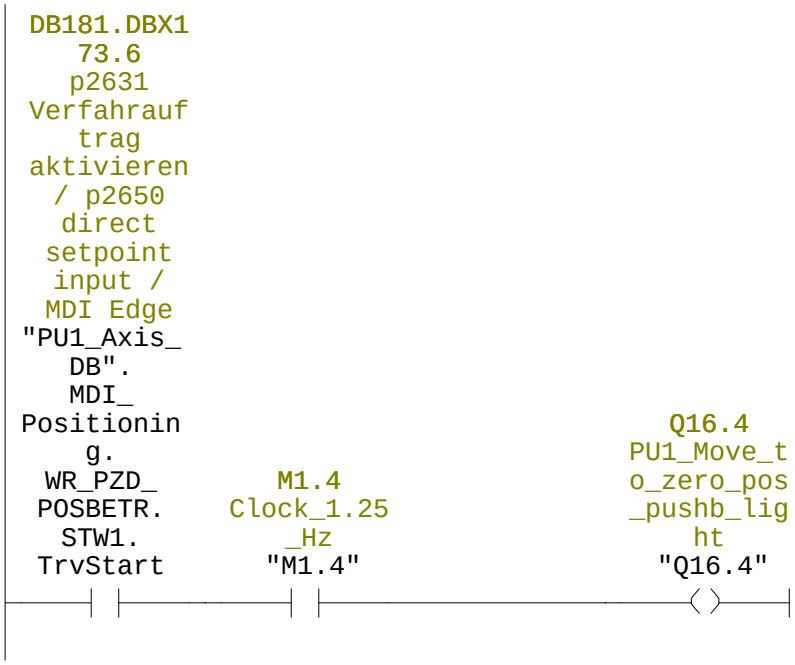
Liite 4
14(15)

Network: 21 PB light - Fault indicator



Network: 22 PB light - Move to zero pos indicator

Liite 4
15(15)



Liite 5
1(27)

FB60 - <offline>

"Register_control"

Name: **Family:**
Author: TPi **Version:** 0.1
Block version: 2
Time stamp Code: 03/05/2019 02:23:11 PM
Interface: 09/28/2018 03:25:45 PM
Lengths (block/logic/data): 02636 02070 00092

Name	Data Type	Address	Initial Value	Comment
IN		0.0		
Enable	Bool	0.0	FALSE	Control block is enabled
Safety_sw	Bool	0.1	FALSE	Motor safety switch
Line_running	Bool	0.2	FALSE	Line is running / auto adjust enabled
Size_selection	Bool	0.3	FALSE	Size selection (1000ml/1750ml)
Hatch_closed	Bool	0.4	FALSE	Safety hatch in down position sensor
Keycolor	Bool	0.5	FALSE	Keycolor-unit
Cassette_zero_reset	Bool	0.6	FALSE	Reset cassette actual zero point
Auto_man	Bool	0.7	FALSE	Automatic / manual selection
Up_down_man	Bool	1.0	FALSE	Servo up or down (manual mode)
Up_down_auto	Bool	1.1	FALSE	Servo up or down (automatic mode)
Move_OP_ext	Bool	1.2	FALSE	External move to OP-side (automatic mode)
Move_GR_ext	Bool	1.3	FALSE	External move to GR-side (automatic mode)
Move_OP_int	Bool	1.4	FALSE	Internal move to OP-side (manual mode)
Move_GR_int	Bool	1.5	FALSE	Internal move to GR-side (manual mode)
Keycolor_move_OP	Bool	1.6	FALSE	Keycolor move to OP-side
Keycolor_move_GR	Bool	1.7	FALSE	Keycolor move to GR-side
Move_zero_man	Bool	2.0	FALSE	Move to zero position (manual mode)
Cassette_sensor	Bool	2.1	FALSE	Cassette in drive position sensor
Cyl_home_sensor	Bool	2.2	FALSE	Servo in home position sensor
Cyl_MLl000_sensor	Bool	2.3	FALSE	Servo in 1000ml cassette position sensor
Cyl_MLl750_sensor	Bool	2.4	FALSE	Servo in 1750ml cassette position sensor
Servo_fault_ack	Bool	2.5	FALSE	Servo fault reset
Servo_ready	Bool	2.6	FALSE	Servo is ready to move (from servo)
Servo_pos_reached	Bool	2.7	FALSE	Servo has reached set position
Servo_fault	Bool	3.0	FALSE	Servo fault
Servo_pos_pv_in	DInt	4.0	L#0	Actual position from servo
Cassette_pos_pv_in	Int	8.0	0	Actual position from analog sensor
OUT		0.0		
Servo_up_down	Bool	10.0	FALSE	Up/down valve control
Fault_servo	Bool	10.1	FALSE	Common servo fault
Fault_other	Bool	10.2	FALSE	Common fault
Mech_limit_reached_OP	Bool	10.3	FALSE	Keycolor needs to move to GR
Mech_limit_reached_GR	Bool	10.4	FALSE	Keycolor needs to move to OP
Servo_pos_sp	DInt	12.0	L#0	Position sp for servo
Servo_pos_pv_out	Real	16.0	0.000000e+000	Actual servo position (+-3mm)
Cassette_pos_pv_out	Real	20.0	0.000000e+000	Actual cassette position (+-3mm)
Speed_override	DInt	24.0	L#0	0-16384
ServoDB_DBX172_0	Bool	28.0	FALSE	Jog dir 1
ServoDB_DBX172_1	Bool	28.1	FALSE	Jog dir 2
ServoDB_DBX172_2	Bool	28.2	FALSE	MDI pos
ServoDB_DBX173_0	Bool	28.3	FALSE	Pos betr stw1 off1
ServoDB_DBX173_1	Bool	28.4	FALSE	Coast stop off 2
ServoDB_DBX173_2	Bool	28.5	FALSE	Quick stop off 3
ServoDB_DBX173_3	Bool	28.6	FALSE	Pos betr
ServoDB_DBX173_4	Bool	28.7	FALSE	Rej tru task
ServoDB_DBX173_5	Bool	29.0	FALSE	Int m stop
ServoDB_DBX173_6	Bool	29.1	FALSE	Trvs start

Name	Data Type	Address	Initial Value	Comment
ServoDB_DBX173_7	Bool	29.2	FALSE	Fault reset
ServoDB_DBX174_0	Bool	29.3	FALSE	MDI ps typ
ServoDB_DBX174_7	Bool	29.4	FALSE	MDI mode on
ServoDB_DBX177_1	Bool	29.5	FALSE	Pos reset
ServoDB_DBX174_4	Bool	29.6	FALSE	MDI tr typ
ServoDB_DBD186	DInt	30.0	L#0	MDI vel
IN_OUT		0.0		
Zero_pos_sp	Real	34.0	0.000000e+000	Zero point sp value from HMI
STAT		0.0		
Jog_OP	Bool	38.0	FALSE	
Jog_GR	Bool	38.1	FALSE	
Jog_down	Bool	38.2	FALSE	
Ref_need	Bool	38.3	FALSE	
Ref_ok	Bool	38.4	FALSE	
Ref_enabled	Bool	38.5	FALSE	
Ref_auto_start	Bool	38.6	FALSE	
Ref_run	Bool	38.7	FALSE	
Zero_point	DInt	40.0	L#0	
Enabled	Bool	44.0	FALSE	
Keycolor_OP	Bool	44.1	FALSE	
Keycolor_GR	Bool	44.2	FALSE	
P1	Bool	44.3	FALSE	
P2	Bool	44.4	FALSE	
P3	Bool	44.5	FALSE	
P4	Bool	44.6	FALSE	
P5	Bool	44.7	FALSE	
P6	Bool	45.0	FALSE	
P7	Bool	45.1	FALSE	
P8	Bool	45.2	FALSE	
P9	Bool	45.3	FALSE	
Up_down_pulse	Bool	45.4	FALSE	
Jog_down_time_full	Bool	45.5	FALSE	
Start	Bool	45.6	FALSE	
Ref_start_delay	Bool	45.7	FALSE	
Keycolor_max_time	Bool	46.0	FALSE	
Timer1	TON	48.0		Jog_down
Timer2	TOF	70.0		
Timer3	TON	92.0		
Timer4	TON	114.0		
Timer5	TOF	136.0		
Timer6	TON	158.0		
P10	Bool	180.0	FALSE	
Cassette_mech_zero	Real	182.0	0.000000e+000	
Mech_limit_OP	Bool	186.0	FALSE	
Mech_limit_GR	Bool	186.1	FALSE	
P11	Bool	186.2	FALSE	
Timer7	TOF	188.0		
Faster_auto_adj	Bool	210.0	FALSE	
TEMP		0.0		
Temp1	Real	0.0		
Temp2	Real	4.0		
temp3	Real	8.0		
Temp4	Real	12.0		
Temp5	Real	16.0		
Temp6	Real	20.0		
Temp7	Real	24.0		
Temp8	Real	28.0		

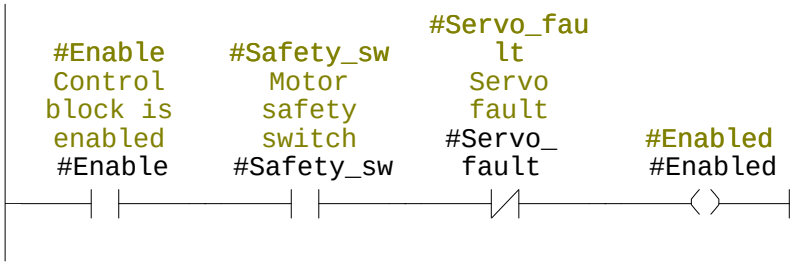
Liite 5
3(27)

Name	Data Type	Address	Initial Value	Comment
Time1	Time	32.0		
Time2	Time	36.0		
Time3	Time	40.0		
Time4	Time	44.0		
Time5	Time	48.0		
Time6	Time	52.0		
Temp9	Real	56.0		
Temp10	Real	60.0		
Scale_retval	Word	64.0		
Time7	Time	66.0		
Temp11	Bool	70.0		
Temp12	Real	72.0		

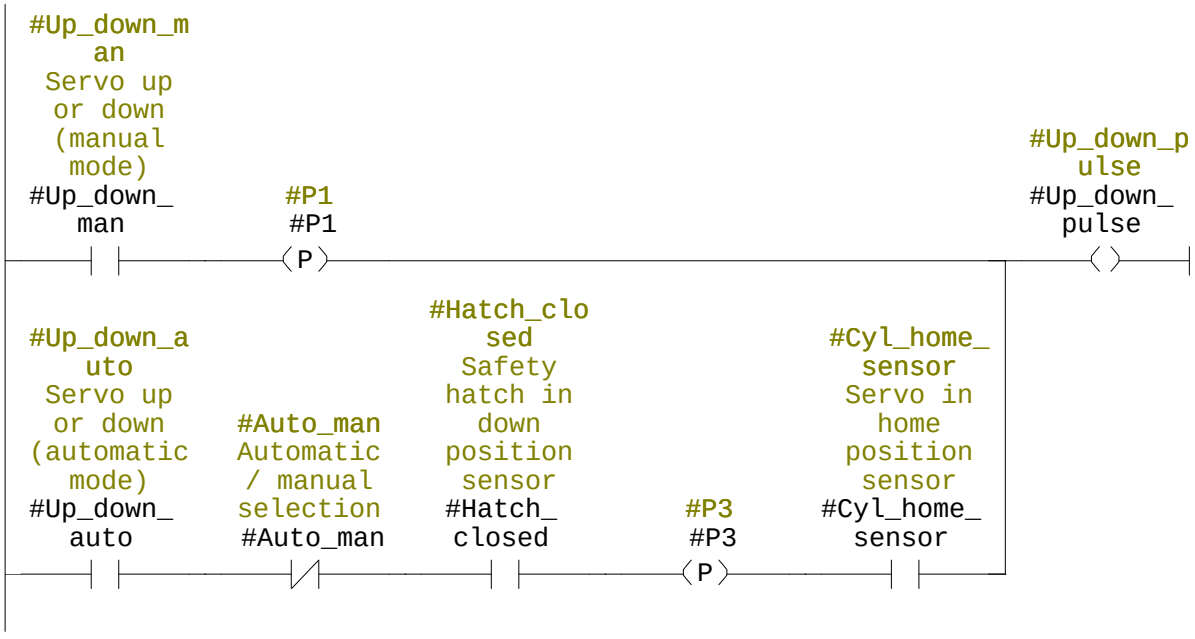
Block: FB60 Side register control

Tetra Pak / T. Pinomäki 5.3.2019

Network: 1 Enable

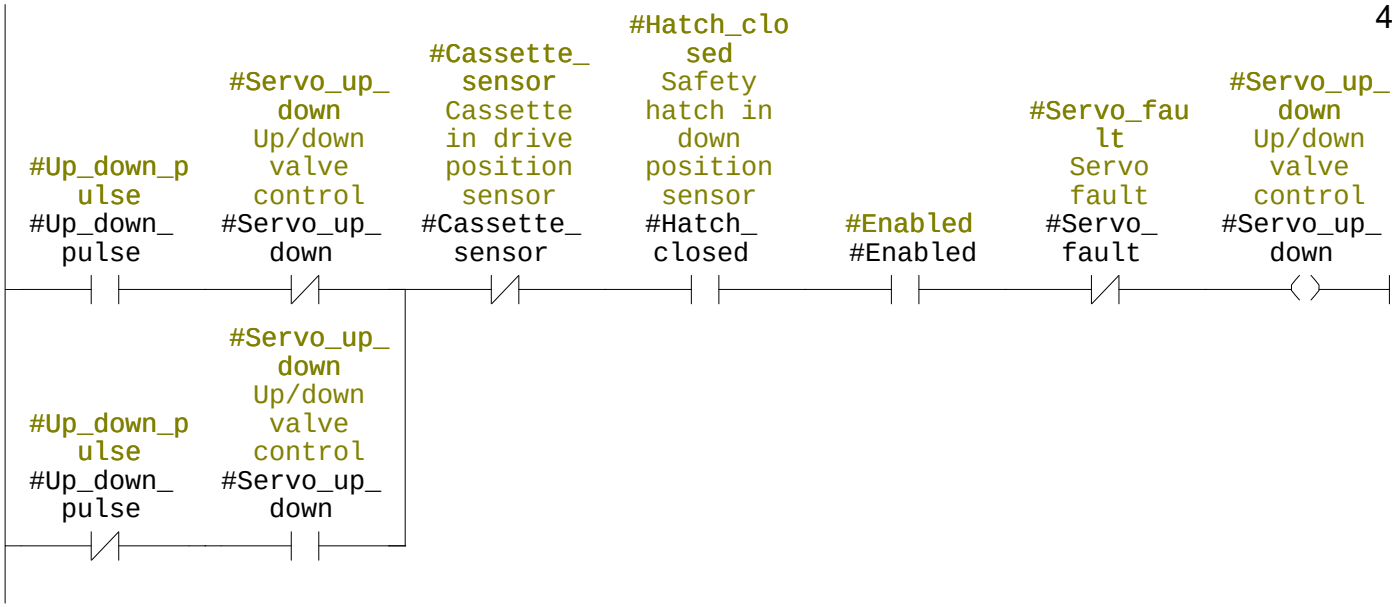


Network: 2 Servo up or down pulse

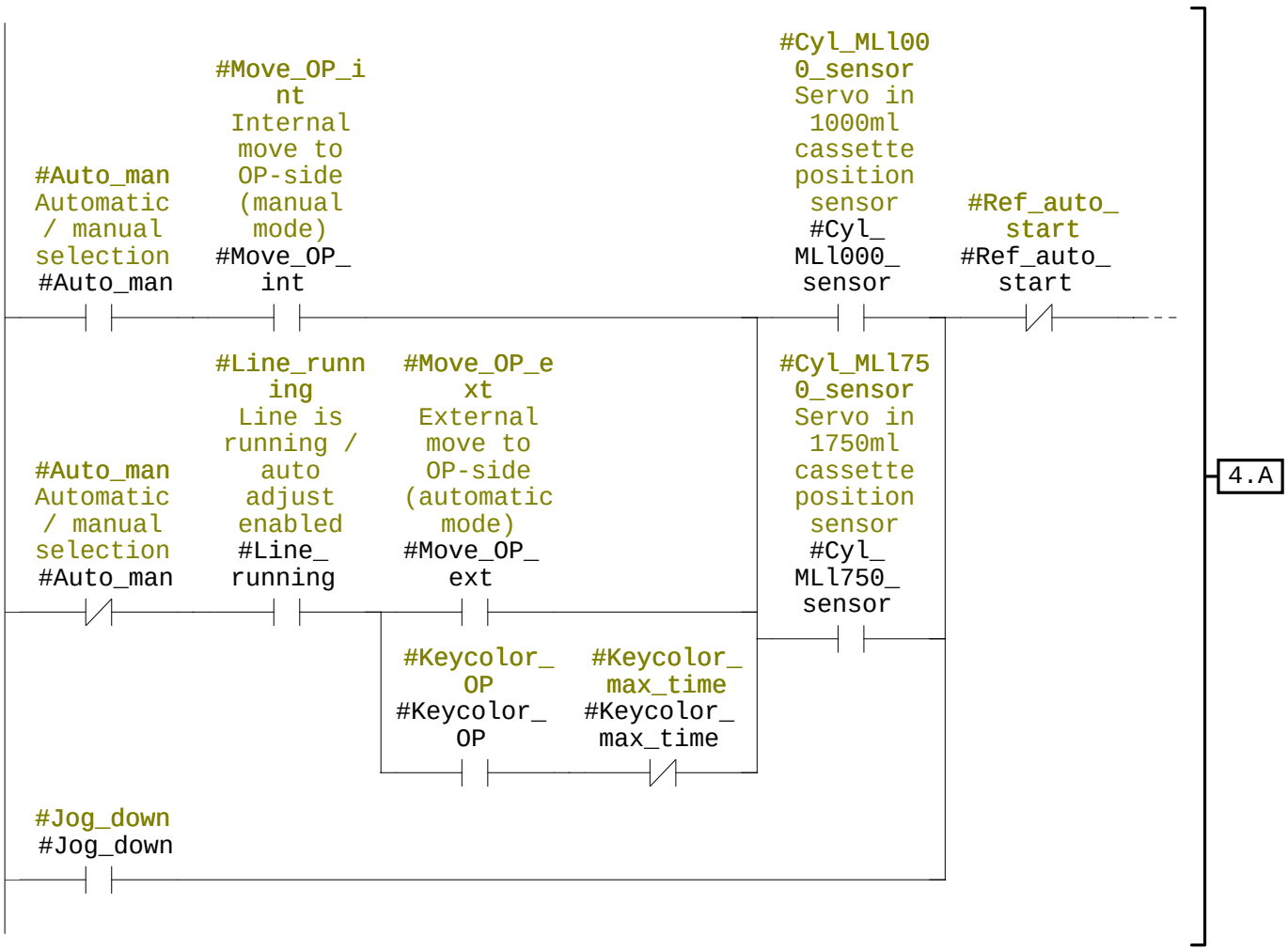


Network: 3 Servo up or down

Liite 5
4(27)

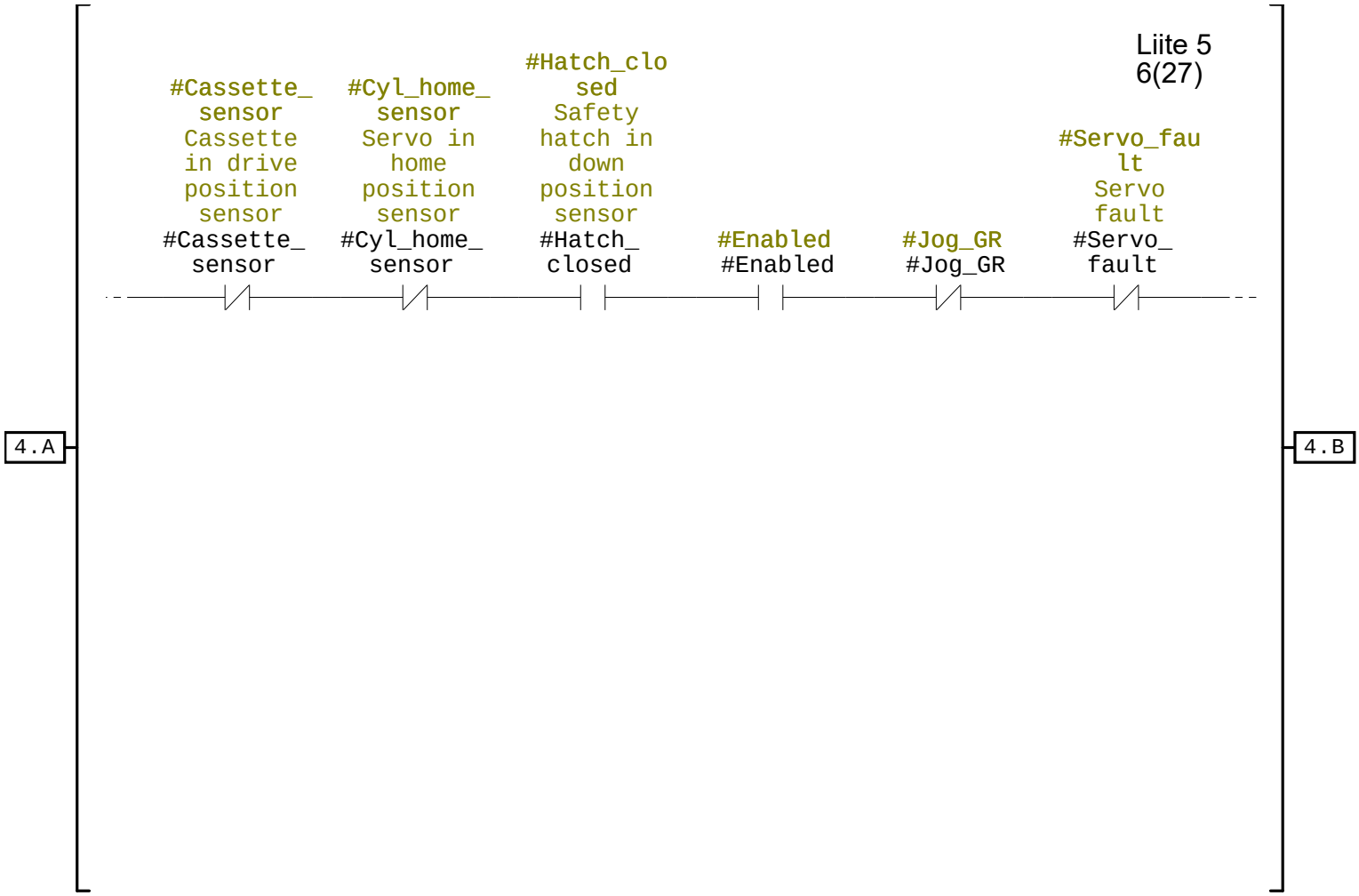


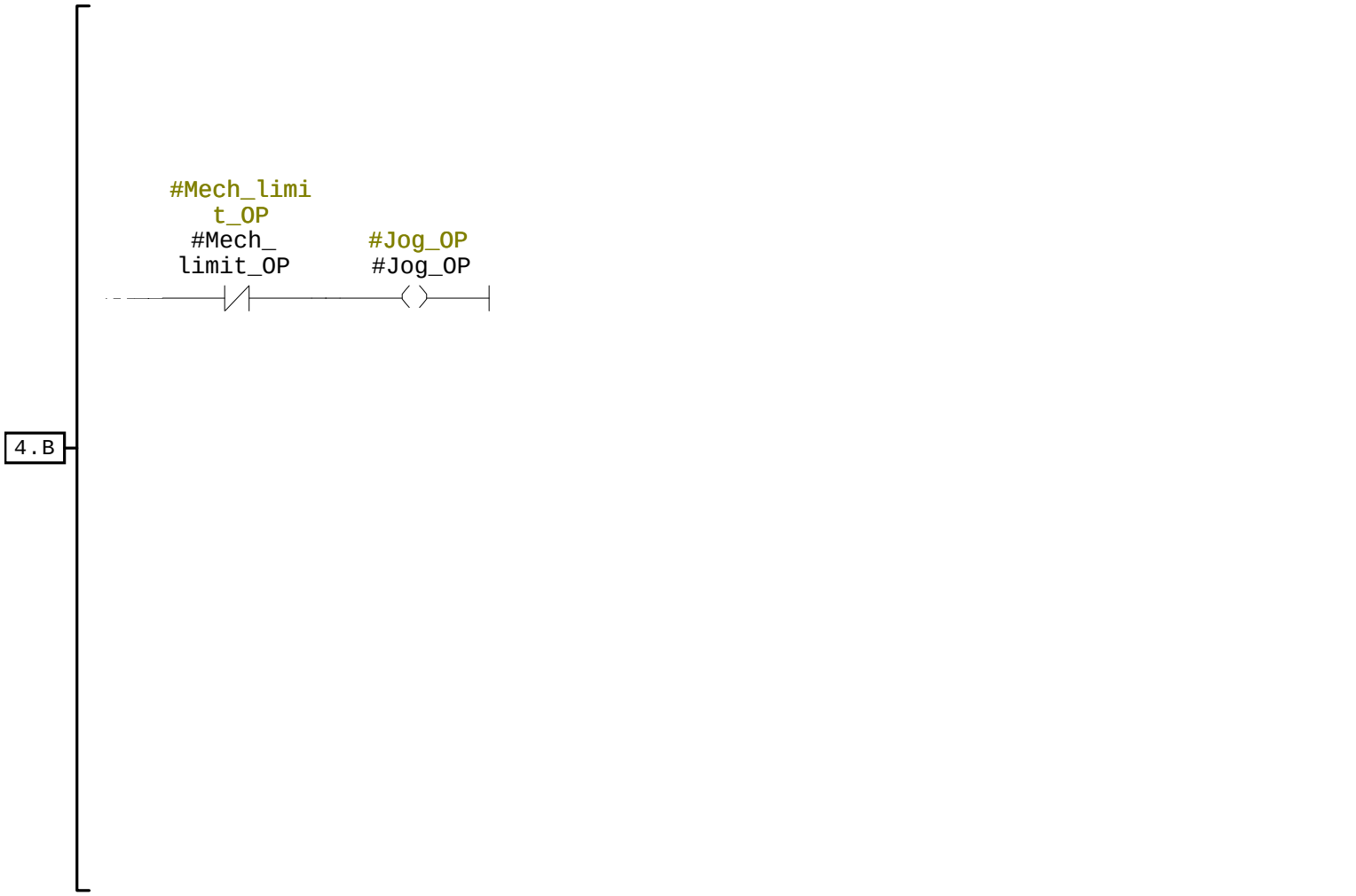
Network: 4 Jog OP-side



Liite 5
5(27)

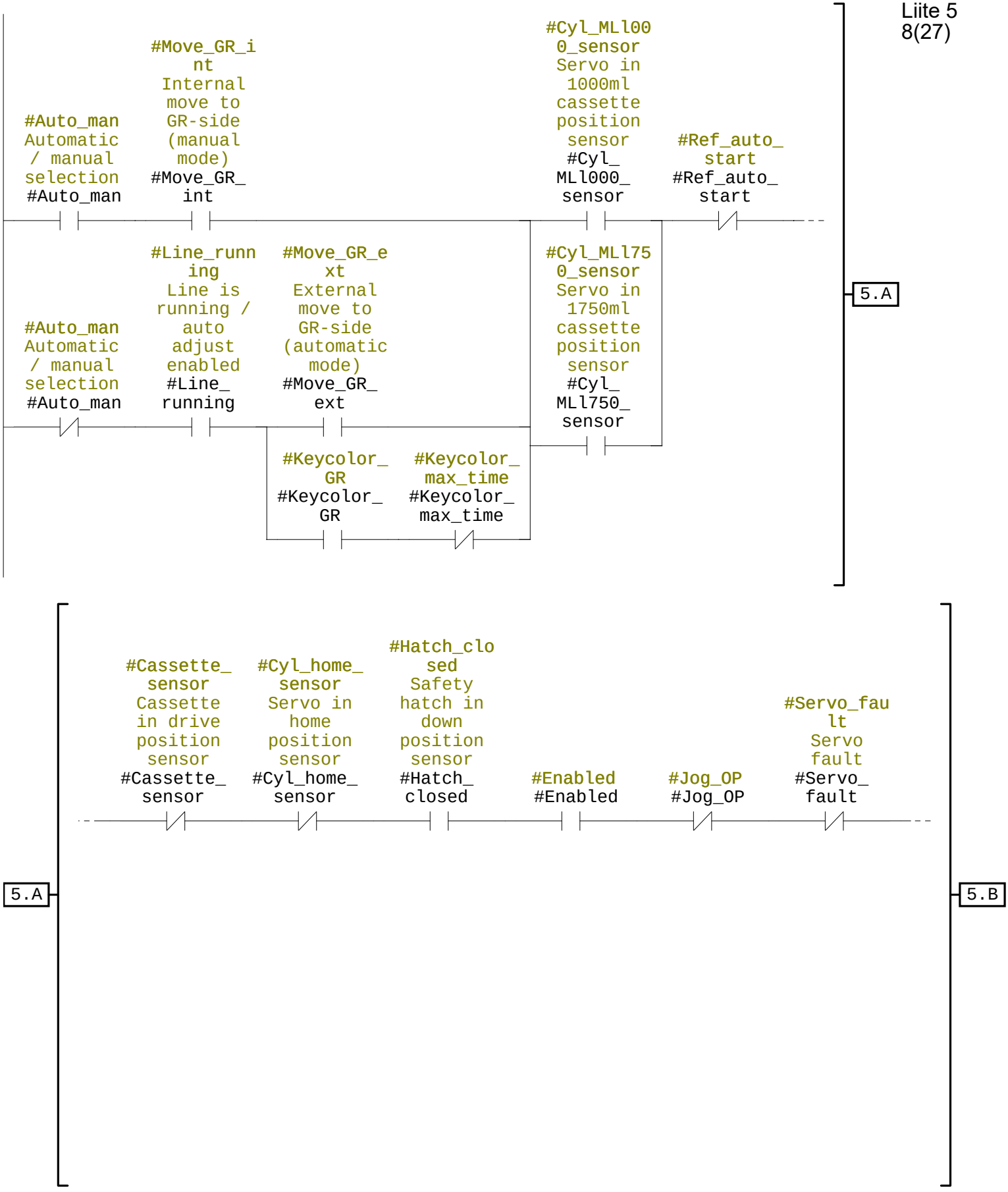
Liite 5
6(27)

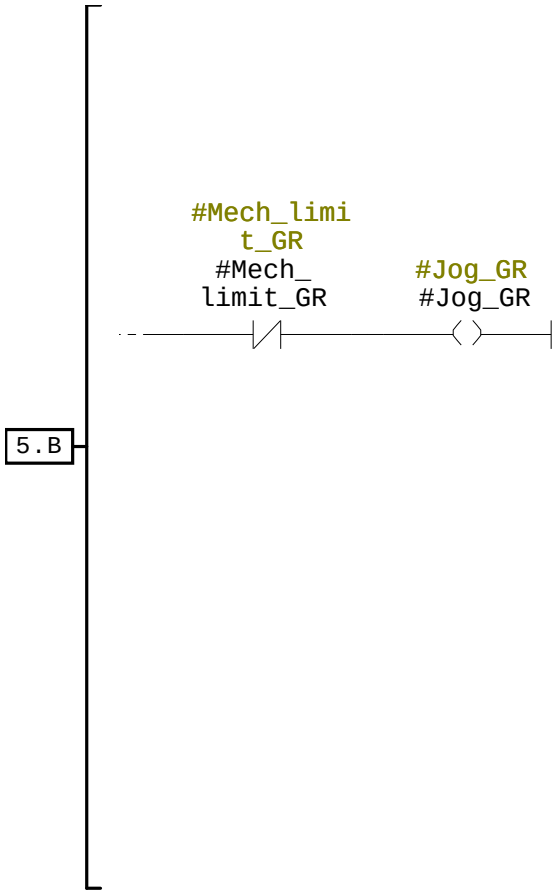

#Cassette_ sensor
Cassette in drive position sensor
#Cassette_ sensor
#Cyl_home_ sensor
Servo in home position sensor
#Cyl_home_ sensor
#Hatch_closed
Safety hatch in down position sensor
#Hatch_ closed
#Enabled
#Enabled
#Jog_GR
#Jog_GR
#Servo_fault
Servo fault
#Servo_ fault
4.A
4.B


#Mech_limit_OP
#Mech_ limit_OP
#Jog_OP
#Jog_OP
4.B

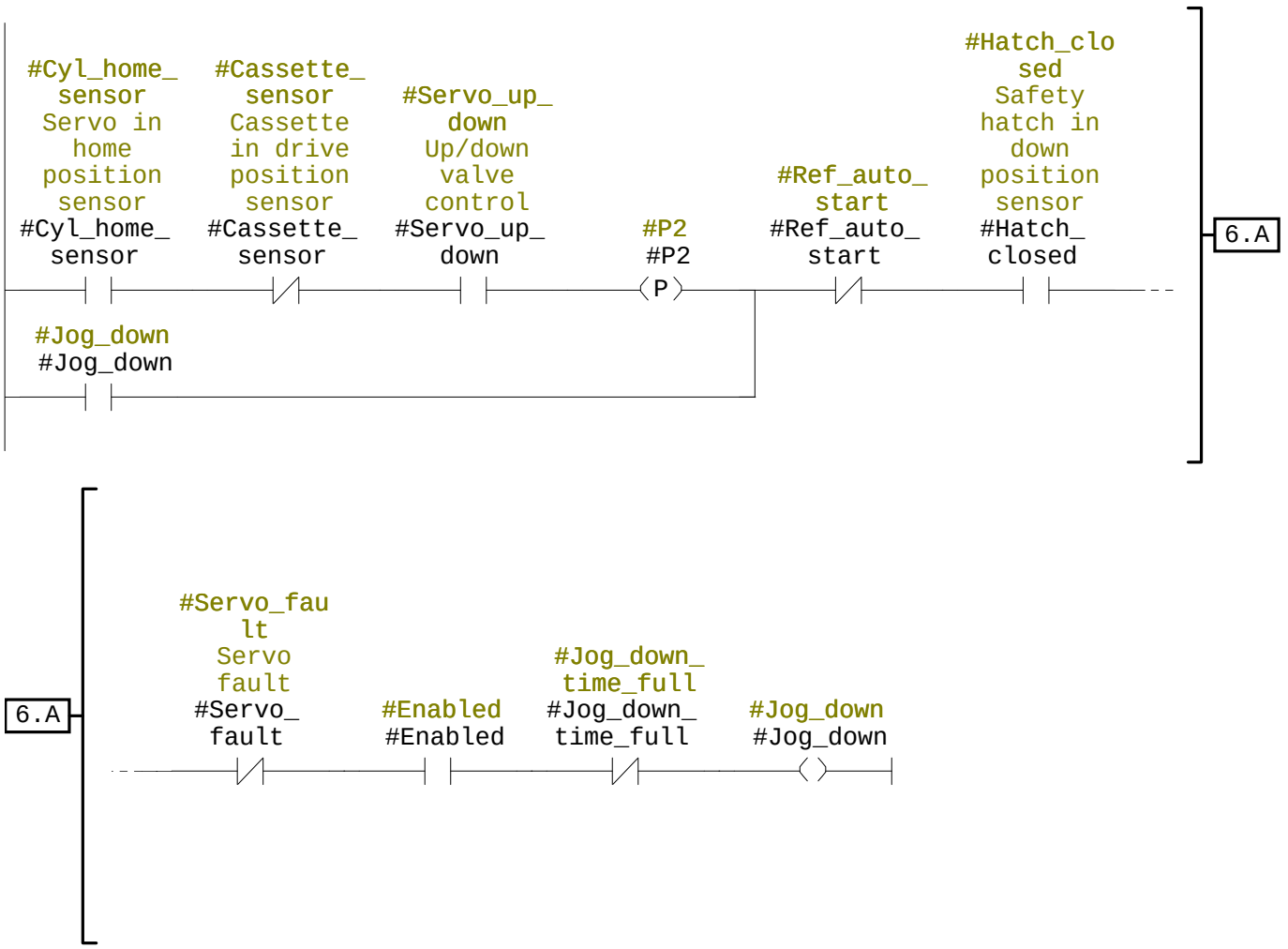
Network: 5 Jog GR-side

Liite 5
8(27)

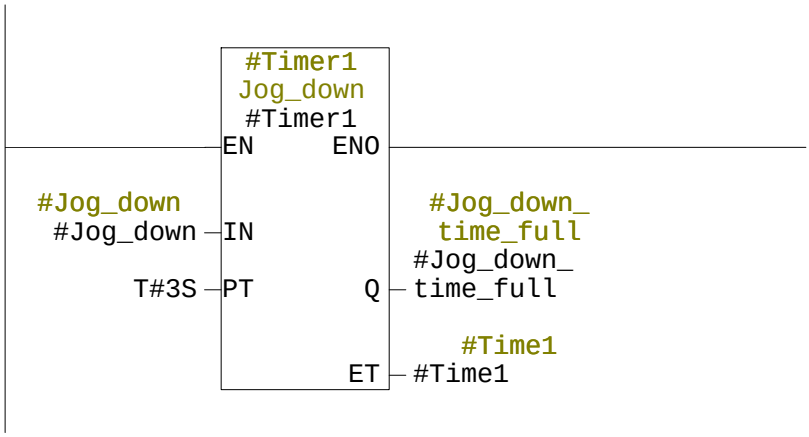



#Mech_limi
t_GR
#Mech_
limit_GR
#Jog_GR
#Jog_GR
5.B

Network: 6 Servo down Jog

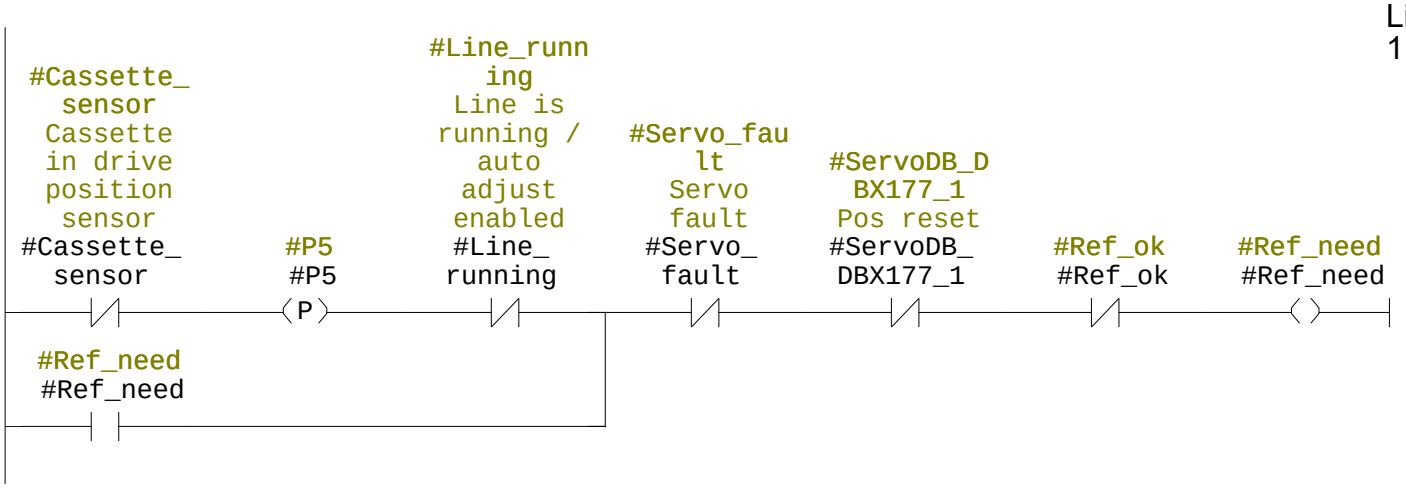


Network: 7 Jog down time

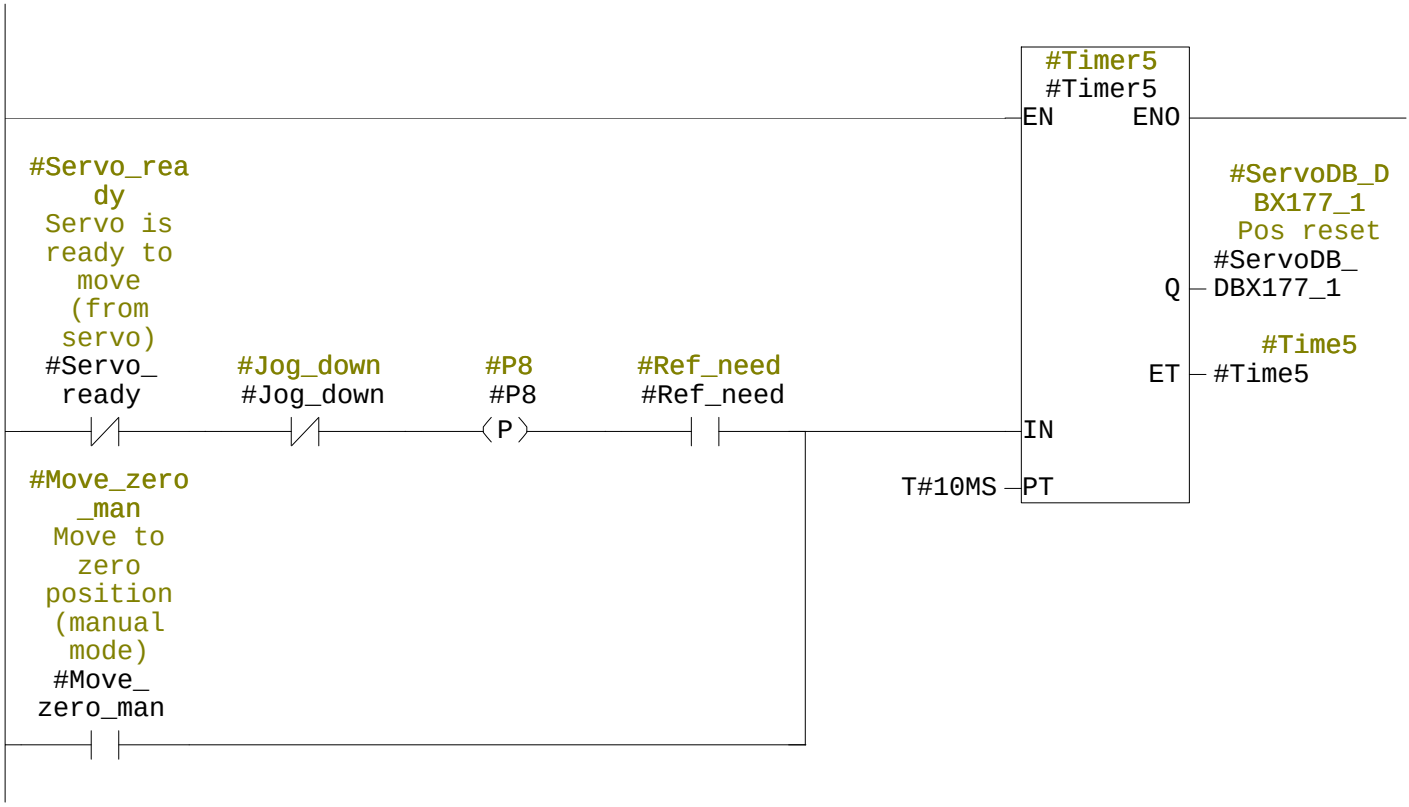


Liite 5
11(27)

Network: 8 Need of reference run

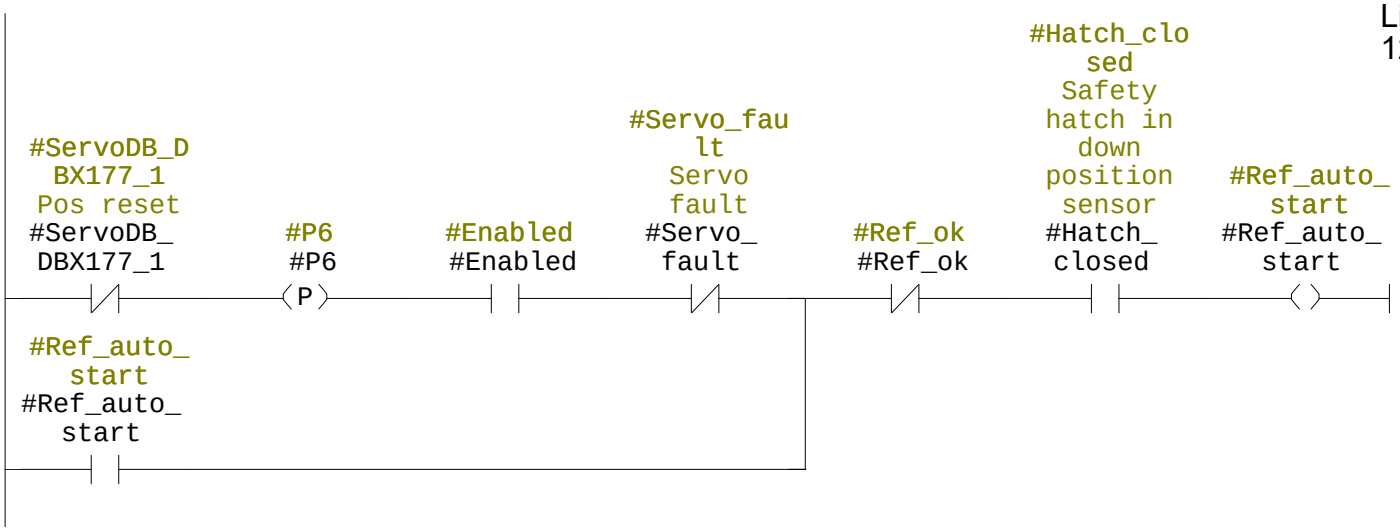


Network: 9 Reset servo position

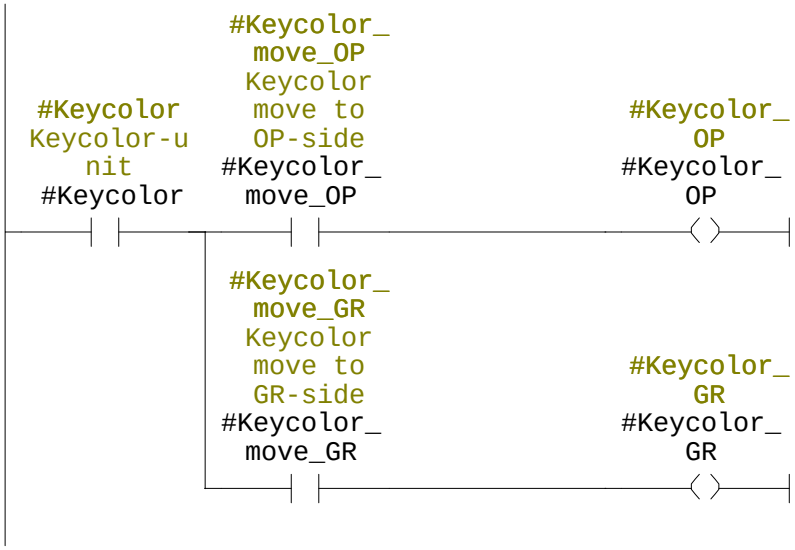


Liite 5
12(27)

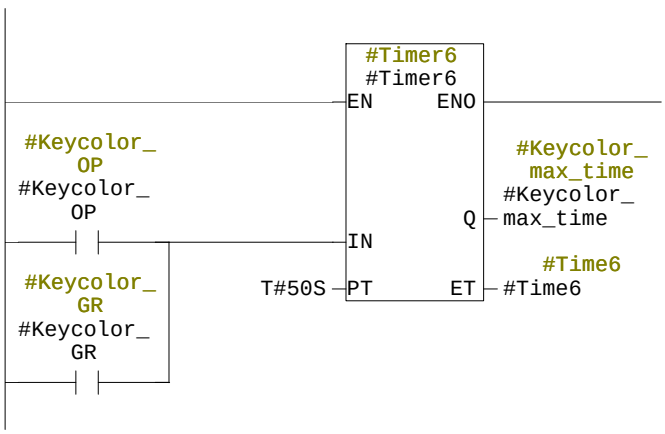
Network: 10 Start auto referencing



Network: 11 Keycolor run



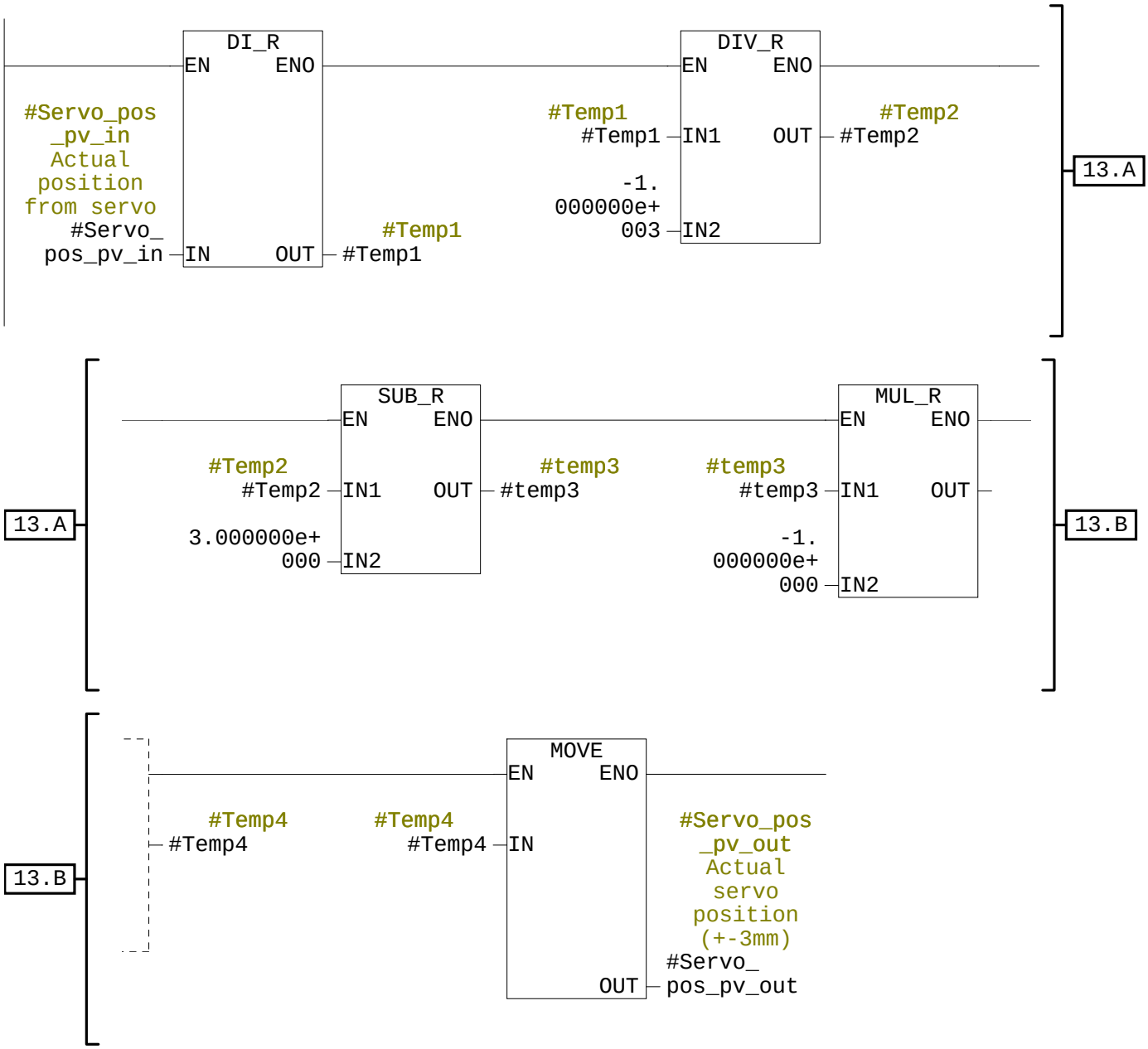
Network: 12 Keycolor run maximum time



Network: 13 Convert servo position (0-6mm) to +/-3mm

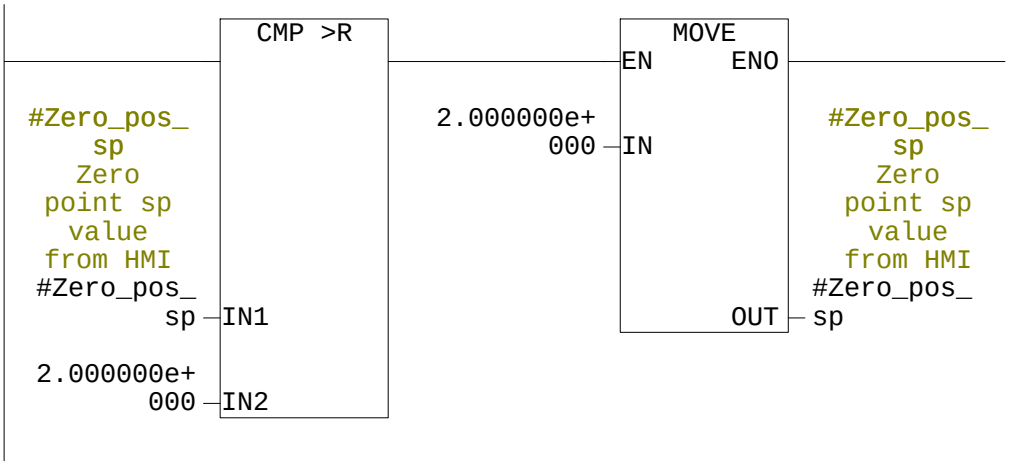
Millim. --> microm. (x1000)

Gear side (ref point) = 0mm
Operator side = 6mm
0mm --> +3.0mm
6mm --> -3.00mm

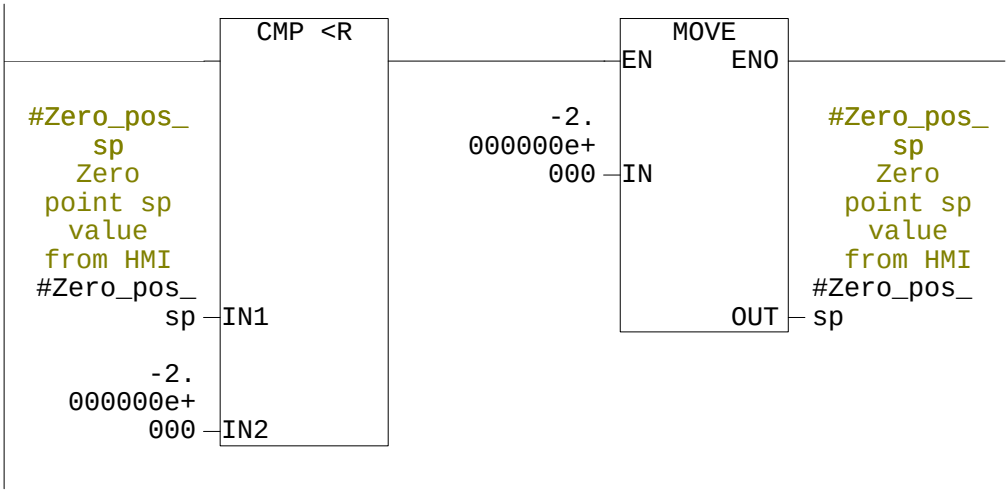


Network: 14 Zero point SP is within upper limit

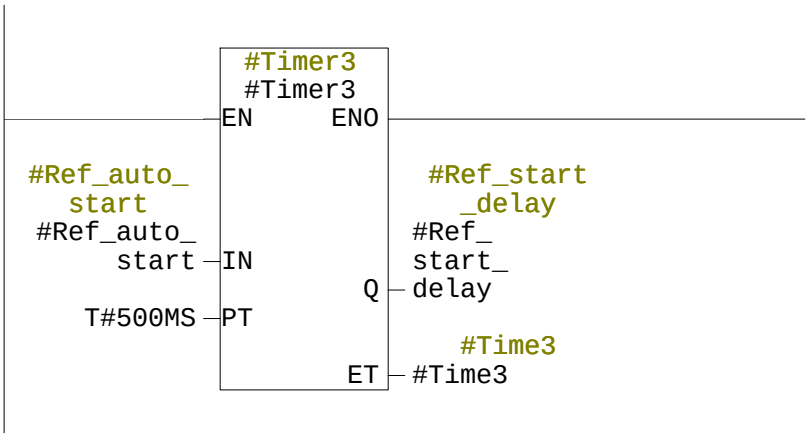
Liite 5
14(27)



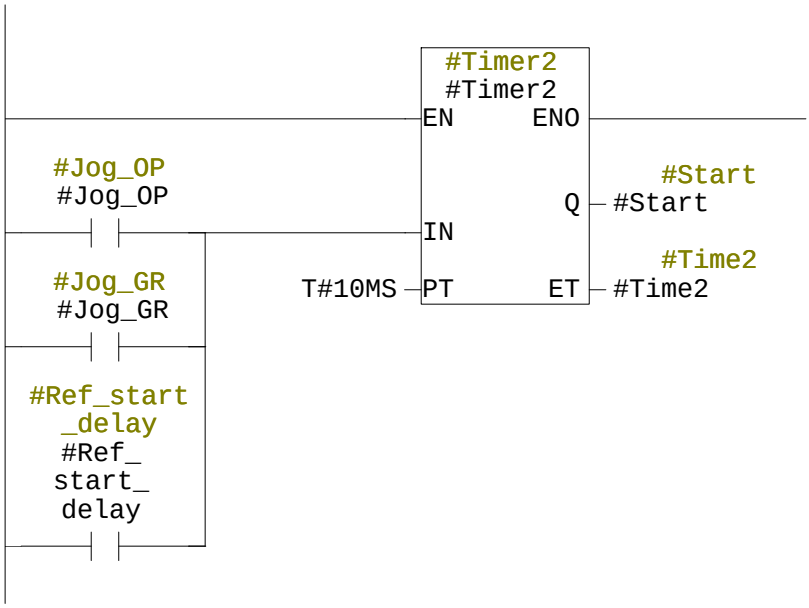
Network: 15 Zero point SP is within lower limit



Network: 16 Ref start delay



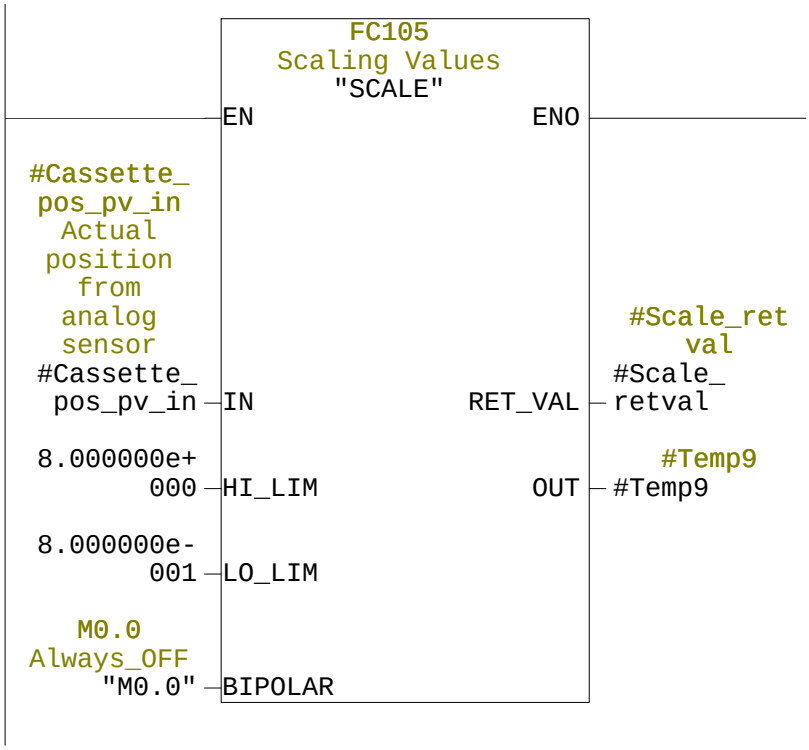
Network: 17 Start (enabled in Jog and Pos -modes))



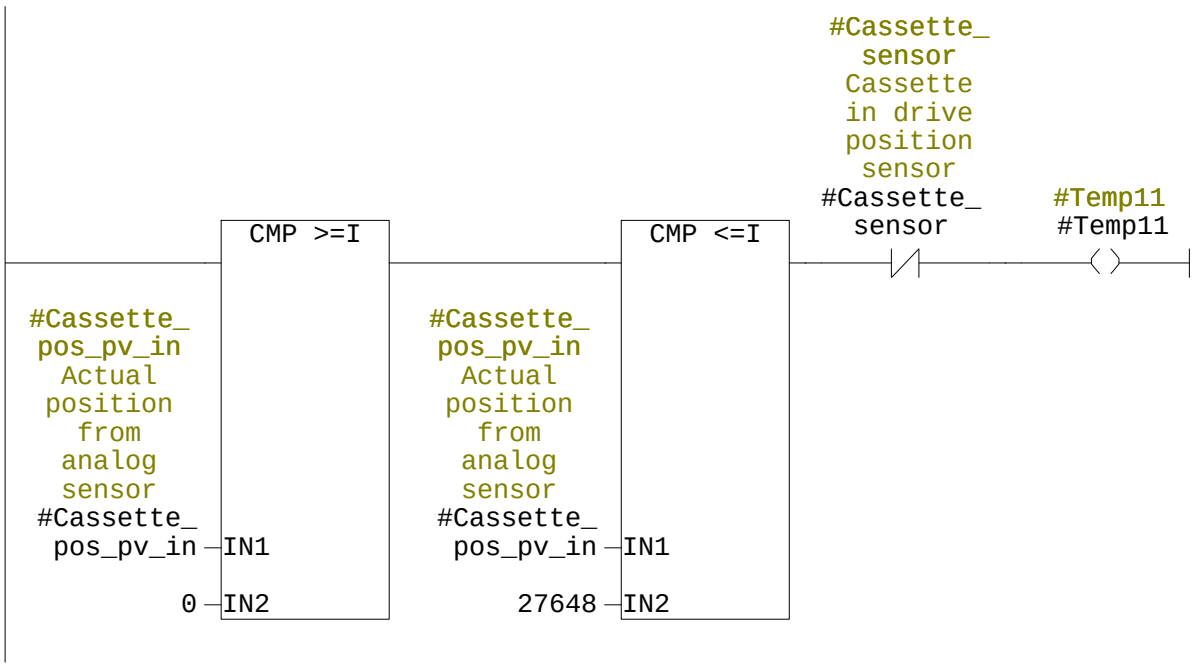
Network: 18 Actual position scale -Analog sensor

Sensor surface = 0mm
Cassette mechanical mid-point = (0)
Cassette distance from sensor:
0,8mm <-----(0)-----> 8mm
0 <-----(0)-----> 27648

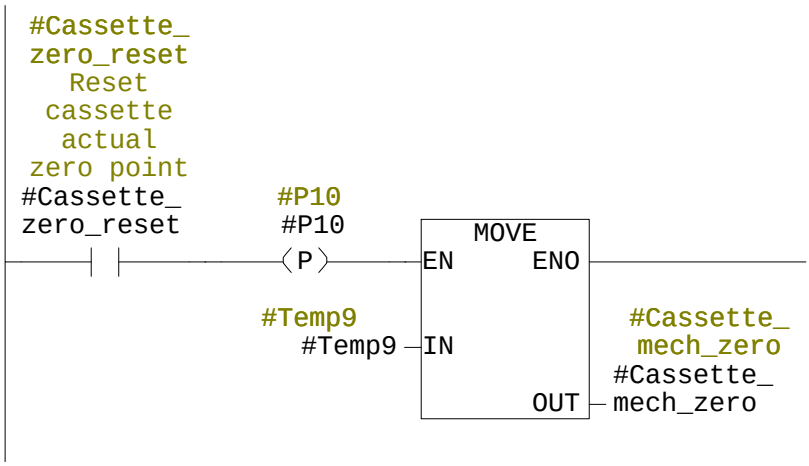
Cassette adjusting range = +/-3mm



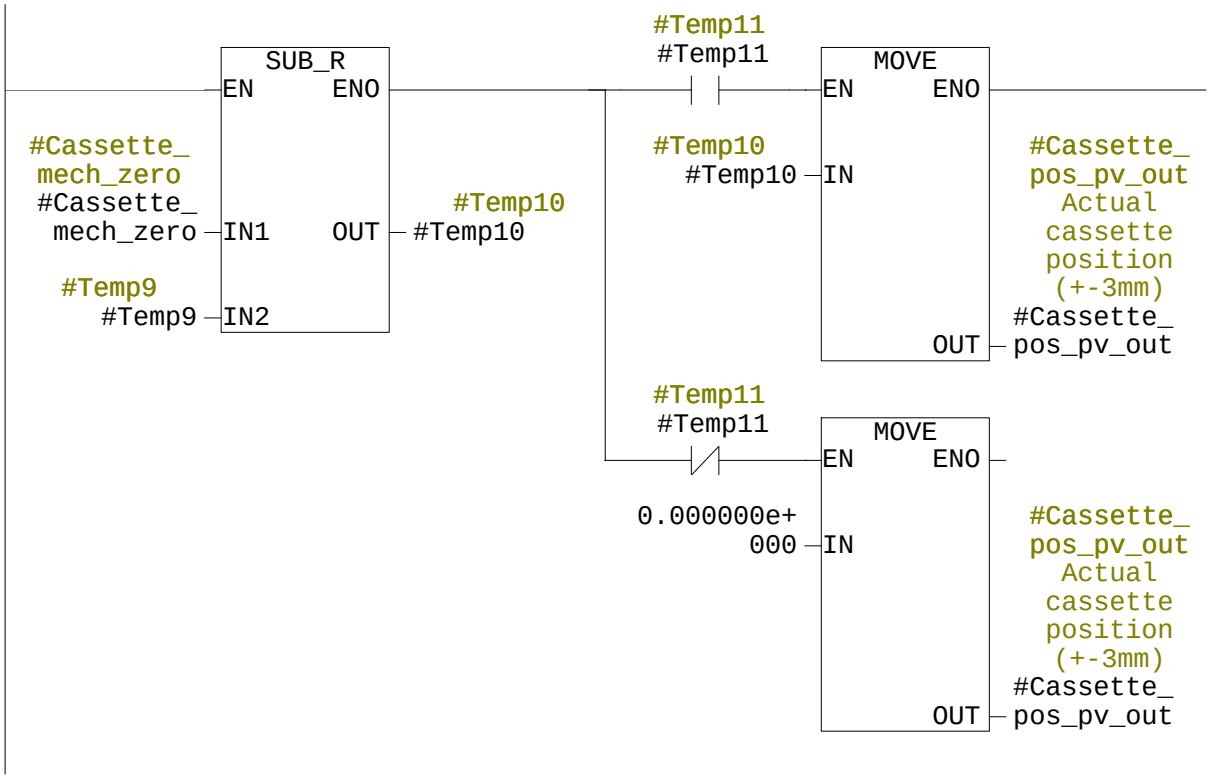
Network: 19 Cassette value within 0-27648 and cassette up



Network: 20 Calibrate cassette mechanical zero



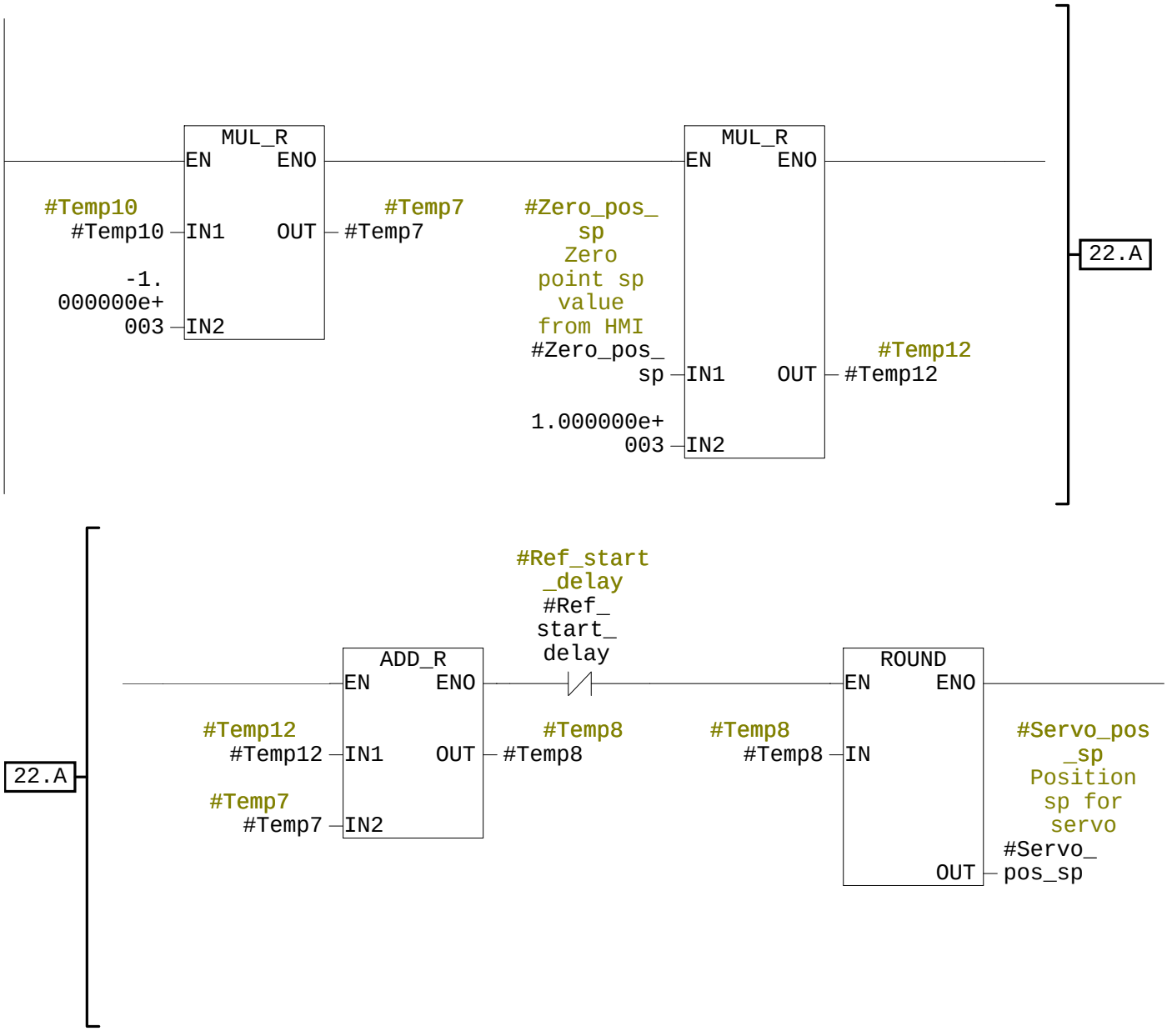
Network: 21 Convert/scale sensor vs cassette actual



Network: 22 Requested zero position

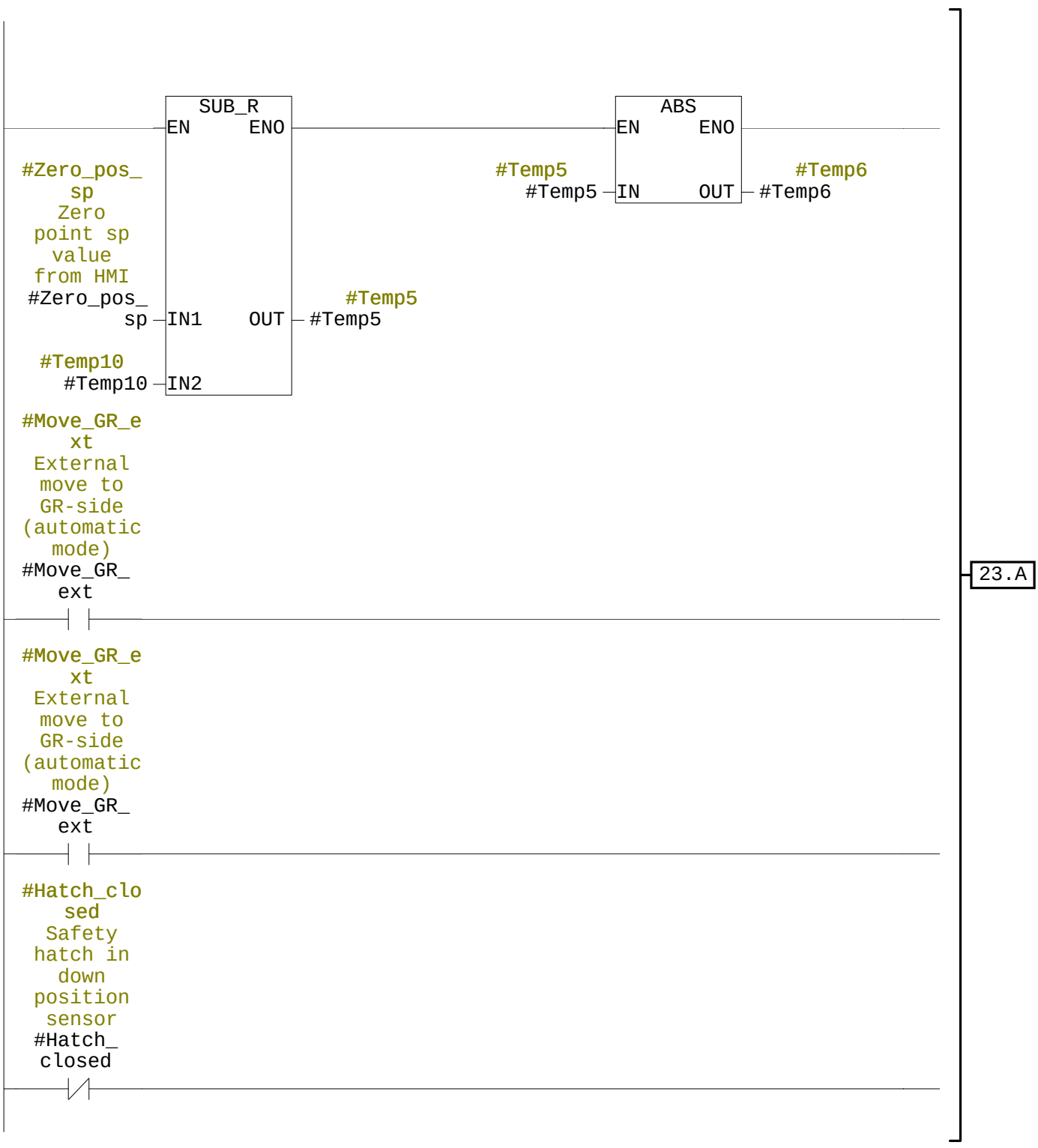
Millim. --> microm. (x1000)

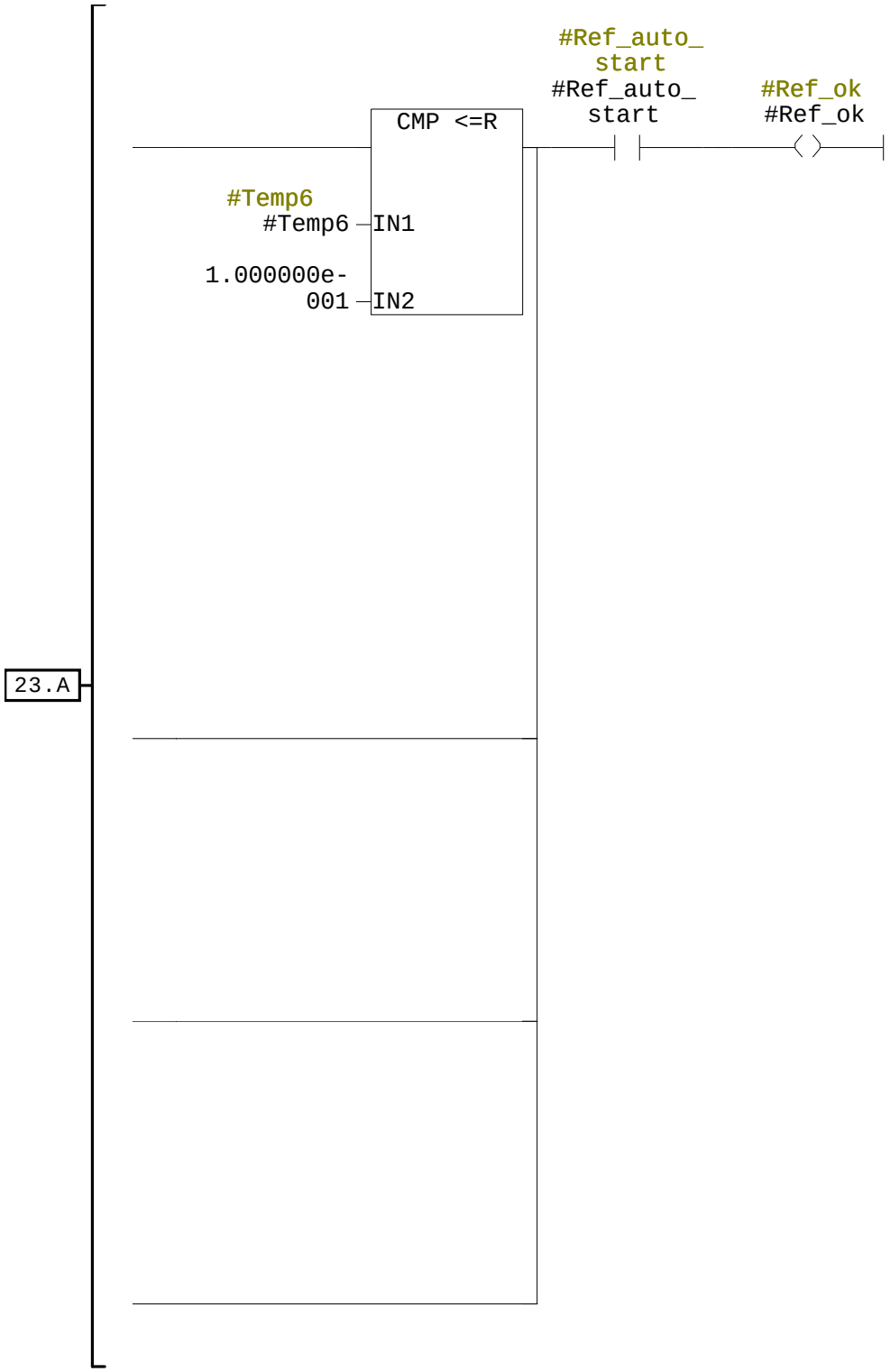
Gear side (ref point) = 0mm
Operator side = 6mm
0mm --> +3.0mm
6mm --> -3.00mm



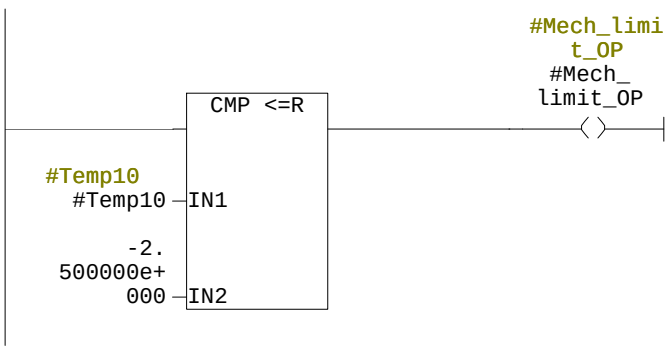
Network: 23 Position is within set range or Registar takes control

Liite 5
19(27)



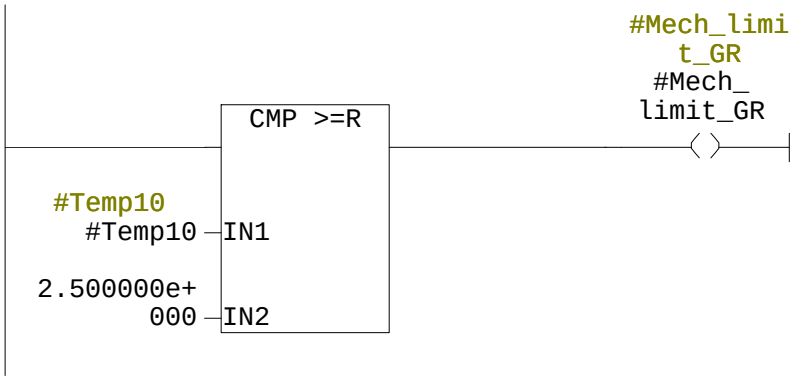


Network: 24 Mechanical limit OP

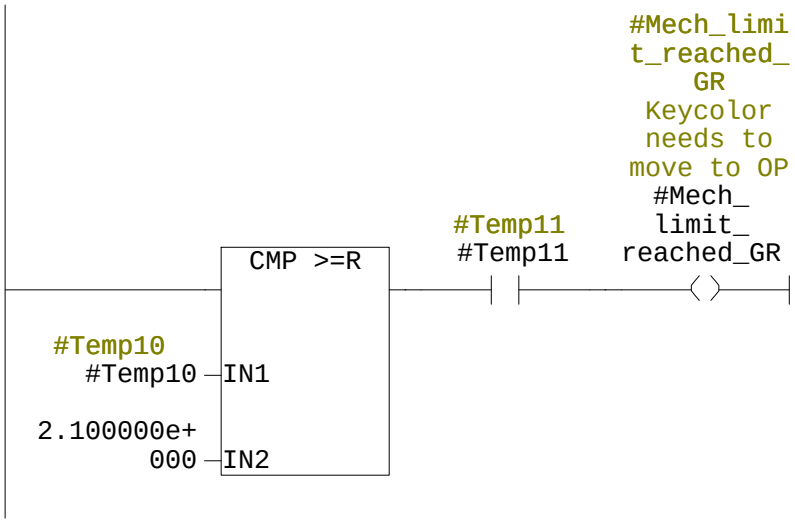


Network: 25 Mechanical limit GR

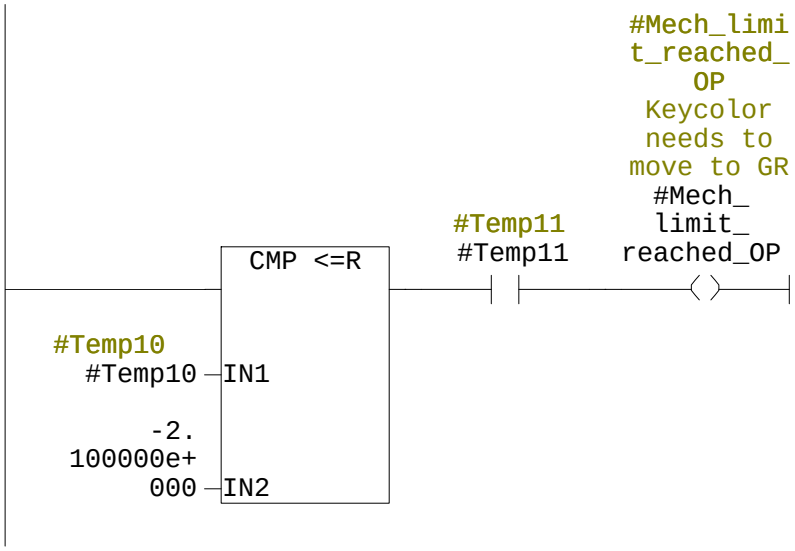
Liite 5
21(27)



Network: 26 Keycolor moving required to OP side



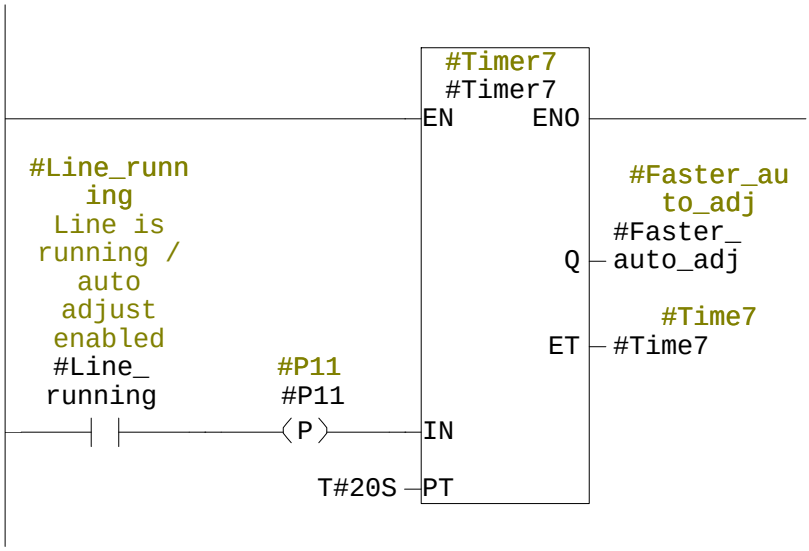
Network: 27 Keycolor moving required to GR side



Liite 5
22(27)

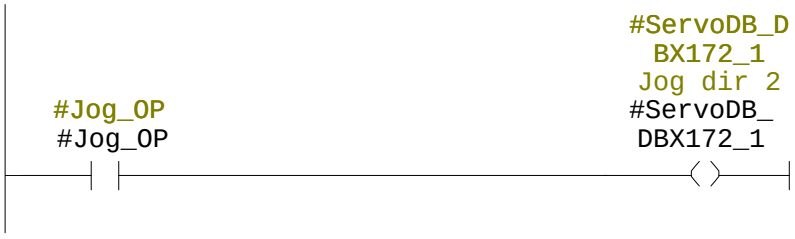
Network: 28 Faster speed after enabled adjusting

Register adjusting can be blocked by main PLC and speeded up after enable.
Need after splice in web to minimize the register variation.

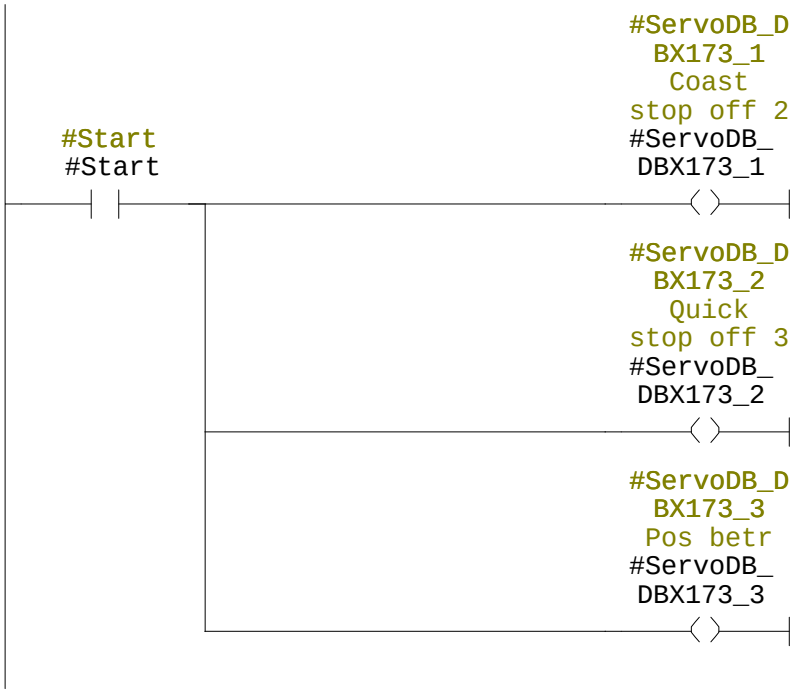


Network: 29 Jog OP

Network: 30 Jog GR

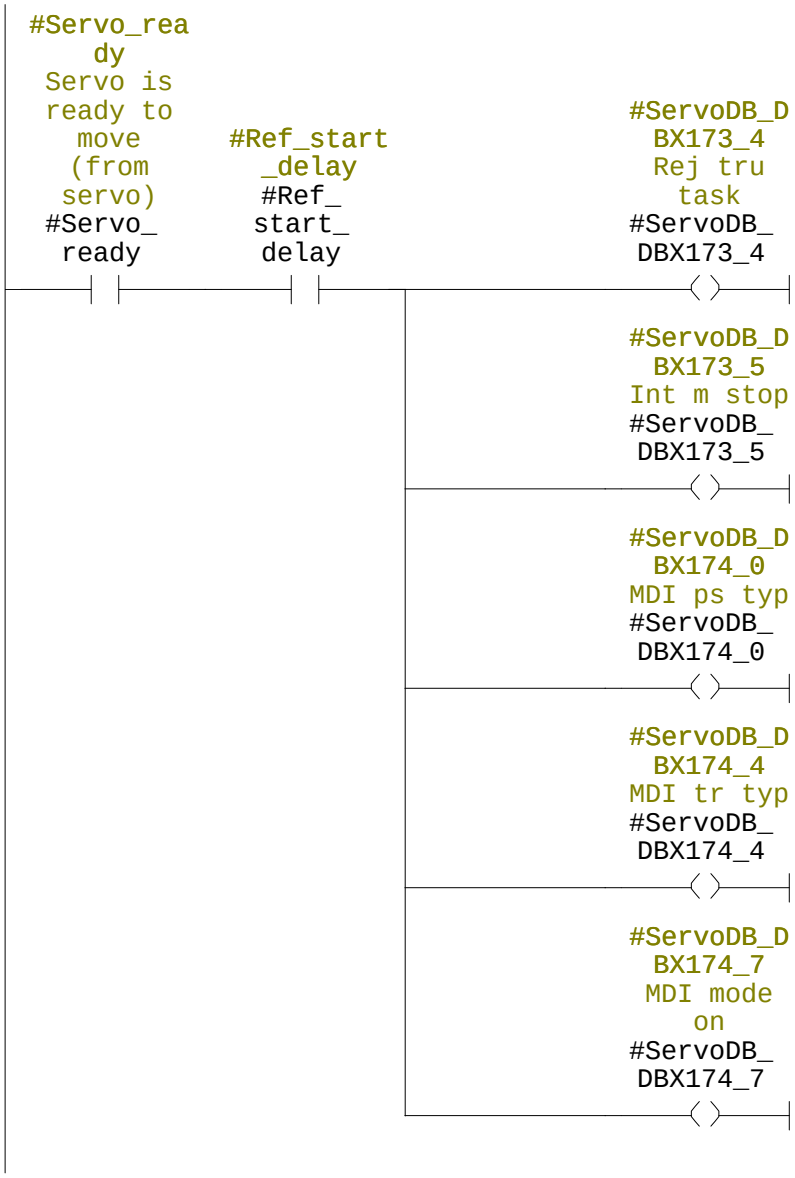


Network: 31 Start command

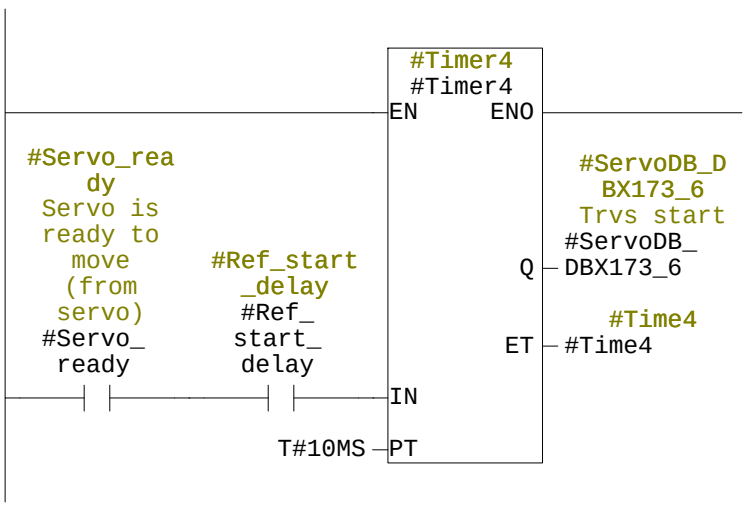


Network: 32 Servo start

Network: 33 Servo start -Pos

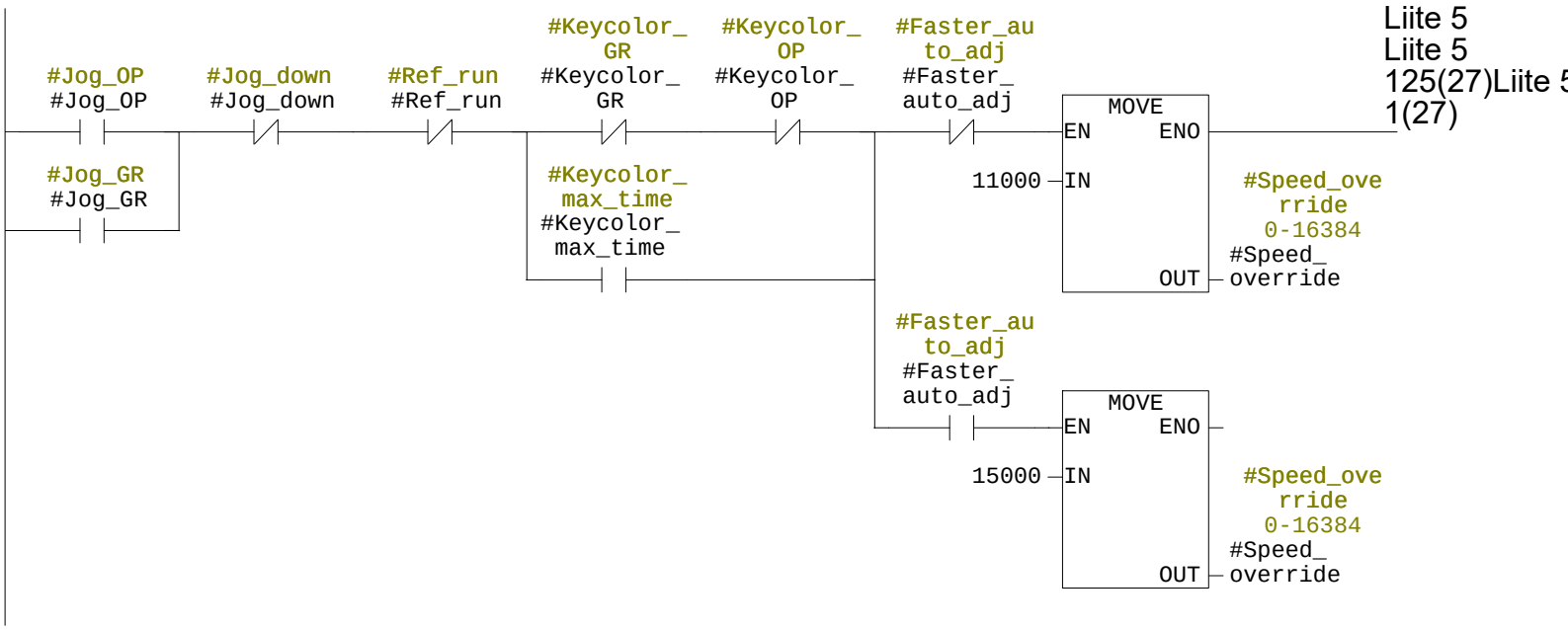


Network: 34 Servo start -Pos



Network: 35 Speed override selection -Jog

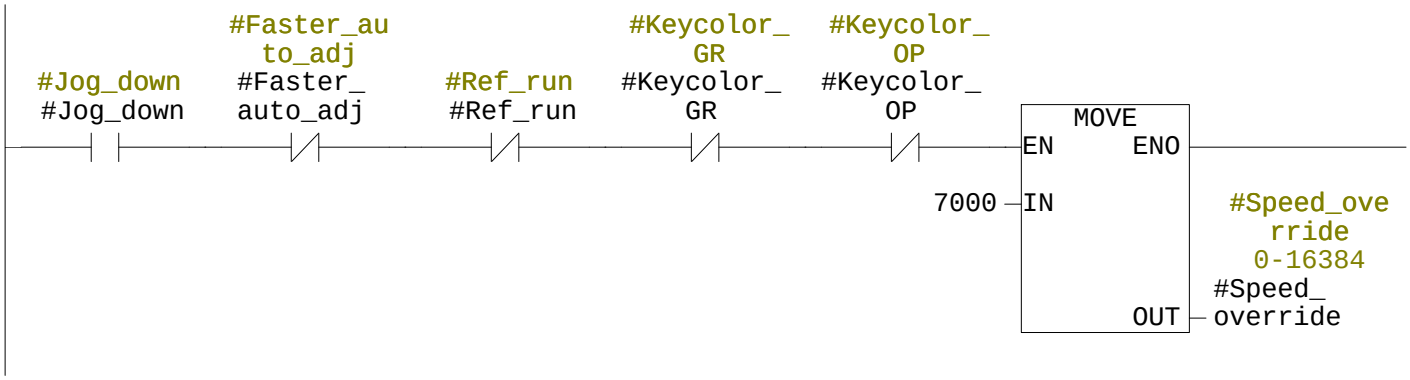
Max 16384



Liite 5
Liite 5
125(27)Liite 5
1(27)

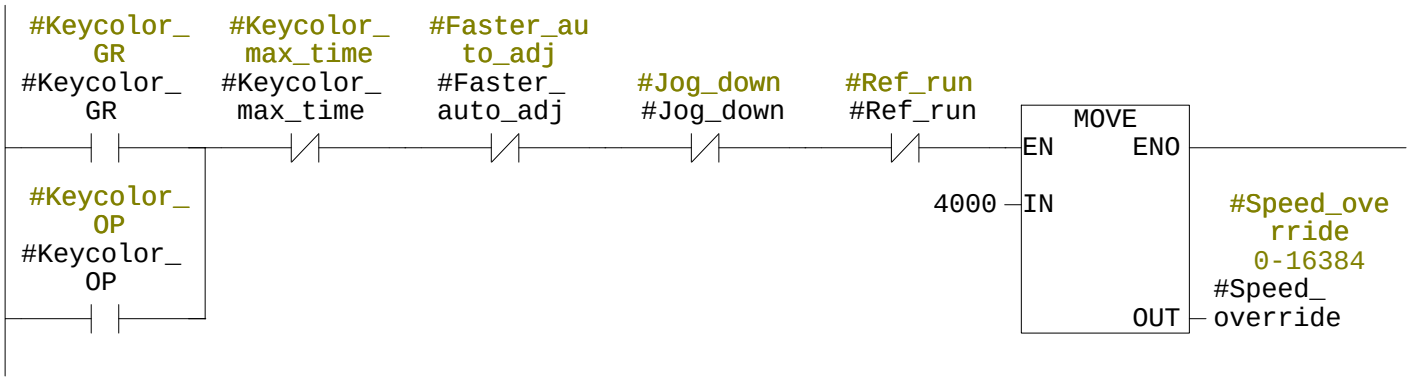
Network: 36 Speed override selection -Jog down

Max 16384



Network: 37 Speed override selection -Keycolor jog

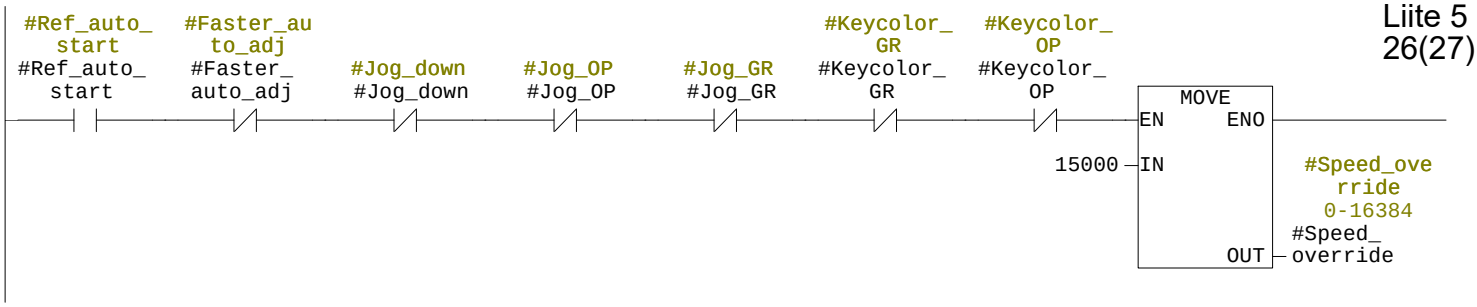
Max 16384



Liite 5
26(27)

Network: 38 Speed override selection -Position

Max 16384



Network: 39 MDI positioning

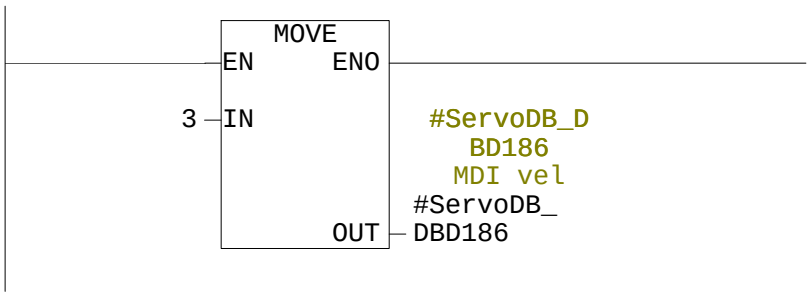
Network: 40 Servo fault

Network: 41 Fault reset

Network: 42 Other faults

Reserved for register control block internal faults.

Network: 43 MDI velocity



Liite 6
1(3)

FB70 - <offline>

"Analog_sensor_ave"

Name: **Family:**
Author: TPi **Version:** 0.1
Block version: 2
Time stamp Code: 03/05/2019 02:11:12 PM
Interface: 10/04/2018 02:58:34 PM
Lengths (block/logic/data): 00328 00202 00008

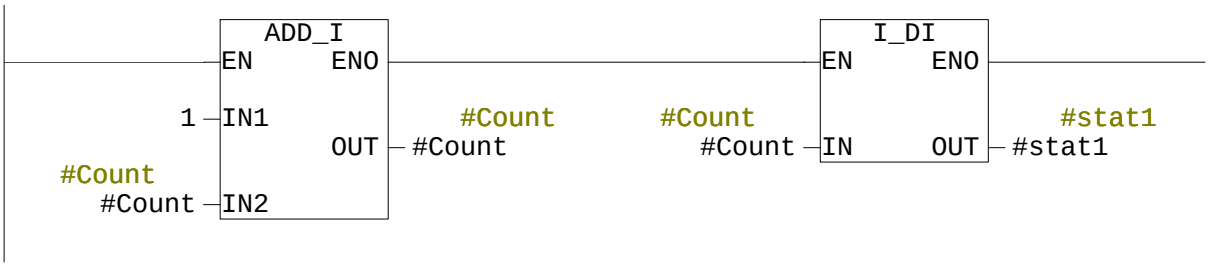
Name	Data Type	Address	Initial Value	Comment
IN		0.0		
Sensor_in	Int	0.0	0	Analog sensor PIW in
Samples	Int	2.0	0	Number of samples
OUT		0.0		
Sensor_ave_out	Int	4.0	0	Averaged sensor value out
IN_OUT		0.0		
STAT		0.0		
Count	Int	6.0	0	
stat1	DInt	8.0	L#0	
stat2	DInt	12.0	L#0	
stat3	DInt	16.0	L#0	
TEMP		0.0		
temp1	DInt	0.0		
temp2	DInt	4.0		

Block: FB70 Analog sensor averaging

Tetra Pak / T. Pinomäki 5.3.2019

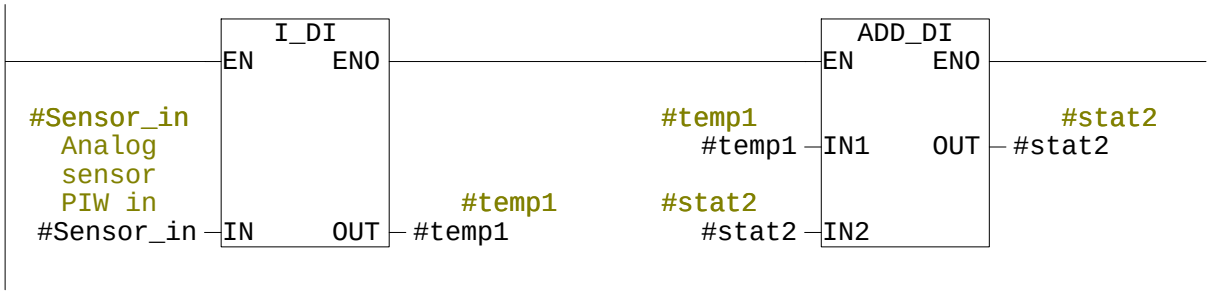
Called in 200ms cycle with adjustable sample rate

Network: 1 Call count

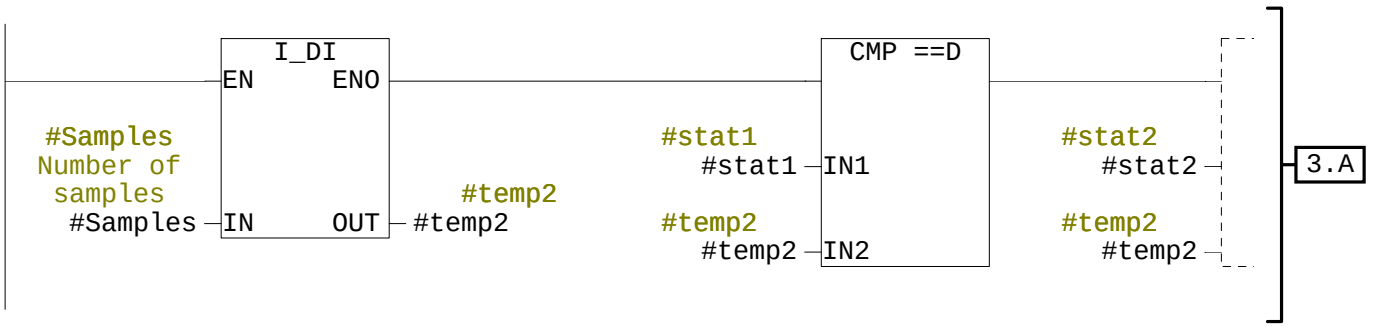


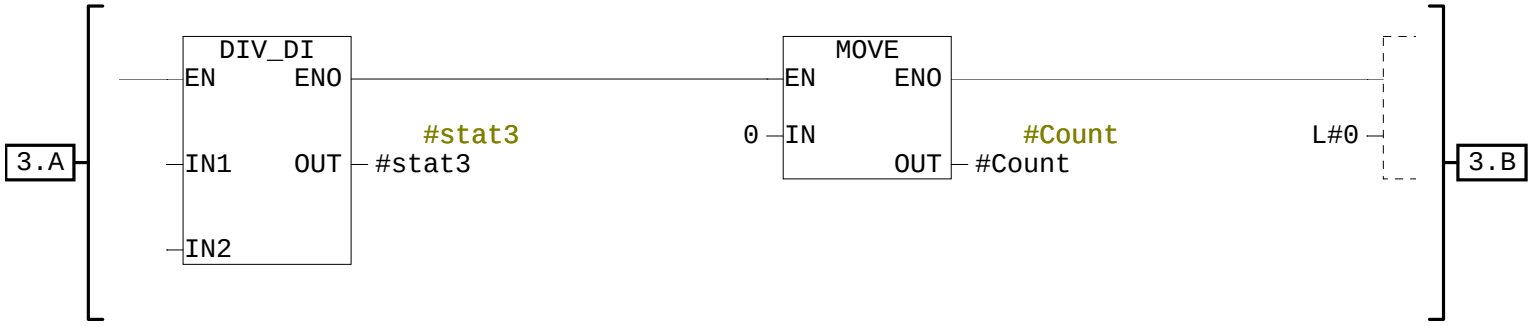
Network: 2 Add values

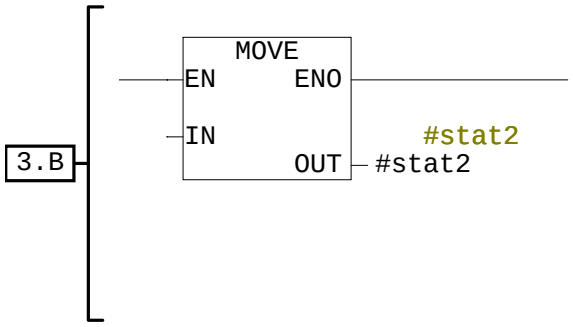
Liite 6
2(3)



Network: 3 Compare sample value with call count and make averaging

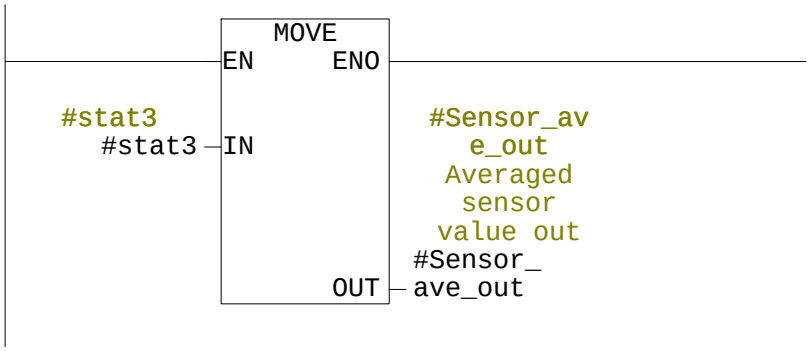






Network: 4 Move average out

Liite 6
3(3)



Liite 7
1(3)

FB80 - <offline>

"Auto_zero_calc"

Name: **Family:**
Author: TPi **Version:** 0.1
Block version: 2
Time stamp Code: 03/05/2019 02:13:51 PM
Interface: 10/08/2018 02:14:33 PM
Lengths (block/logic/data): 00346 00216 00000

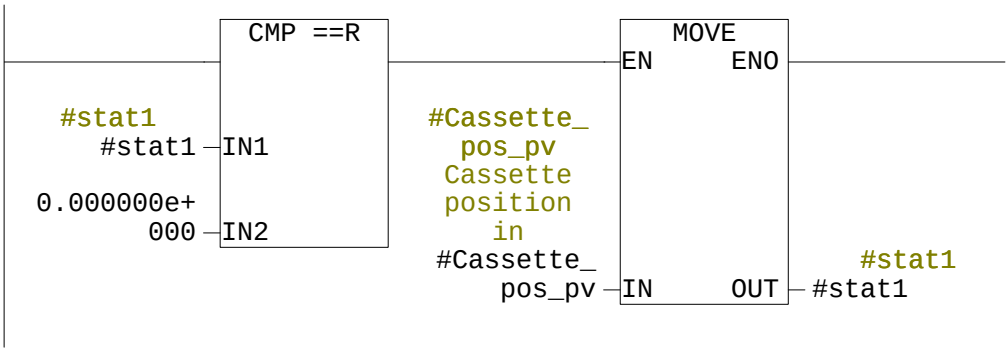
Name	Data Type	Address	Initial Value	Comment
IN		0.0		
Cassette_pos_pv	Real	0.0	0.000000e+000	Cassette position in
Samples	Int	4.0	0	Number of samples
OUT		0.0		
Cassette_zero_pos	Real	6.0	0.000000e+000	Automatic cassette zero pos value
IN_OUT		0.0		
STAT		0.0		
stat1	Real	10.0	0.000000e+000	
stat2	Real	14.0	0.000000e+000	
stat3	Real	18.0	0.000000e+000	
stat4	Int	22.0	0	
stat5	Real	24.0	0.000000e+000	
TEMP		0.0		

Block: FB80 Automatic zero calculation

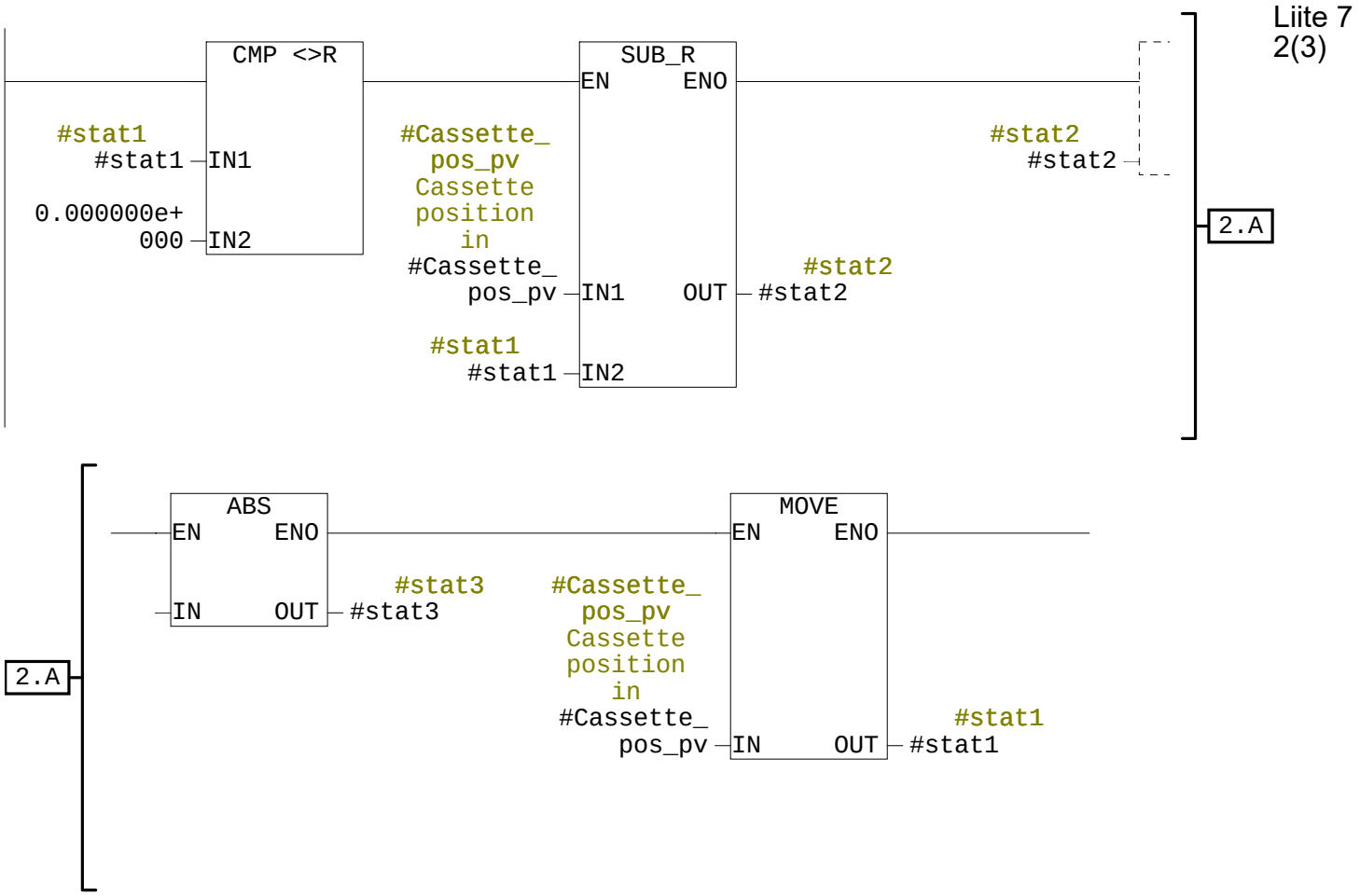
Tetra Pak / T. Pinomäki 14.2.2019

Called in 1000ms cycle with adjustable sample rate

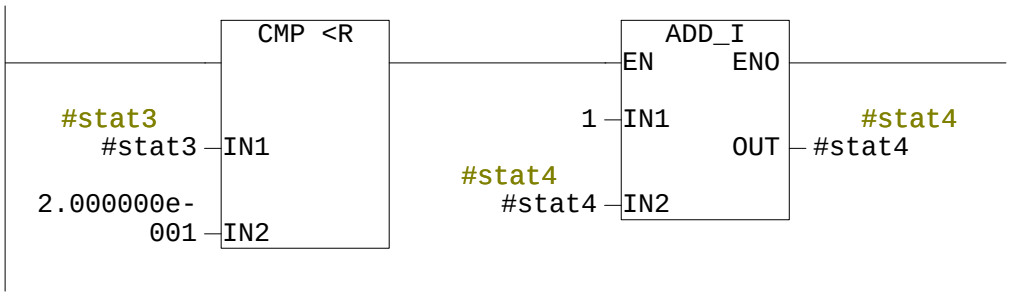
Network: 1 If last value is 0, then save current



Network: 2 Subtract last from current and take absolute value

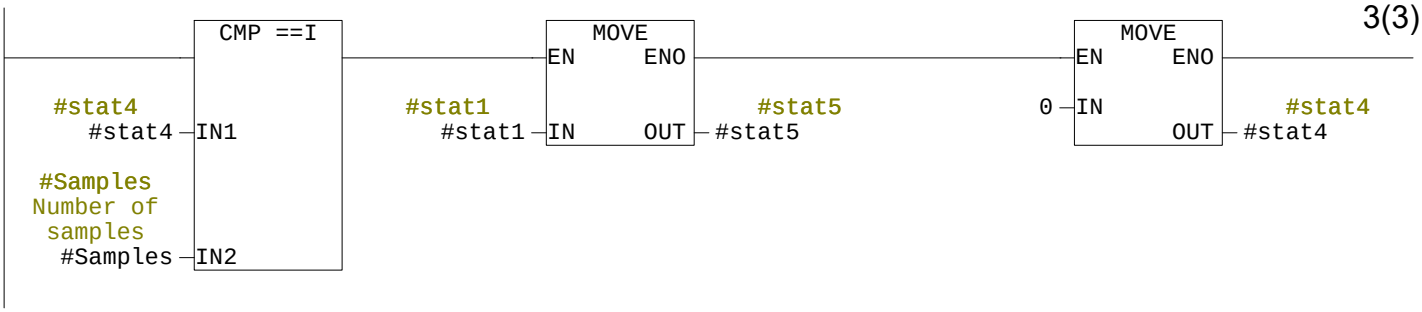


Network: 3 If absolute is less than 0.2, then increase sample counter



Network: 4 When sample counter reach "number of samples", then move last

Liite 7
3(3)



Network: 5 Move out