Annika Lahti

UV-ALD-laitteen ohjelmistosuunnittelu

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Automaatiotekniikka

Insinöörityö

4.5.2017

Metropolia

Tekijä(t) Otsikko Sivumäärä Aika	Annika Lahti UV-ALD-laitteen ohjelmistosuunnittelu 36 sivua + 1 liitettä 4.5.2017
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Automaatiotekniikka
Ohjaaja(t)	Automaatiotiiminvetäjä Matti Välikylä Lehtori Jari Savolainen

Tämän insinöörityön tavoitteena oli osana kehitysprojektia suunnitella ohjelma UV-ALD-laitteelle. Työssä myös suunniteltiin projektin automaatiojärjestelmä. Työ tehtiin suomalaiselle ALD-laitteiden toimittajalle Picosun Oy:lle.

Ohjelmointi toteutettiin TwinCAT 3 -ohjelmointiympäristössä ja logiikkana toimi Beckhoffin Embedded PC CX2020, eli sulautettu PC CX2020. Tarkoitus oli tutustua käytettyyn ohjelmointiympäristöön sekä logiikkaan, ja pohtia millaista lisäarvoa näillä voitaisiin saada verrattuna nykyiseen järjestelmään. Lisäksi tutustutiin EtherCAT-kenttäväylään ja EtherCAT-laitteiden ohjelmointiin. Ohjelmointikielenä tässä työssä käytettiin IEC61131-3-standardin mukaista kieltä ST:tä.

Työn tuloksena saatiin ohjelma tehtyä vaiheeseen, jossa UV-ALD-laitteen käyttöönotto voitiin aloittaa. Projektin myötä saatiin kokemusta TwinCAT 3:sta, sekä ST-ohjelmointikielestä. Jos pohditaan millaista lisäarvoa tulevaisuudessa projektissa käytetyllä järjestelmällä voisi yritykselle olla, erityisesti voidaan manita EtherCAT-väylään liitettyjen laitteiden helppo ja nopea ohjelmoitavuus sekä kommunikointi.

Avainsanat	ALD, TwinCAT 3, Sulautettu PC, EtherCAT

Author(s) Title Number of Pages Date	Annika Lahti Software Development of UV-ALD Tool 36 pages + 1 appendix 4 May 2017
Degree	Bachelor of Engineering
Degree Programme	Automation Engineering
Instructor(s)	Matti Välikylä, Automation Team Leader Jari Savolainen, Senior Lecturer

The aim of this graduate study was to design and develop software for UV-ALD-tool as a part of a R&D project. The study also included designing the automation system of the project. The study was carried out for Finnish ALD manufacturer Picosun Oy.

The programming was done with TwinCAT 3 software and the hardware used in this project was Beckhoff's Embedded PC CX2020. The system also included EtherCAT, which is an Ethernet-based fieldbus.

The secondary goal of this study was to investigate the capability of the used system, and to estimate the possible added value for the company compared to the current system. The programming language used was ST, which is a programming language supported by the IEC 61131-3 standard.

As a result of the graduate study the software was at the point where the commissioning of the UV-ALD-tool could be started. As concerns the added value of the system used for the company, the easy connectivity and programming of EtherCAT devices decreases the time of commissioning new devices.

Keywords	ALD, TwinCAT 3, Embedded PC, EtherCAT

Sisällys

Lyhenteet

ALD *Atomic Layer Deposition* eli atomikerroskasvatus, on pinnoitustapa, jossa luodaan nanometritason kalvoja.

IEC 61131-3 Standardi, joka käsittelee logiikkaohjelmointia ja siinä käytettäviä kieliä.

ST *Structured Text*, standardin IEC 61131-3 mukainen ohjelmoitavan logiikan ohjelmointikieli.

OOP *Object-Oriented Programming* eli Olio-ohjelmointi, on ohjelmoinnin ajatusmalli. Olio-ohjelmointi keskittyy olioihin, jotka ovat tietorakenteita, jotka sisältävät tietoa ja toiminnallisuutta.

1 Johdanto

Tämän opinnäytetyö tehdään Picosun Oy:lle, joka on maailmanlaajuisesti johtava ALD-laitteiden ja ratkaisujen kehittäjä sekä tuottaja. ALD (*Atomic Layer Deposition*) eli atomikerroskasvatus tarkoittaa pinnoitusmenetelmää, jolla saadaan kasvatettua erittäin korkeatasoisia ohutkalvopinnoitteita. Opinnäytetyön tarkoituksena on osana kehitysprojektia suunnitella ohjelma UV-ALD-laitteelle. Ohjelmistosuunnitteluun kuuluu logiikkaojelman suunnittelu TwinCAT 3 -ohjelmalla. Koska kehitysprojekti poikkeaa melko paljon yrityksen normaaleista projekteista, nähdään tämä myös mahdollisuutena kehittää ohjelmistopuolta. Näin ollen osittain tarve työlle liittyy vaihtoehtoisen logiikkavalmistajan logiikkaan ja ohjelmointiympäristöön tutustumiseen sekä niiden uusien ominaisuuksien testaamiseen ja soveltamiseen.

Laite tulee Picosunin henkilökunnan käyttöön. Samalla kun projektia tehdään, tutustutaan uuteen logiikkaan ja sen ohjelmointiin, joka ei vielä ole käytössä yrityksen tuotannossa, ja voidaan pohtia sen kyvykkyyttä. Tavoitteena siis opinnäytetyössä on saada kehitysprojektia varten toimiva laite sekä näkemystä ja kokemusta yritykselle vaihtoehtoisesta logiikkavalmistajasta.

1.1 Picosun Oy

Picosun valmistaa ALD-laitteita teollisuuden tuotantoon sekä kehitys- ja tutkimustyöhön. Yrityksen pääkonttori sijaitsee Espoossa ja tuotanto Kirkkonummella, jossa kaikki Picosunin ALD-laitteet valmistetaan. Viennin osuus tuotannosta on yli 95 %. Picosunilla on viisi tytäryhtiötä eri puolilla maailmaa: Pohjois-Amerikassa, Singaporessa, Taiwanissa, Kiinassa ja Japanissa. Yrityksen liikevaihto oli vuonna 2015 oli noin 10 miljoonaa. [1; 2.]

Picosun työllistää Suomessa hieman yli 50 henkilöä, ja maailman laajuisesti noin 100 henkilöä. Vuonna 2015 uusien tilauksien arvo nousi 16 miljoonaan euroon, jolloin myös uusien tilausten arvo kasvoi 41 %. Asiakkaita ja yhteistyökumppaneita yrityksellä on maailmanlaajuisesti, mukaan lukien teollisuudenalan yrityksiä ja huippuyliopistoja. [2.]

2 ALD

2.1 ALD-teknologia

ALD (*Atomic layer deposition)* eli atomikerroskasvatus on kehittynyt pinnoitustapa, jolla saadaan luotua todella ohuita, yhtenäisiä ja tasaisen aukottomia kalvoja. Atomikerroskasvatus perustuu kemiallisiin reaktioihin kaasujen faasien välillä, jolloin saavutetaan kontrolloitu kalvon muodostuminen jopa tarkemmin kuin nanometritasolla. Pinnoitettava pinta kyllästetään lähdeaineella, jolloin pinta reagoi kemiallisesti lähdeaineen kanssa. Ylimääräinen lähdeaine tai reaktiossa syntyneet sivutuotteet huuhdellaan pois reaktorista, jonka jälkeen voidaan annostella seuraava kerros samaa tai eri lähdeainetta. Pinnoitukseen käytettävät kaasut eivät pääse reagoimaan ennen kuin ne ovat pinnan päällä, ja näin ollen atomeista tai molekyyleistä syntyy kerroksia pinnoitettavaan pintaan. Atomikerroksista muodostuu halutun mukainen kalvo pinnan päälle. Alla kuvassa ALD-laite ja teknologialla pinnoitettuja piikiekkoja. [3; 4.]

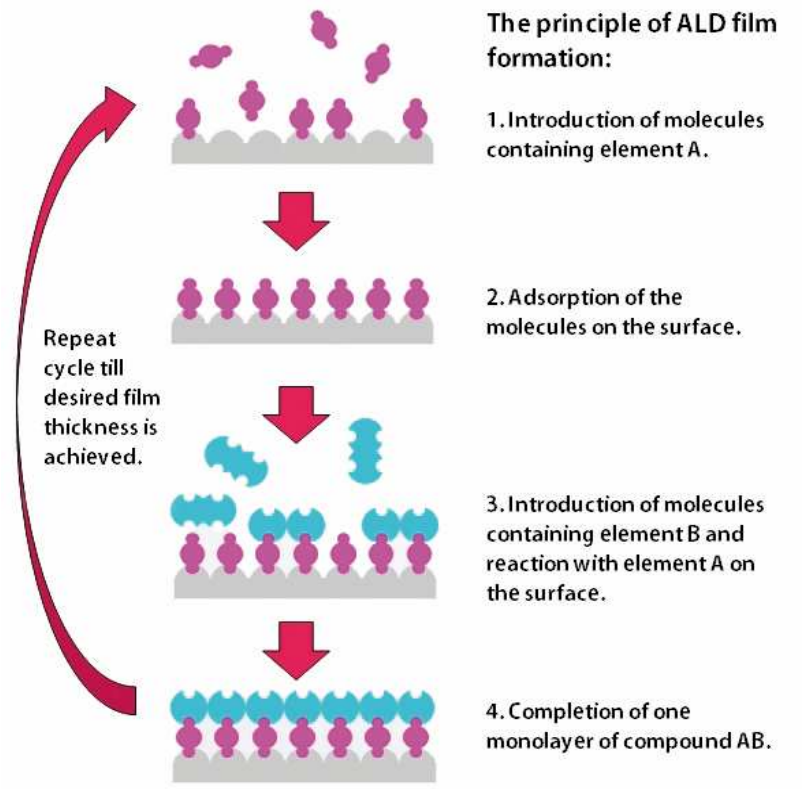
Kuva 1. ALD-laite PICOSUN™ P-300 ALD. [5.]

ALD-teknologian kasvua kiihdyttävät sen ominaisuudet, jotka voittavat kaikki tunnetuimmat pinnoitustekniikat. Tekniikalla voidaan pinnoittaa pienemmissä lämpötiloissa, prosessia voidaan hallita yhden atomikerroksen tarkkuudella sekä pinnoitettua saadaan laajoja ja kaiken muotoisia pintoja. Lisäksi teknologian avulla

voidaan tutkia pinnoitteen kasvatukseen liittyviä kemiallisia reaktioita tarkemmin kuin aikaisemmin. Erityisesti puolijohdeteollisuudessa ALD:n käyttö lisääntyy tulevaisuudessa, pinnoitettavien rakenteiden geometrian monimutkaistuessa. [6.]

2.2 Toimintamekanismi

Kalvon muodostuminen perustuu atomien tai molekyylien adsorptioon. Adsorptio tarkoittaa atomien, ionien tai molekyylien vetovoimaa ja tarttumista johonkin pintaan, niin että muodostuu kalvo. Adsorptio on seurausta kappaleen pintaenergiasta. ALD-prosessissa pinnan atomit tekevät kemiallisia sidoksia ensimmäisen ALD-lähdeaineen kanssa, ja aina seuraava lähdeaineen atomit tai molekyylit tekevät sidokset edellisen kanssa. Adsorptiota ei tule sekoittaa absorptioon, jossa jokin aine imeytyy toiseen aineeseen, toisin kuin adsorptiossa aine kiinnittyy suoraan pintaan. [6.]



Kuva 2. Atomikerroskasvatuksen periaate [3.]

Perusperiaate, jotta ALD-prosessi onnistuu, on että sidosenergia pinnan ja ensimmäisen atomi- tai molekyylikerroksen välillä tulee olla vahvempi kuin seuraavien päälle

muodostuvien kerrosten sidosenergia. Lämpötilansäätelyn avulla voidaan auttaa kerrosta pysymään pinnan päällä, ja samalla sallimaan uusien kerrosten lisäämisen. [6.]

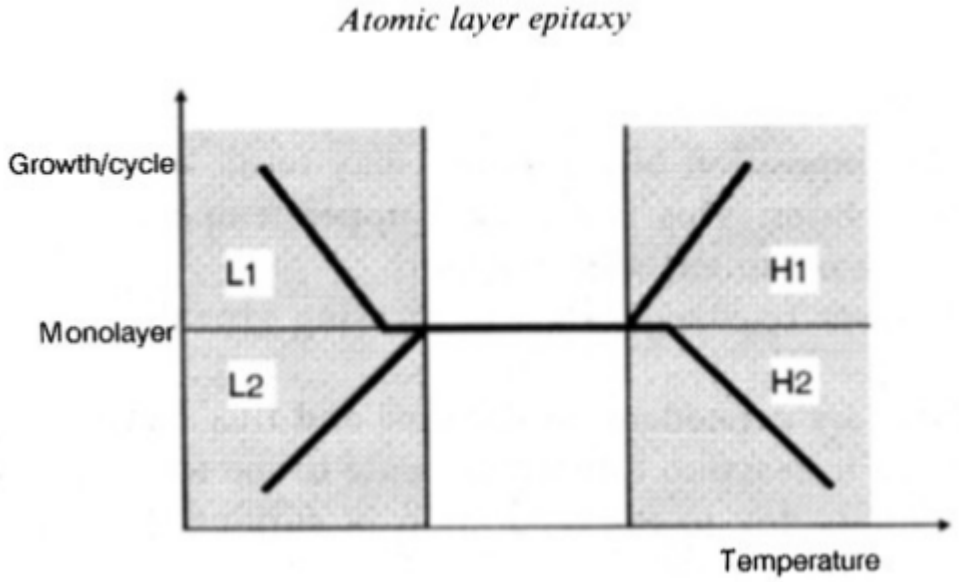
2.3 Prosessi

ALD:n prosessissa toistuu käytännössä sama vaihe jaksotettuna aina uudelleen, jossa pintaan lisätään haluttua materiaalia kerroksen verran. Kuten myös kuvassa 2 havainnollistetaan, toistettava vaihe eli reaktiosykli voidaan jakaa neljään osaan:

- Vaihe 1: Tuodaan lähteestä reaktiokammioon reaktanssia, joka reagoi pinnoitettavalla pinnalla.
- Vaihe 2: Kammio huuhdellaan, jotta päästään eroon reagoimattomista lähdeaineista tai kaasumaisista sivutuotteista.
- Vaihe 3: Seuraavaksi kammioon voidaan uusi reaktanssi joka reagoi ensimmäisen reaktanssin kanssa, tai voidaan lisätä uusi kerros ensimmäistä reaktanssia.
- Vaihe 4: Lähdeaineen lisäämisen jälkeen huuhdellaan kammio. [7, s. 3.]

Reaktiosykliä toistetaan niin kauan, kunnes saavutetaan haluttu kalvon paksuus. Koska pinnoittaminen tehdään molekyylikerros kerrallaan, voidaan kalvon paksuutta helposti hallita säätämällä pinnoitussyklien määrää. Kalvon kasvamista kuvataan yksiköllä GPC (growth per cycle) eli kasvulla per sykli, joka tyypillisesti ALD-prosessissa 0,05–0,1 nm. Syklin kestoon vaikuttavat lähteiden pulssitusajat ja huuhteluajat. [4; 8.]

Jotta reaktanssi tarttuu pintaan oikein, pitää pinnan olla hyvin tunnetussa ja hallitussa tilassa, siksi pinta stabilisoidaan ennen prosessin alkua. Stabilisointi voidaan toteuttaa esimerkiksi kuumentamalla. Lämpötilansäätely on tärkeä osa prosessia ja sen onnistumista. Liian kuumassa voivat atomit tai molekyylit karata pinnasta, mutta liian kylmässä uudet kerrokset eivät tartu. Reaktorin kuumennus tehdään ennen prosessin aloittamista. Seuraavassa kuvassa prosessin lämpötilataulukko, jonka keskikohta kuvaa sopivaa lämpötilaa, ja sen ulkopuolella on kuvattu liian kuuman tai kylmän lämpötilan vaikutuksia pinnoituksessa. [4; 6.]



Kuva 3. Prosessin lämpötilataulukko. L1 kuvastaa reaktanssin tiivistymistä, L2 reaktioiden aktivointienergian vähyyttä, H1 reaktanssin hajoamista ja H2 ensimmäisen kerroksen uudelleen haihtumista [6.]

ALD-tekniikan kalvonmuodostamistavan ansiosta saadaan tiheä, naarmuton, kulumaton ja tasainen pinnoite, jonka paksuutta ja ominaisuuksia voidaan muunnella. Eri materiaaleja voidaan sekoittaa ja kerrostaa, jonka vuoksi ALD-pinnoitukselle on useita sovelluskohteita. [3.]

ALD-teknologian rajoittavia puolia teollisuudessa on sen hitaus kerroksien kasvattamisessa (100–300 nm/h) ja käytännöllisten lähdeaineiden puute. Rajoitteita on pyritty vähentämään batch-prosesseilla sekä lähdeaineiden kehityksellä. Batch-prosessilla tarkoitetaan prosessia, jossa pinnoitetaan tuotantoerä, yhden kappaleen sijaan.

2.4 Lähdeaineet

Kaasujen faasit muodostuvat tarkasti kontrolloiduista ja pulssittamalla vaiheistetuista lähteistä. Lähteet ovat kaasumuodossa olevia molekyylejä, ja ne pystyvät täyttämään koko päällystettävän pinnan sen geometriasta huolimatta. Koska lähteet eivät prosessissa reagoi kaasumuotoisina, estyvät reaktiot joidenka lopputuloksena syntyy rakeisia kalvoja. Lähdemateriaaleja on useita erilaisia, kuten esimerkiksi oksidit, nitriitit,

fluoridit, karbidit, sulfiitit, erilaiset yhdisteet ja jopa erilaisia metalleja, mukaan lukien arvometallit. Yleisin ALD-lähdeyhdistelmä on trimetyylialumiini (TMA) ja H_2O, joka toimii mallireaktiona tehokkuutensa vuoksi. [3; 9, s.112.]

Tärkeimpiä vaatimuksia lähdeaineille ALD-tekniikassa ovat, että tiedetään niiden olevan stabiileja tietyissä lämpötiloissa, että ne kaasuuntuvat herkästi ja että ne ovat todella reaktiokykyisiä. Lisäksi toivottavia ominaisuuksia ovat esimerkiksi aineen puhtaus, halpa hinta, helppo käsiteltävyys, kaasumaiset ja reagoimattomat sivutuotteet sekä myrkyttömyys. Sopivien lähteiden löytäminen ja tutkiminen on osa ALD-tekniikkaan liittyvää jatkuvaa kehitystyötä. [4.]

Lähdeaineet voivat olla olomuodoltaan kaasuja, nestemäisiä tai kiinteitä. Kiinteiden aineiden käyttämisen pinnoittamisessa mahdollistaa lähteen lämmitys, sillä oikeassa paineessa ja lämpötilassa tiettyjä kiinteitä aineita saadaan vaporisoitumaan ja näin ollen ne voidaan kuljettaa reaktiokammioon. Viilennys puolestaan mahdollistaa nestemäisten lähteiden vaporisoinnin, kun lähdeaine on sellaista, että se vaporisoituu oikeassa lämpötilassa ja paineessa. [10.]

Plasmalla voidaan tehostaa ALD-prosessia, ja metodin suosio onkin noussut nopeasti viime vuosien aikana. Plasmalla avustettua ALD:tä kutsutaan PEALD:ksi(Plasma Enhanced ALD). Käytettäessä plasmaa saadaan suurempi valikoima lähteitä, materiaalien ominaisuuksia, pinnan lämpötila vaihtoehtoja ja prosessiolosuhteita. Plasman etuina sen korkea reaktiivisuus, joka on ALD:ssä tärkeää, jopa suhteellisen alhaisissa lämpötiloissa ja on. Haasteena puolestaan tässä prosessiaskeleessa on ollut, että kalvon yhteneväisyys saattaa kärsiä ja plasmavahinkoa syntyä. Ajan myötä on kuitenkin onnistuttu kehittämään plasmalähteitä, niin ettei pinnoitteen laatu kärsi. [8.]

Plasma koostuu partikkeleista, jotka ovat sekoitus ioneja ja elektroneja sekä varattuja ja neutraaleja atomeja. Plasmaa tehdään kiihdyttämällä ja kuumentamalla kaasumaisessa muodossa olevien aineiden elektroneita sähkökentän avulla. Tällöin kaasu ionisoituu ja mahdolliset molekyylien sidokset erottautuvat. Ionisoitumisen johdosta syntyy reaktiivisia neutraaleja atomeja tai molekyyleja, joita kutsutaan plasmaradikaaleiksi, sekä ioneja ja fotoneita. Yleensä ionisointimäärä on pieni ja kaikki ionit sekä elektronit ehtivät kadota plasman kulkeutuessa reaktoriin, joten näin ollen pääosin pinnalla tapahtuvat reaktiot ovat plasmaradikaalien aikaansaamia. [8.]

2.5 Historiaa

ALD:n kehitti suomalainen keksijä Tuomo Suntola 1970-luvulla nimellä ”Atomic Layer Epitaxy” (ALE), ja hän patentoi menetelmän, jossa kasvatetaan yhdisteistä valmistettuja ohuita kalvoja. Menetelmä oli vuosikymmen aikaisemmin löydetty Neuvostoliitossa nimellä ”molecular layering” (ML), mutta molemmat keksinnöt tapahtuivat riippumattomina toisistaan. [11, s.332.]

Tuomo Suntola keksi menetelmän, kehittäessään ohutta paneelinäyttöä, elektroluminenssinäyttöä Instrumentarium Oy:lle. Aikaisemman kokemuksensa ansiosta tekniikan tohtorina Suntola tiesi, että olisi saatava aikaan täydellisen hallittu kidemuoto, jotta saataisiin oikealaiset ominaisuudet kalvolle. Kidemuodon luomiseksi täytyi materiaalin rakentumiseen luoda ympäristö, jossa se voidaan toteuttaa kontrolloidusti. [11, s. 334.]

Katsellessaan seinällä olevaa jaksollista järjestelmää ja pohtiessaan luonnon symmetriaa Suntola sai ajatuksen yhdisteen elementtien yksittäisestä lisäämisestä sekvensseissä. Hän ymmärsi, että yksiatomisten kerrosten luominen onnistuu, jos sidokset elementtien välillä ovat vahvempia kuin aineen omien atomien välillä. Tämän läpimurron myötä Suntola alkoi tekemään laskelmia, saadakseen selville lämpötilan ja paineen, jossa voidaan kasvattaa sinkkisulfaattia. Valonemittointi ominaisuutensa vuoksi sinkkisulfaatti sopi materiaaliksi elektroluminenssinäytöille. [11, s. 334.]

Laskelmien perusteella tehtiin ensimmäinen ALD-kone ja atomikerroskasvatusta päästiin testaamaan loppukesästä 1974. Jo ensimmäinen kokeilukerta onnistui, vaikkakin oletetun yhden molekyylikerroksen sijaan saatiin yksi kolmasosakerrosta kierrosta kohden. Pian ALE-keksintö patentoitiin kansainvälisellä patentilla ja Tuomo Suntola kiersi esittelemässä keksintöään. Suntola siirtyi suunnittelemaan EL-näyttöjä rakennusmateriaaliyritys Lohja Oy:lle, ja vuonna 1983 saatiin ensimmäiset kaupalliset ALE-tekniikalla valmistetut EL-näytöt markkinoille, joita asennettiin esimerkiksi Helsinki-Vantaan kentälle informaatiotauluiksi. Alla kuvassa informaatiotaulujen testiasennus. [11, s. 335–337.]

Kuva 4. ALE-tekniikalla valmistettujen EL-näyttöjen testiasennus Helsinki-Vantaalla vuonna 1983. Kuva on Tuomo Suntolan henkilökohtaisesta kokoelmasta. [11.]

Seuraava askel tapahtui, kun ALE-tekniikkaa alettiin kehittämään aurinkopaneeleita varten, jolloin lopulta päädyttiin siihen, että aurinkopaneeleita voidaan valmistaa halvemmalla muilla keinoilla. Tästä lähtien on ALD-teknologiaa kehitetty pitkälle ja sille löytyy uusia käyttötarkoituksia jatkuvasti. Tuomo Suntola jatkaa edelleen uraansa ja kuuluu esimerkiksi Picosun Oy:n hallitukseen. [11, s.339–342.]

2.6 Käyttökohteita

Puolijohteiden valmistus on ollut suuri syy ALD-teknologian vilkkaaseen kehitykseen viime aikoina. Komponenttien koon pieneneminen on johtanut todella korkeatasoisiin rakenteisiin, jotka on tärkeää pinnoittaa laadukkaasti. Yhteneväinen ja peittävä pinnoite vaatii kalvojen valmistukselta prosessin tarkkaa hallintaa atomitasolla, ja ALD yltää pinnoittamisen vaatimuksissa korkeimmalle. [9, s. 112.]

Kuten Mooren laissa todetaan, komponenttien määrä mikropiireissä kaksinkertaistuu parin vuoden välein. Komponenttien määrän kasvu ja puolestaan laitteiden koon pienentyminen tarkoittaa, että komponenttien koon on pienennyttävä jatkuvasti. Tällöin komponenttien, kuten transistorien, kondensaattorien ja muistien, kalvorakenteiden tulee olla entistä pienempiä. Jo parikymmentä vuotta sitten ymmärrettiin pienentymisen johtavan siihen, että esimerkiksi CMOS-transistoreiden hilojen dielektriset kalvot tulevat ohentumaan niin paljon, että sähkö pääsee vuotamaan rakenteesta läpi. [12.]

Mikropiirien valmistamisen lisäksi teknologiaa hyödynnetään myös muilla suurenluokan teollisuuden aloilla, kuten sensorien, LED:ien ja mikrosysteemien valmistamisessa. ALD:n käyttökohteita löytyy myös optiikasta ja optoelektroniikasta, aurinkoenergian sovelluksista, ekologisista pakkauksista, lääketieteen sovelluksista kuten sydämentahdistimista ja implanteista. Kuvassa 5. voidaan nähdä muutamia näistä esimerkeistä. Teknologia tarjoaa myös paljon ratkaisuja esimerkiksi vedenkestävyyteen sekä korroosion ja naarmuuntumisen estämiseen. Jokapäiväisestä elämästä pinnoitteita löytyy pattereissa, sekä esimerkiksi asusteissa tummumista estämässä tai koristeellisena pinnoitteena. [3; 12.]

Kuva 5. Esimerkkejä ALD-pinnoituksen sovelluskohteista. [12.]

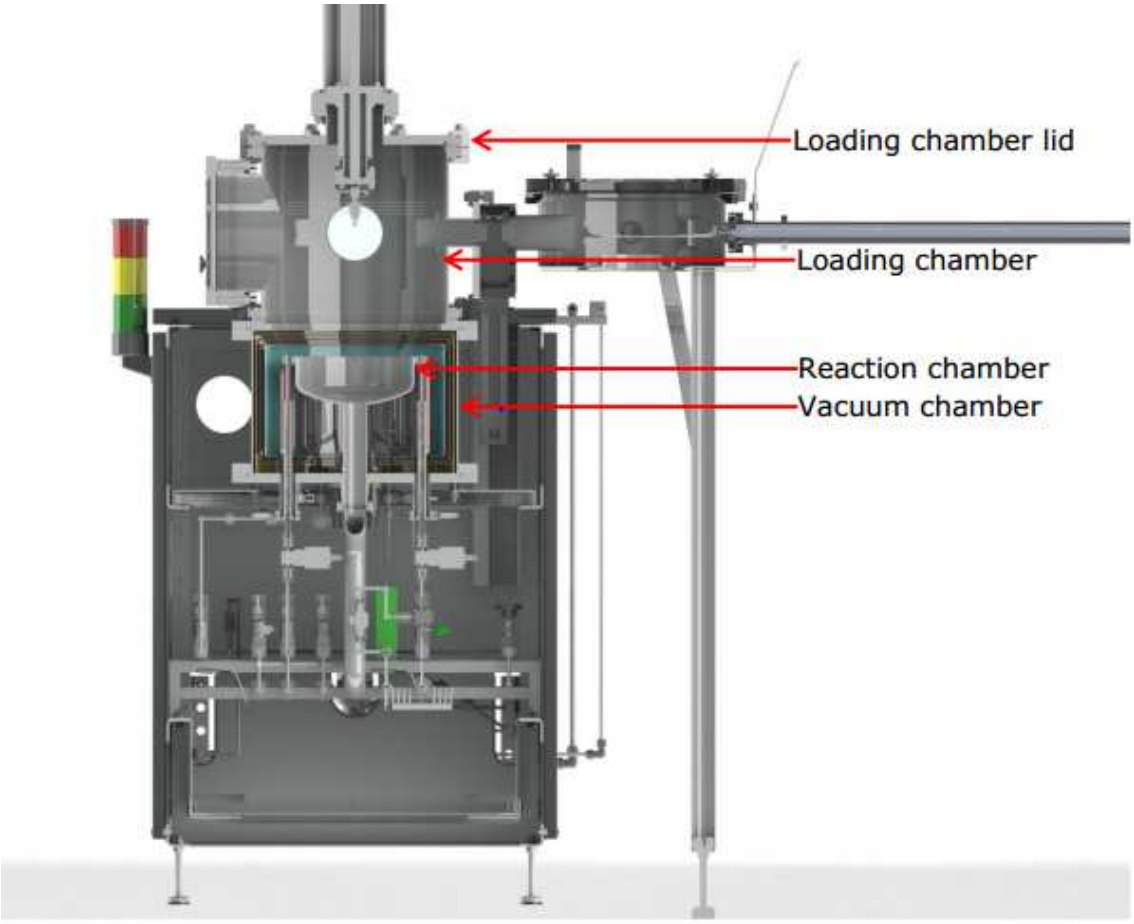
2.7 ALD-reaktori

Picosunin PICOSUN™ ALD-reaktorit rakentuvat samoista perusosista, jonka päälle rakennetaan jokainen yksilöity laite. Perusosiin kuuluu reaktiokammio, kantajakaasujärjestelmä, lähdejärjestelmä, vakuumijärjestelmä sekä nostojärjestelmä. [10.]

Perusosien lisäksi laitteisiin voidaan lisätä robotiikkaa ja laitteita, käyttötarkoituksen mukaan. Esimerkiksi pinnoitettavan kohteen käsittely voidaan robotiikan avulla täysin automatisoida, laitteeseen voidaan lisätä täristin jauheiden pinnoittamista varten sekä tekniikkaa erilaisten muotojen tehokkaampaa pinnoitusta varten. [5.]

2.7.1 Reaktiokammio

Reaktiokammion rakenne on kaksiosainen, joka koostuu ulommasta kammiosta sekä sisemmästä reaktiokammiosta. Näiden kahden välissä olevaa tilaa kutsutaan välitilaksi. Koko kammiolla on yksi kansi, joka sulkee sen koko rakenteen. Kansi toimii pneumaattisilla sylintereillä ja venttiileillä. Reaktiokammion sisäpuolella on pidike pinnoitettavalle kohteelle, joka on kiinni kammion kannessa. Kannen laskeutuessa pidike myös laskeutuu kammioon. Alla oleva kuva havainnollistaa reaktorin rakennetta. [10.]



Kuva 6. PICOSUN™ R-200 Advanced ALD-reaktori, jossa on Picoloader™ Handyman manuaalinen substraatin käsittelyjärjestelmä. [10.]

Kammio lämmitetään lämmityselementeillä, jotka on kiinnitetty kammion ympärille. Ulomman kammion sisäseinän ja lämmityselementtien väliin on lisätty lämmönheijastimia, jotta infrapunasäteily pysyy paremmin reaktiokammiossa. Koska pinnoitettava kohde lämpenee viimeisenä ja reaktiokammion sisäpuolelta ei voida mitata lämpötilaa, pitää lämpötilan antaa tasaantua ennen pinnoittamisen aloittamista. [10.]

Koska on tavoitteena, että lähteet reagoivat ainoastaan pinnoitettavan pinnan päällä, jokaisella lähteellä on omalle linjalleen oma sisääntulo reaktorissa. Eriteltyjen sisääntulojen ansioista linjoissa ei pääse tapahtumaan reaktioita ennen reaktoria. Reaktoriin tullessaan lähteet ja kantajakaasu kulkeutuvat kaasunjakelijan läpi, jolloin lähdeaineet leviävät tasaisesti pinnalle. [10.]

Kammioiden välitilalla on oma kantajakaasulinjansa, jossa on oma massavirtaus- ja painemittauksensa. Reaktiokammion vakuumijärjestelmän sisääntulo sijaitsee ulomman kammion vakuumijärjestelmän sisääntulon sisäpuolella. Välitilan ilmanpaine lasketaan differentiaalisen pumppausaukon kautta, joka yhdistää reaktorin ja välitilan. Pumppaustekniikka varmistaa, että reaktorin paine on aina alhaisempi kuin välitilan. [10.]

Lisäominaisuutena reaktorin yläpuolelle voidaan liittää erillinen plasmageneraattori. Laitteen avulla saadaan lähdekaasuista luotua radikaaleja, jotka reagoivat herkästi. Radikaalit mahdollistavat tehokkaamman ALD-prosessin. [10.]

2.7.2 Kantajakaasujärjestelmä

Kantajakaasujärjestelmällä on kaksi tehtävää: lähteiden kuljettaminen reaktiokammioon sekä kammion huuhtelu. Jokainen lähde on kytketty kantajakaasujärjestelmään pneumaattisella ALD-pulssitusventtiilillä, jonka läpi virtaa kantajakaasua venttiilin tilasta huolimatta. Kun venttiiliä avataan, lähdeainetta pääsee kantajakaasulinjaan, jossa se kulkeutuu kaasun mukana reaktoriin ja näin ollen mahdollistaa ALD-reaktion. [10.]

Pulssitusventtiilien ollessa kiinni voidaan kantajakaasua käyttää reaktionsäiliön huuhteluun. Huuhtelu suoritetaan jokaisen pulssin jälkeen. Lisäksi kantajakaasujärjestelmä palauttaa pinnoittamisen jälkeen vakuumissa olevaan säiliöön normaalin ilmanpaineen. [10.]

2.7.3 Lähdejärjestelmä

Lähdekemikaalit on jokainen kiinnitetty omaan kantajakaasulinjaansa, joten ne eivät pääse sekoittumaan ja reagoimaan keskenään. Picosunin koneisiin saa kerrallaan käyttöön 6–12 lähdettä ja 4–6 lähdelinjaa. Näiden lisäksi kehittyneisiin versioihin voidaan asentaa plasmalähde, jolla on oma kantajakaasujärjestelmänsä. Yleensä plasmalähteiden kantajakaasuna käytetään argonia. [10.]

Lähdettä annostellaan prosessiin pulssitusventtiilin avulla, jonka pulssin pituus määrittää kuinka suuri määrä ainetta annostellaan. Jotta aineet kulkeutuvat oikeaan suuntaan venttiilissä, lähdejärjestelmän paineen tulee olla suurempi kantajakaasulinjan. [10.]

Lähdepulloihin voidaan liittää lämmitys- tai jäähdytyselementti, jolloin saadaan olosuhteiden avulla monipuolisempi lähdevalikoima. Lähdepullon lämmitys tapahtuu pullon alaosan ympärillä olevalla resistiivisellä termoelementillä, sekä pullon yläosaan sijoitetulla termoelementillä. Termoparien avulla huolehditaan, ettei pullo ylikuumene. Peltier-jäähdyttimellä puolestaan viilennetään lähdettä niin, että se on viileämpää kuin ympäröivä huoneilma. Linjoihin voidaan lisäksi liittää toinen venttiili ja kapillaari, joiden avulla voidaan pakottaa kantajakaasu virtaamaan lähdepullon kautta, jolloin vaporisoitunut aine kulkeutuu tehokkaammin reaktoriin. [10.]

Lähdejärjestelmään voi kuulua myös otsonigeneraattori, josta saadaan enimmillään 20 wt%-otsonikaasua. Generaattori tekee hapesta tai hapesta ja typestä otsonikaasua, jota voidaan käyttää lähdeaineena. [10.]

2.7.4 Vakuumijärjestelmä

Vakuumijärjestelmään kuuluu pumppu, hiukkasansa, jälkipoltin ja kaksoissäätöventtiili. Vakuumijärjestelmä on liitetty reaktiokammioon, ja sen tehtävänä on asettaa ulompi kammio vakuumiin eli laskea kammion paine lähelle nollaa. Kun ulompi kammio on vakuumissa, se auttaa eristämään reaktorin huoneilmalta. [10.]

Jälkipolttimen on tarkoitus polttaa reaktorista tulevat ylimääräiset reaktanssit, jotta ne eivät reagoisi keskenään hallitsemattomasti. Palamisreaktiossa käytetään vesi- tai ilmalähdettä. Jälkipolttimen jälkeen sijaitsee hiukkasansa, jonne jäävät palamisesta

syntyneet hiukkaset, jotka eivät enää pysty reagoimaan keskenään. Hiukkasansa suojelee pumppua hiukkasilta ja tukkeutumiselta. [10.]

Kaksoissäätöventtiili V1/V2 yhdistää pumpun muuhun vakuumijärjestelmään. Kaksoissäädön idea on, että voidaan aloittaa pumppaamaan hitaasti avaamalla V2 ja kun kammion paine on tarpeeksi alhaalla, voidaan avata toinenkin venttiili V1 ja pumpata suuremmalla nopeudella. Pumppauslinjassa on painemittaus, joka auttaa kammion paineen määrittämisessä. [10.]

2.8 UV-ALD

UV-ALD-laitteella on tarkoitus luoda prosessiin otsonia UV-lampun ja hapen avulla otsonigeneraattorin sijaan. Toivottu etu prosessissa on, että voidaan ajaa näytteitä matalemmissa lämpötiloissa kuin tavallisessa ALD-laitteessa hyvin tuloksin. Tällöin saavutettaisiin se etu, että voitaisiin pinnoittaa myös korkealle lämpötilalle herkkiä materiaaleja.

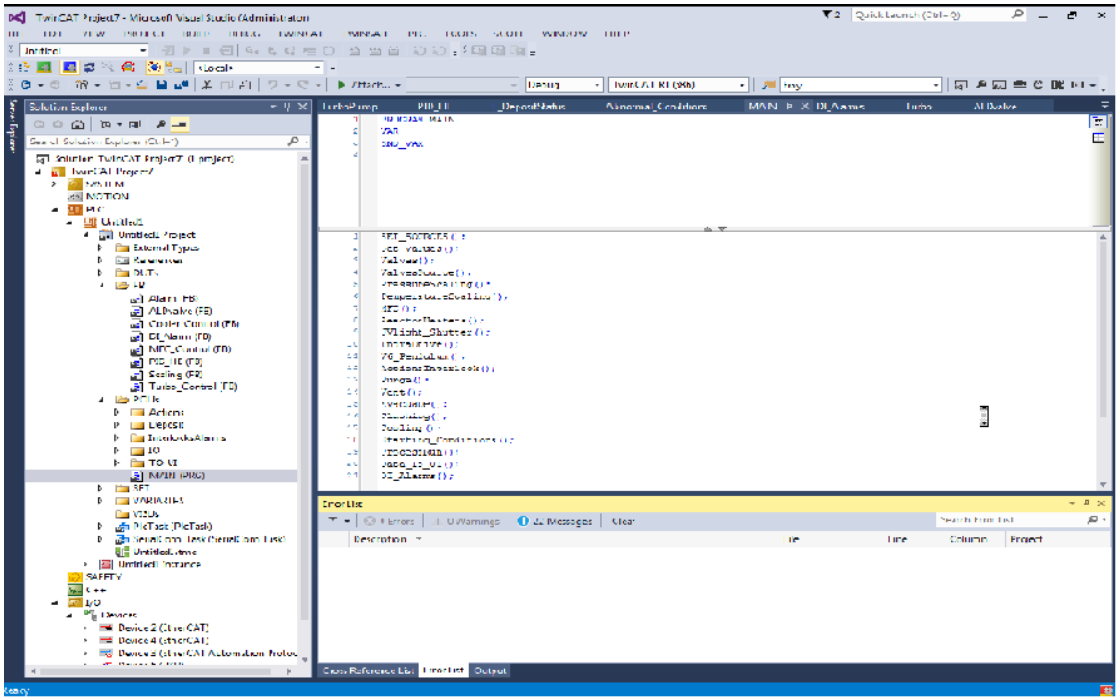
UV-ALD-laite eroaa pääosin tavallisesta laitteesta siinä, että substraattia liikutellaan reaktiokammion sisällä, koska sellaista UV-valokeilaa ei saada kohdistettua koko kammioon. Siksi tässä kehitysprojektissa kammio on suorakaiteen muotoinen, jonka sisällä on näytteenpidike. Näytteen pidikettä liikutellaan kammiossa servomoottorin avulla, joka pyöriessään liikuttaa hammasrattailla olevaa pidikettä.

3 TwinCAT 3

Bechoffin uusin ohjelmankehitysympäristö TwinCAT 3 on rakennettu integroituna Microsoftin Visual Studioon. Ohjelmiston filosofia perustuu siihen, että se yhdistelee informaatioteknologiaa ja tieteellisiä ohjelmistotyökaluja automaatioteknologian kanssa. TwinCAT 3:ssa pyritty suuntaan, jossa ohjelma on modulaarista, eli se koostuu erilaisista osista, kuten koneen yksiköistä ja funktioista, sekä on struktuuriltaan hierarkkista. Eri moduulit pystyvät keskustelemaan, vaikka ne olisi kirjoitettu erilaisilla ohjelmointikielillä, valittavina ovat kaikki IEC 61131 -kielet tai mm. C/C++. Lisäksi TwinCAT 3:lla voidaan ohjelmoida MATLAB®Simulink®-moduulien avulla. Monipuolisuutensa vuoksi ohjelmaa kutsutaan myös nimellä eXtended Automation (XA). [13.]

TwinCAT:in eri moduulit pystyvät kommunikoimaan keskenään ADS-rajapinnan kautta.

Tietokoneohjelmoinnin yhdistelyn perinteiseen logiikkaohjelmointiin huomaa ohjelmointikielien lisäksi esimerkiksi siitä, ettei ohjelmoidessa konfiguroinnin jälkeen tarvitse välittää I/O-osoitteista. Kun kerran konfiguroidut tulot ja lähdöt on linkitetty muuttujiin, niitä voidaan käyttää kuin mitä tahansa muuttujia, eli niitä voidaan kutsua nimellä missä tahansa ohjelmassa. Kuvassa 7. nähdään TwinCAT 3:en ikkunanäkymä.



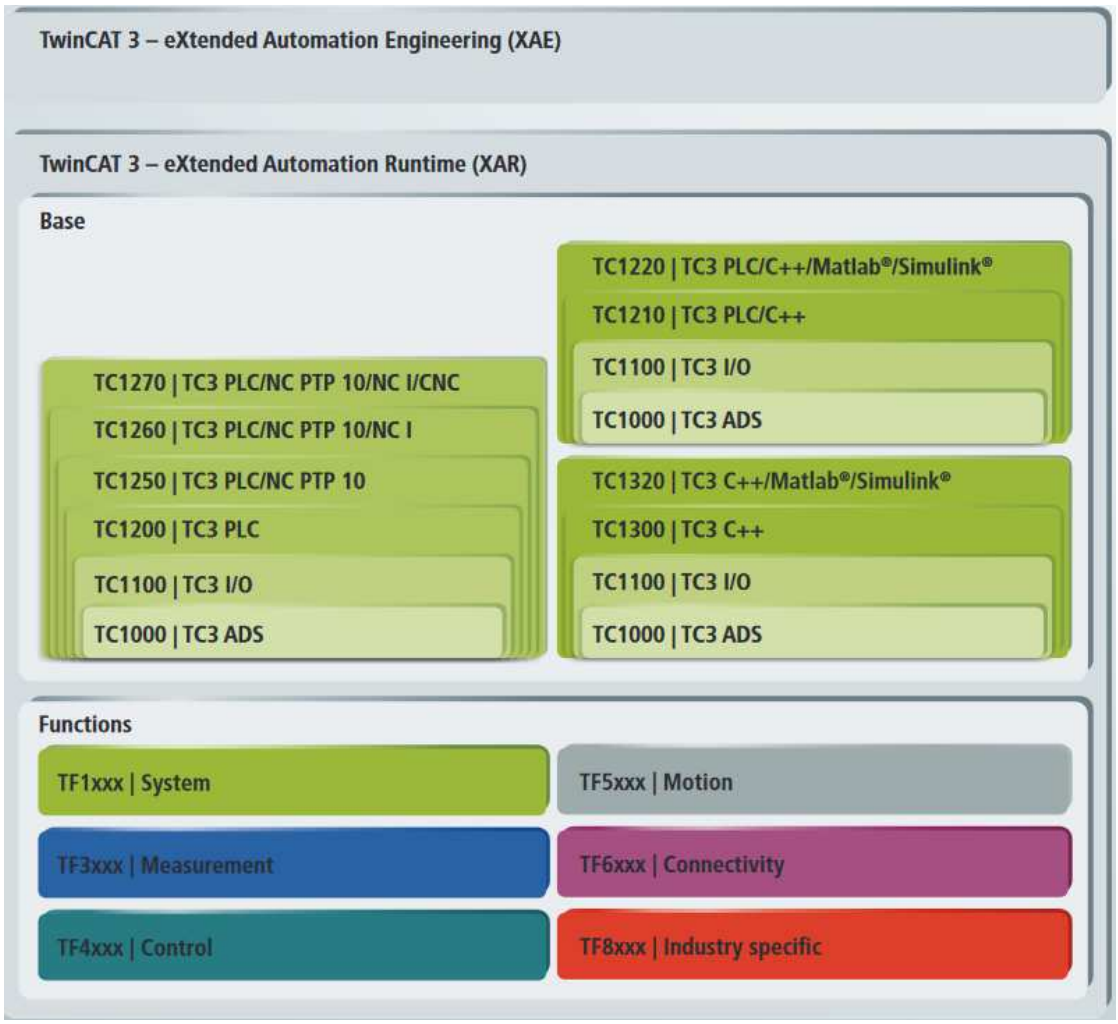
Kuva 7. TwinCAT 3 ikkunanäkymä.

Yksi TwinCAT 3:en eduista on se, että moniytimisillä laitteilla PLC-projekteja voidaan jakaa ytimien kesken niin, että ytimillä on omat projektit suoritettavinaan. Tällöin saadaan suuria hyötyjä tehokkuudessa. Ominaisuuden avulla voidaan myös esimerkiksi kaikissa tietokoneissa asettaa, kuinka monen ytimen halutaan suorittavan Windowsia ja kuinka monta annetaan TwinCAT:ille käyttöön. [14.]

3.1 Ohjelmiston rakenne

Ohjelma on jaettu kahteen eri versioon: eXtended Automation Engineering (XAE) ja eXtended Automation Runtime (XAR). TwinCAT eXtended Automation Engineering (XAE) on moduulien suunnittelua ja tekemistä varten, kun taas eXtended Automation

Runtime (XAR) on tuote, jossa voidaan ladata, suorittaa ja hallita tehtyjä moduuleja. [13.]



Kuva 8. TwinCAT 3:n rakenne, joka koostuu päällekkäin rakentuvista komponenteista.

Kuvassa 8. on selvennetty TwinCAT 3 -rakennetta, ja voidaan huomata, että eXtended Automation Runtime on jaettu vielä kahteen luokkaan: TC3 Base ja TC3 Functions. Jokainen luokka, eli siis Engineering, Base ja Functions, koostuu komponenteista. Basen komponentit ovat perusosia, ja sen avulla onnistuu esimerkiksi I/O-datan keruu prosessikuviin (process images). Functions komponenteilla voidaan laajentaa perustoimintoja, kuten tehdä datasta visualisointeja tai tallentaa mittauksia. Engineering on tarkoitettu suunnittelua varten, joten sen komponentit ovat konfigurointia, ohjelmointia ja ohjelman ajoa varten. Valitsemalla komponentteja käyttöönsä voi ohjelmasta muokata juuri omiin käyttötarkoituksiin sopivan. [15.]

Projektissa käytettyjä TwinCAT-komponentteja olivat TF4110 Temperature Controller ja TF6340 Serial Communication.

3.2 TC3 Engineering

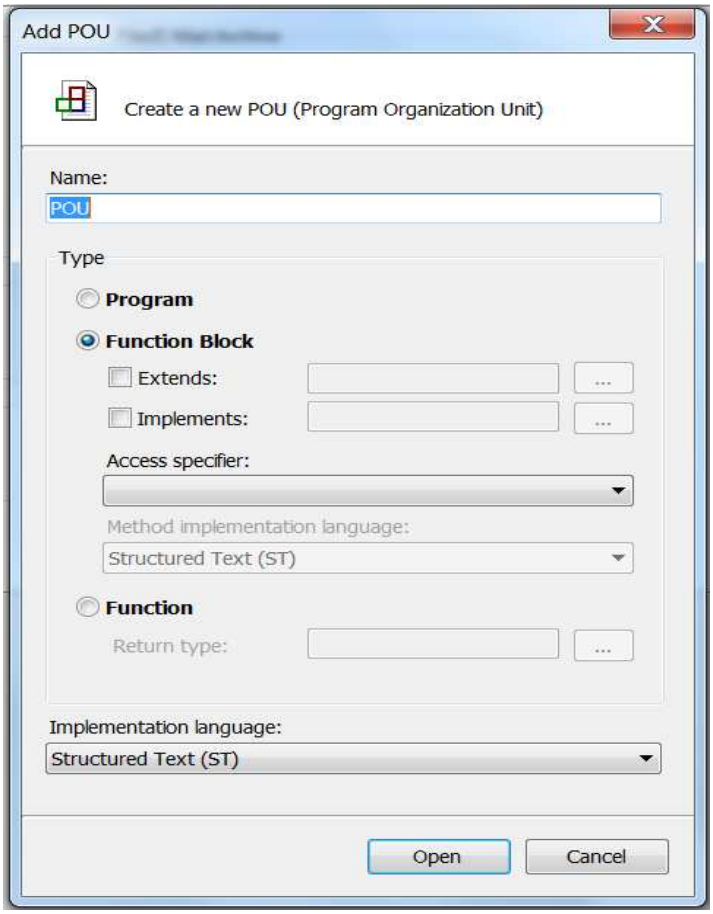
3.2.1 Ohjelmoinnin perusrakenne

Ohjelmointi TC3 Engineeringissä perustuu standardiin IEC 61131-3 ja OOP:hen (Object Oriented Progrramming), ja näitä kahta yhdistellen on saatu monipuolinen logiikkaohjelmointiympäristö. TwinCAT 3:ssa ohjelma koostuu POU:ista (Program Organization Unit) eli ohjelman rakenteellisista yksiköistä ja niiden laajennuksista. [13.]

3.2.2 POU:t

Ohjelmointiympäristö sisältää kaikki standardissa IEC 61131-3 kuvatut rakenteelliset yksiköt eli POU:t. Erilaisia POU-tyyppejä ovat Program, Function Block ja Function, eli Ohjelma, Toimilohko ja Funktio. Tyypistä riippuen voidaan POU:hun lisätä metodeita, ominaisuuksia, toimia ja siirtymiä (TwinCAT:ssa method, property, action ja transition). Jokaisella POU:lla on oma määrittelyosionsa, johon voidaan määritellä sisäiset muuttujat, sekä osio johon suoritettava ohjelma kirjoitetaan. [13.]

POU:n voi lisätä Solution Explorer -valikon puurakenteessa (kts. kuva 14) PLC-projektin alle, ja kansioinnin avulla ohjelman yksiköitä voidaan järjestää. Oletuksena PLC-projektissa on kansio nimeltä VISUs ja sen alla kansio POUs, jossa on valmiina pääohjelman POU. Lisättäessä uusi POU avautuu alavalikko, jossa voidaan määritellä POU:n nimi, tyyppi sekä se, millä ohjelmointikielellä se toteutetaan. Kuvassa 9 esiintyy tämä alavalikko. [13.]



Kuva 9. POU:n lisääminen.

Ohjelmatyypin POU toimii kuten yleensä ohjelmat, eli se pystyy palauttamaan useita arvoja kerrallaan ja arvot luetaan suorituksen aikana. Luetut arvot säilytetään seuraavaa ohjelman suoritusta varten. [13.]

Toimilohkotyypin POU:t palauttavat myös yhden tai useamman arvon sekä säilyttävät outputtien ja sisäisten inputtien arvot seuraavalle suorituskerralle. Tällöin samoilla tuloilla ei välttämättä aina seuraa samaa arvoa lähdöissä, toisin kuin funktiotyypin POU:issa. Toimilohkoja voidaan jatkaa toisilla toimilohkoilla, sekä niissä voidaan käyttää työkaluina metodeita. Toimilohkoja kutsutaan aina niiden instanssien avulla, joka on tavallaan kopio toimilohkosta. Jokaisella instanssilla on oma nimi ja ne määritellään kuten muuttujat, ne voivat olla globaaleita tai paikallisia. [13.]

Funktiot puolestaan palauttavat vain yhden dataelementin. Koska funktiot palauttavat aina saman arvon samoista sisääntuloista, eivät ne voi sisältää globaaleita muuttujia.

IEC 61131-3 -standardin mukaan funktioissa tulee olla mahdollisuus useisiin outputeihin, joten funktioihin voidaan määrittää useampi ulostulo. [13.]

POU:iden hierarkia riippuu siitä, missä järjestyksessä puurakenteessa niitä kutsutaan, sillä POU:t voivat myös kutsua toisiaan. Kuitenkin ohjelman suoritus tapahtuu pääohjelmassa, josta täytyy silloin kutsua seuraavaa hierarkiatasoa. [13.]

3.2.3 POU:den laajennukset

Jos POU:t tukevat IEC 61131-3 -standardia, niin niihin lisättävät laajennukset tuovat OOP:n eli olio-ohjelmoinnin ohjelmointiympäristöön. Metodeita voidaan käyttää työkaluna käskysarjojen määrittelyyn. Metodi ei ole itsenäinen ohjelman yksikkö, vaan sen voi liittää osaksi toimilohkoa, muuten se muistuttaa paljolti funktiota. Työkalua voidaankin ajatella funktiona, joka sisältää toimilohkonsa instanssin, sillä metodi palauttaa arvon ja sillä on myös oma määrittelyosionsa kuten funktiolla. [13.]

3.2.4 DUT

Standardien datatyyppien, kuten boolean tai integer, lisäksi käyttäjä voi määritellä omia datatyyppejä DUT:eja eli Data Type Uniteja projektiinsa. Nämä erilaiset datatyypit toimivat ohjelmoijan työkaluina, jotka helpottavat ohjelmoijan työtä, esimerkiksi vähentämällä kirjoitettavan koodin määrää ja ulkoa muistamisen tarvetta. Erilaisia DUT:eja ovat struktuurit, enumeraatiot ja referenssit. [16.]

Struktuurit ovat datatyyppejä jotka sisältävät joukon muuttujia, jotka voivat olla mitä tahansa TwinCAT:in datatyyppejä. Struktuurit auttavat vähentämään kirjoitetun koodin määrää, kun yksi muuttuja sisältää useampia parametrejä. Struktuureja voidaan käyttää muuten kuten tavallisia muuttujia, mutta niitä ei voida linkittää IO:lle. [16.]

Enumerointi auttaa ohjelmoijaa, jos halutaan indeksoida joitain muuttujia. Tällöin siis halutaan luoda muuttuja jolla voi olla erilaisia arvoja, joista jokainen kuvastaa jotain tilannetta, olotilaa tai vaikka laitemallia. Enumeroinnilla voidaan näille indekseille antaa nimet, jotka ohjelmassa siis palauttavat tämän numeron. Erona indeksointiin enumeroinnilla on oikeastaan se, että ohjelmoijan ei tarvitse ulkoa muistaa, mitä mikäkin numero tarkoittaa, ja näin ollen myös mahdollisuus virheille tässä vähenee.

Enumerointia voidaan käyttää kun halutaan ohjelmassa tarkailla ja luoda erilaisia statuksia ja tiloja, joiden avulla ohjataan lohkoja tai saadaan tietoa niiden tilasta. Lisäksi enumerointi auttaa modulaarisessa ohjelmoinnissa, jolloin voidaan tehdä ohjelman varioinnit riippumaan annetuista indekseistä. Silloin ei tarvitse kirjoittaa ohjelmaa aina uudelleen vaan voidaan indeksien avulla määrätä, mitkä moduulit ovat milläkin variaatioilla käytössä.

Referenssit puolestaan ovat aliaksia muuttujille. Alapuolella esimerkki referenssien käytöstä (Kuva 10).

Example declaration:

```
ref_int : REFERENCE TO
INT;
a : INT;
b : INT;
```

ref_int is now available for being used as an alias for variables of type INT.

Example of use:

```
ref_int REF= a; (* ref_int
now points to a *)
ref_int := 12; (* a now has value 12 *)
b := ref_int * 2; (* b now has value 24 *)
ref_int REF= b; (* ref_int now points to b *)
ref_int := a / 2; (* b now has value 6 *)
ref_int REF= 0; (* explicit initialization of the reference*)
```

Kuva 10. Referenssin määrittely ja käyttöesimerkki

3.3 Lisenssit

Jokainen voi kokeilla TwinCAT 3:a ilmaiseksi, sillä ohjelmalla on 7 päivän ilmainen kokeilulisenssi. Lisenssin voi päivittää ohjelmassa aina sen päätyttyä, eli käytännössä saa ilmaiseksi täyden version käyttöön. Lisenssin päivitys onnistuu myös tässä vain muutamalla klikkauksella ohjelmassa.

4 Toteutus

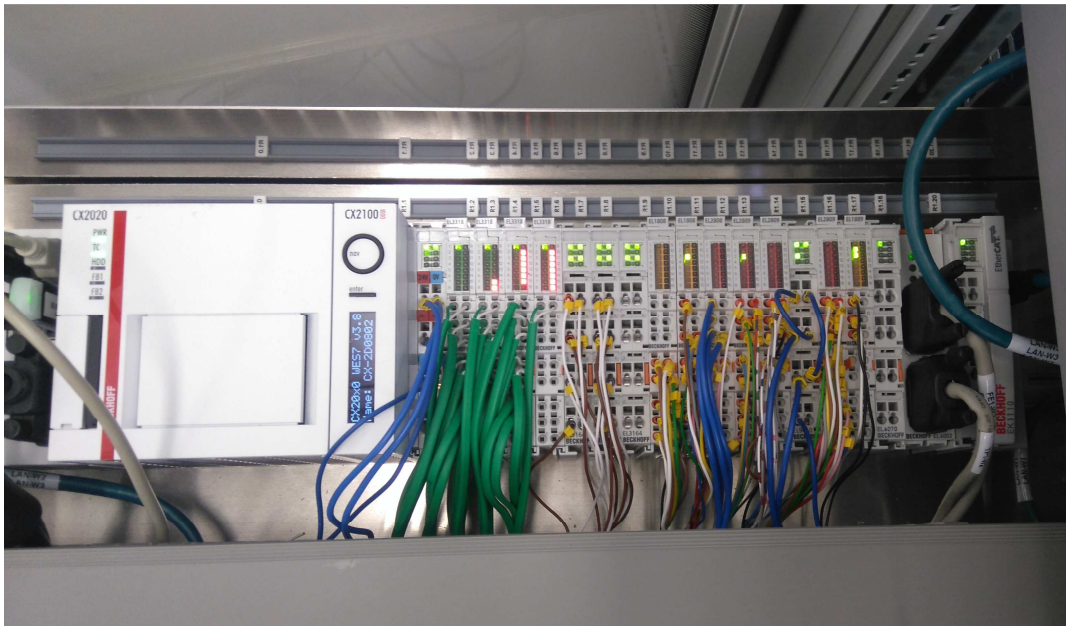
4.1 Esitiedot

Esitietona projektin toteutukselle olivat PI-kuva, lyhyt toimintakuvaus, sekä yrityksen käytössä oleva pohjaohjelma ALD-laitteille. Pohjaohjelma on tehty toisella logiikkaohjelmointiohjelmalla ja on pääosin Ladder-kielellä toteutettu.

Lisäksi esitietona toimivat projektin laitteiden manuaalit.

4.2 Järjestelmä

Projektin logiikkana toimii Beckhoffin Embedded-PC CX2020. Logiikkaan on kytketty IO:ta seuraavasti: 4 TC-korttia, 3 AI-korttia, 3 DI-korttia, 4 DO-korttia sekä sarjaliikenteellä kytketty 2 laitetta ja EtherCAT-kenttäväylään kytketty 8 laitetta. Lisäksi vaatii erillisen virtalähdeyksikön CX2100, joka toimii myös tiedon ja virranvälittimenä I/O:lle. Kuvassa 11 nähdään projektin logiikka kytkettynä.



Kuva 11. Projektin logiikka

4.2.1 Beckhoff Embedded-PC

Embedded-PC eli sulautettu PC yhdistelee PC-teknologiaa ja PLC-teknologiaa sekä modulaarista I/O-tasoa. Toisin sanoen laitteessa teollisuustietokone ja nykyaikainen automaatioprosessienohjaus kohtaavat. Beckhoffin CX-sarjassa on useita erilaisia ja eritehoisia vaihtoehtoja; erilaisille käyttötarkoituksille on valittavana 15 erilaista laitetta. [17.]

Työssä käytetyt Beckhoffin CX-sarjan sulautettu PC on Windows pohjainen ja siinä ei ole tuulettimia eikä pyöriviä komponentteja. CX2020:ssa on 1,4GHz:n Intel® Celeron® prosessori ja 2 GB RAM-muistia. Liitäntä mahdollisuuksia ovat kaksi itsenäistä Ethernet-porttia, DVI-D-portti ja neljä USB-porttia. Ethernet-portit on konfiguroitu niin, että ylempi on IT-portti ja alempi on lähettävänä osapuolena EtherCAT-kommunikointia varten. [17.]

CX2020 vaatii erillisen virtalähdeyksikön CX2100, joka toimii myös tiedon ja virranvälittimenä I/O:lle. CX2100 sisältää lisäksi näytön, josta voidaan nähdä esimerkiksi sulautetun PC:n nimi ja tila, sekä selata parametreja. Käyttöjärjestelmänä CX2020:ssa on Windows Embedded Standard 7 P. Käyttöjärjestelmän ohessa laitteissa pyörii TwinCAT, jolloin laitteista tulee monipuolinen ja tehokas PLC, jossa voi pyörittää useita tehtäviä saman aikaisesti. TwinCAT:in käskyt ovat järjestelmässä suuremmalla prioriteetilla, eli laite kuuntelee ensin TwinCAT:ia sitten vasta Windowsia. [17.]

Etuna Beckhoffin sulautetuissa PC:ssä ja TwinCAT 3:ssa on, että samalla laitteella voidaan pyörittää kehitysympäristöä, logiikkaohjelmaa ja HMI:tä. Tällöin ei tarvita välttämättä enää erillistä kannettavaa tietokonetta, vaan jos logiikkaohjelmaa halutaan parannella, voidaan se tehdä hiiren ja näppäimistön avulla paikan päällä. Samalla laitteella pyörivät logiikkaohjelma ja HMI vaihtavat myös nopeasti tietoa keskenään.

4.2.2 Windows Embedded

Windows Embedded on Microsoftin tuoteperhe, joka tarjoaa alustoja sulautetuille järjestelmille, kuten älypuhelimille ja teollisuustietokoneille. Windows Embedded Standard on kehittyneitä laitteita varten suunniteltu käyttöjärjestelmä, joka on suunnattu laaja-alaisesti teollisuudenalojen yritysten tarpeisiin. Tuotetta käytetään esimerkiksi asiakaspäätteissä, kasinopeleissä ja HMI-laitteissa. [18.]

4.2.3 EtherCAT väylä

Tässä työssä käytettiin EtherCAT-väylää, jonka avulla pyrittiin vähentämään kaapelien määrää sekä helpottamaan ohjelmointia ja kommunikointia. EtherCAT:in etuna on lyhyet sykliajat ja lyhyt viiveen vaihtelu. EtherCAT on Beckhoffin kehittämä reaaliaikainen protokolla.

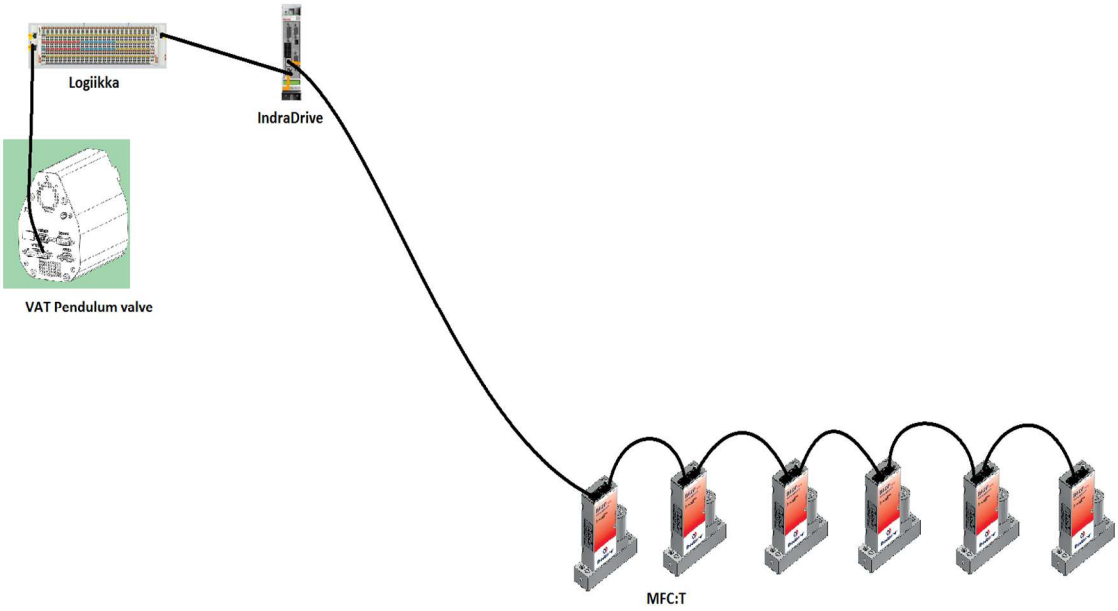
EtherCAT lyhyt sykliaika perustuu siihen, että erillisten pakettien sijaan, lähetetäänkin sanoma, joka kulkeutuu jokaisen verkon solmun läpi. Eli sanoma lähetetään jokaiselle laitteelle, joka verkkoon on kytketty. Sanomaa luetaan ja siihen kirjoitetaan dataa samanaikaisesti kun se liikkuu eteenpäin. Datan liikkumista EtherCAT-väylässä voitaisiin kuvata metrona, jonka pysäkeillä matkustajat jäävät pois vauhdissa ja samalla tulevat kyytiin. [19.]

EtherCAT-isäntä, joka tässä projektissa on CX2020, lähettää sanoman jokaiselle orjalle, jotka ovat EtherCAT-laitteita. EtherCAT-isäntä on ainut solmu verkossa, joka pystyy muodostamaan ja lähettämään EtherCAT-kehyksiä, orjat lähinnä siirtävät kehyksiä eteenpäin verkossa. Kun sanoma saavuttaa avoimen portin, joka tarkoittaa verkon loppupäätä, se lähettää viestin takaisin isännälle. [19.]

Sanoma sisältää paketteja, jotka on aina osoitettu tietyille laitteille. Laitteet tunnistetaan niille annetun EtherCAT-osoitteen perusteella ja jokainen paketti osoitetaan tietylle osoitteelle. Laitteet lukevat vain niille itselleen tarkoitetun paketin sanomasta sekä kirjoittaa pakettiin. [19.]

EtherCAT on Ethernet-pohjainen kenttäväylä, jossa tietoa siirrettään Ethernet-kehyksessä. Kehyksen sisällä tiedonsiirto tapahtuu omalla protokollalla. Kuten Ethernet-verkkossa, EtherCAT-väylän kytkemiseen käytetään tavallisia RJ45-liittimiä. EtherCAT-laitteissa on yleenä kaksi porttia, joista toinen väylän sisääntulolle ja toinen ulostulolle, josta voidaan jatkaa väylää topologiassa eteenpäin. Jos ulostuloportti jätetään tyhjäksi silloin tulkitaan, että väylä päättyy tähän. [19.]

EtherCAT-väylä voidaan kytkeä monien eri topologiamallien mukaisesti, tässä työssä käytin linjatopologiaa (kts kuva 12). EtherCAT-laitteita voidaan kytkeä samaan linjaan jopa 65 353 laitetteeseen asti. [19.]



Kuva 12. Projektin EtherCAT-väylän topologiakuva

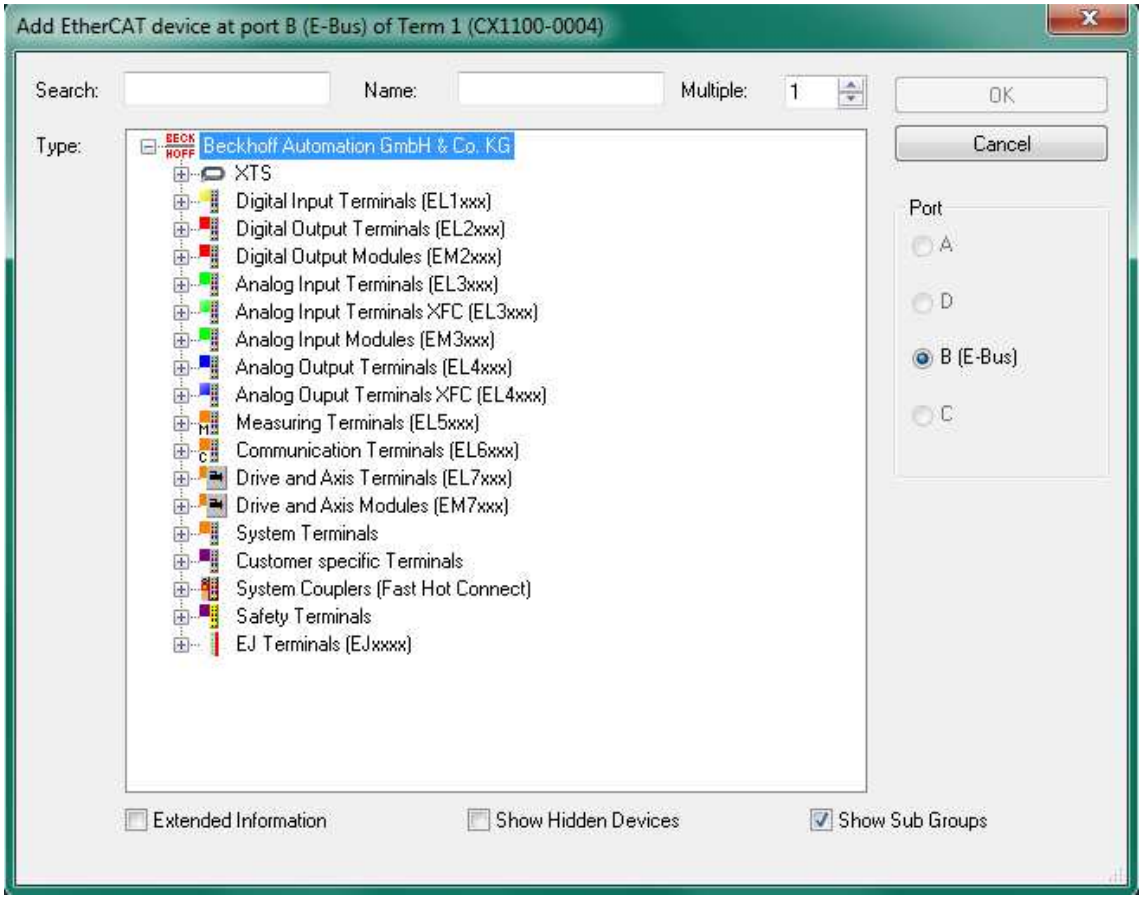
4.3 Projektin aloitus

Esitietoina ohjelmasuunnitteluun mekaniikkasuunnittelijoilta saatiin PI-kuva laitteesta ja toimintakuvaus. Suuri osa ohjelmoinnissa käytettävästä esitiedosta pohjautui vanhoihin projekteihin ja projekteissa käytettävään pohjaohjelmaan jonka päälle projektit yleensä personoidaan.

PI-kuvan perusteella mietittiin, millaista IO:ta tullaan tarvitsemaan. Ensin otettiin selvää instrumenttien teknisistä tiedoista ja siitä, kuinka ne voidaan liittää logiikkaan. Osassa instumenttivalinnoissa oltiin mukana lähinnä liitäntävaihtoehtojen takia, jotta saataisiin mahdollisimman moni laite EtherCAT-väylään.

Instrumenttien perusteella tehtiin alustava IO-lista, jossa oli eroteltu AI:t, AO:T, DI:T ja DO:t, ja eroteltu niiden erilaiset tyypit. Esimerkiksi DI:tä löytyi NPN- sekä PNP-tyyppiä. Tämän listan avulla valittiin logiikan komponentit Beckhoffin Ville Hopposen kanssa tapaamisessa, jossa keskusteltiin logiikan ja IO-korttien valinnasta. Tapaamisen jälkeen saatiin valmis IO-lista, joka voitiin toimittaa jälleen sähkösuunnittelijalle. [20.]

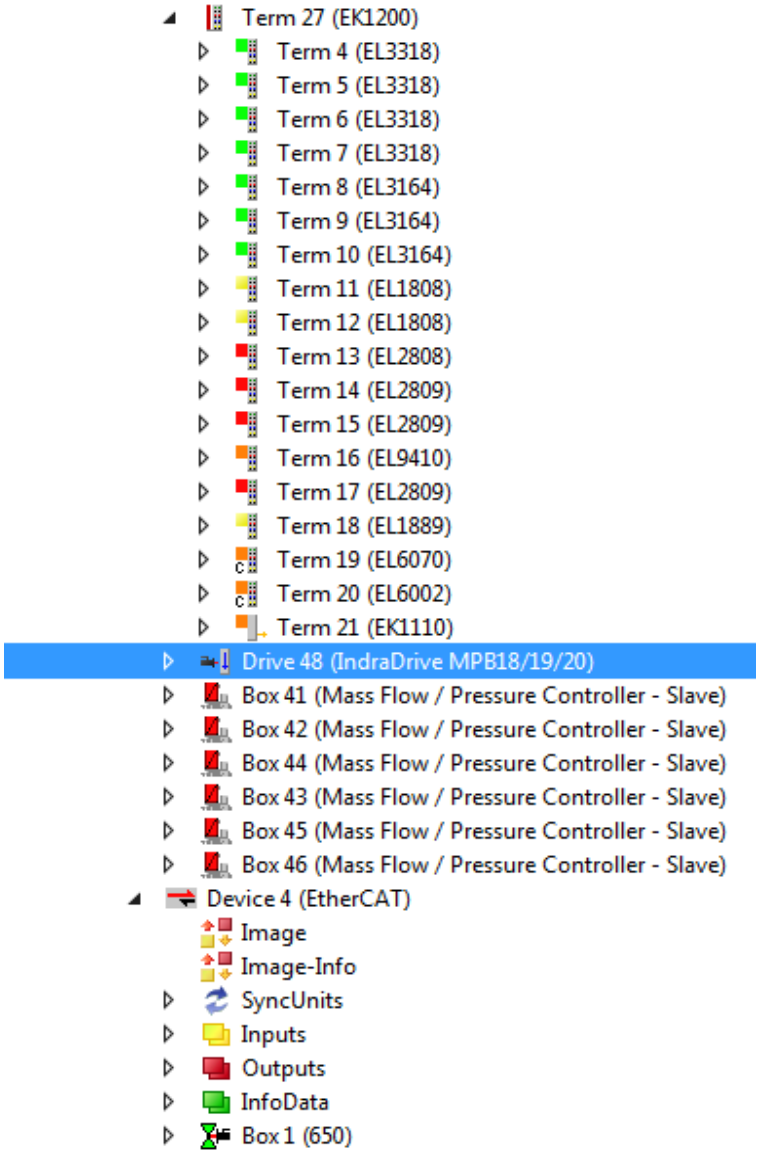
Kun tiedettiin, millaista IO:ta projektissa ohjataan, voitiin luoda TwinCAT-projekti ja rakentaa siihen IO-listan mukainen konfiguraatio. TwinCAT:issä löytyy puuvalikosta kohta ”Devices”, jonka alle lisätään ainakin yksi EtherCAT Master-laite, joka siis kuvastaa tässä tapauksessa logiikan CX2020:aa eli Embedded PC:tä. Tämän Masterin alle voidaan lisätä IO:ta, ja oikean hiiren painikkeen avulla löydetään valikko, josta löytyvät lähes kaikki Beckhoffin komponentit. Alla olevassa kuvassa 13 nähdään täm komponenttivalikko.



Kuva 13. Komponenttivalikko

Sama voidaan tehdä valitsemalla projektissa valitsemalla ”Scan”-toiminto, mutta vain jos logiikka on jo liitettynä PC:lle, jossa TwinCAT:in ohjelmointiympäristö on avattuna. Käyttöönotossa on joka tapauksessa järkevää skannata laitteet, jotta voidaan varmistaa että esimerkiksi EtherCAT-verkko löytää kaikki samat laitteet, jotka oli suunniteltu projektiin.

EtherCAT-laitteiden ESI-tiedostot, eli XML-muotoiset laitekuvaukset, lisätään samaan kirjastoon, missä tiedot muusta IO:sta sijaitsee. Laitekuvaukset kertovat EtherCAT Masterille, millaista tietoa voidaan lukea tai kirjoittaa laitteelle, sekä synkronointivaihtoehdot. Kun tiedosto on liitetty kirjastoon, laitteen voi valita samasta valikosta kuin muutkin komponentit.



Kuva 14. Projektin konfiguraatio TwinCAT:issa

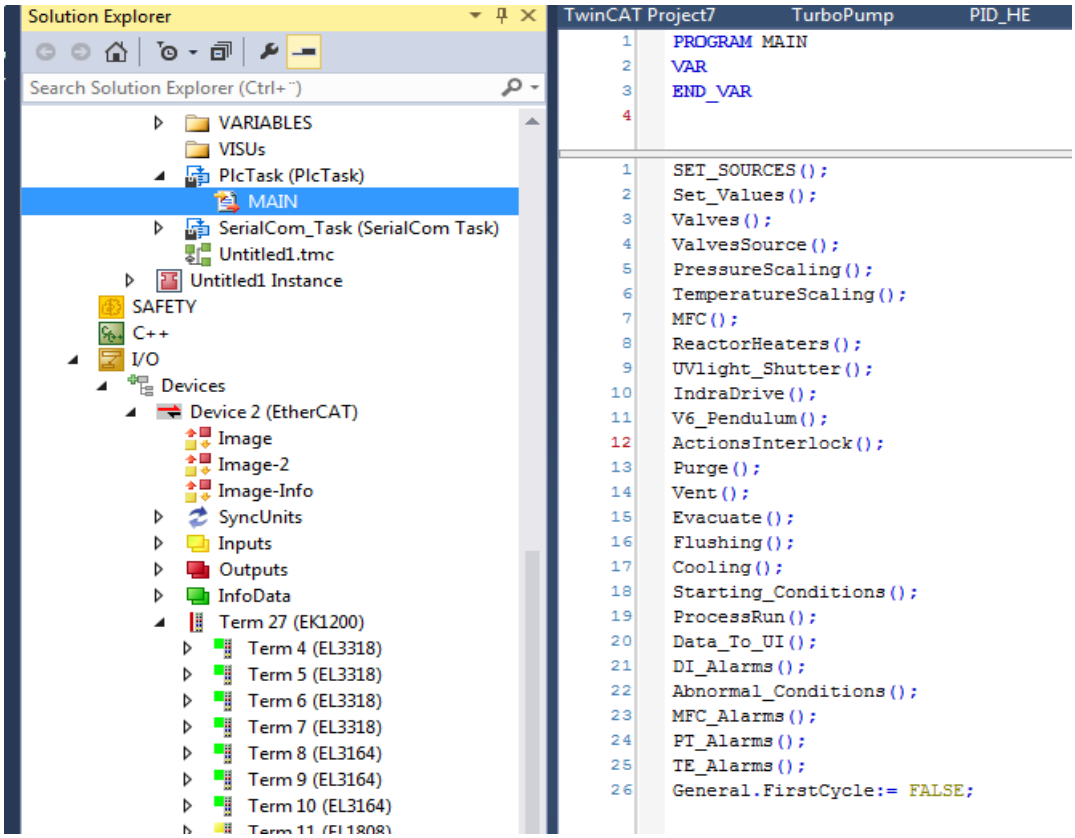
Kun konfiguraatio on lisätty, voidaan vielä tarkistaa, riittääkö E-bussissa virtaa koko IO:lle. Kuvassa 14 konfiguraatio TwinCAT:ssä. Ohjelmointiympäristöstä nähdään, kuinka paljon jokainen IO-komponentti kuluttaa virtaa E-bussista ja paljonko sitä on jäljellä missäkin kohtaa. Tällöin tiedetään jo tässä vaiheessa, jos tarvitaan esimerkiksi

lisää virtalähteitä. Konfiguraatio voidaan myöhempää käyttöä varten tallentaa erillisenä tiedostona.

4.4 Ohjelmointi

Ohjeilmointiin käytin IEC 61131 -standardin mukaista Stuctured Text -ohjelmointikieltä eli ST:tä. Kieli on tarkoitettu ohjelmoitavien logiikoiden ohjeilmointiin.

Aloitin ohjelmoinnin luomalla globaalit muuttujat, jotka linkitettiin IO-muuttujiin. TwinCAT:in puuvalikko auttaa ohjelmanjäsentelyssä, sillä valikkoon voidaan luoda kansioita jonka avulla ohjelman POU:t ja FB:t saadaan järjestykseen.



Kuva 15. Projektin pääohjelma

Ohjelman sykliaika määritellään luomalla tehtävä eli task, jossa ohjelmaa suoritetaan. Tehtävän asetuksista päästään määrittelemään sykliajan lisäksi määrittämään mm. kyseisen tehtävän prioriteetti ohjelman suorituksessa. Tehtäviä voidaan luoda useita, jos esimerkiksi halutaan suorittaa ohjelman eri osia eri nopeuksilla ja eri

tärkeysjärjestyksessä. Tehtävään linkitetään pääohjelma, eli tässä MAIN, jossa muita aliohjelmia kutsutaan (kuva 15).

4.4.1 Lämpötilansäätö

Projektin laitteessa on lämmitettävä kammio, jossa on 12 x 280 W:n lämmitintä, joilla jokaisella on oma lämpötilamittaus. Lisäksi kammiossa on kaksi lämpötilamittausta, joilla kontrolloidaan prosessin lämpötilaa.

Päätin toteuttaa lämmönsäädön vain yhdellä PID-säätimellä, jolloin kammiolle voidaan asettaa yksi tavoitelämpötila. Tällöin lämpötilan säätöön käytetään vain toista kontrollilämpötiloista. Puolestaan toista kontrollilämpötilaa voidaan vertailla tähän lämpötilaan ja tarkkailla näiden kahden eroa.

Lämpötilan ohjelmointiin tarvittiin erillinen Functions-paketti Temperature controller toolbox. TwinCAT:iin ladattiin kirjastot TC2_ControllerToolbox ja TC2_TempController. Lämpötilansäätöön löytyy TwinCAT:issä useita vaihtoehtoja. Valittiin valmis funktioblokki nimeltä FB_CTRL_TempController. Tämä lämmönsäätölohko on monipuolinen ja siitä löytyy useita säätöjä ja parametrejä, mutta yksinkertaisessa lämpötilansäädössä tarvitaan vain pientä osaa näistä.

Lämmönsäätölohkosta löytyy sekä analoginen että digitaalinen lähtö, joista valittiin digitaalisen, koska käytetyt heaterit ovat digitaalisilla lähdöillä releohjattuja. Lohkon digitaalilähdöstä saadaan signaali, jonka pulssin leveys on säädetty PID-säätimellä.

```
ControllerParameter: ST_CTRL_TempCtrlParameter := (
iMode := 1,
iReactionOnFailure := eCTRL_ReactionOnFailure_StopController,
dwAlarmSupp := 16#EFFF,
tCtrlCycleTime := TIME#500MS,
tTaskCycleTime := TIME#10MS,
iTuningMode := 1,
fYTuneHeating := 100.0,
fWMax := 85.0,
fWStartUpVeloPos := 1.0,
fWStartUpVeloNeg := 1.0,
fWVeloPos := 1.0,
fWVeloNeg := 1.0,
fYMax := 100.0,
tPWMCycleTime := TIME#5S0MS,
tPWMMinOffTime := TIME#500MS,
iFilterType := E_FilterType_FIRSTORDER,
iControllerType := eCTRL_ControllerType_PID,
TempHigh := 150,
TempHighHigh := 150,
TempAbsoluteHigh := 200);
```

Kuva 16. Lämpötilasäätimen struktuuri, joka sisältää säätöön tarvittuja parametrejä.

Itse säätimen parametrit sijaitsevat struktuurissa ST_CTRL_TempCtrlParameter, jollainen ohjelmoijan pitää muodostaa, koska lämmönsäätölohko tarvitsee struktuurin yhdeksi lohkon sisääntuloista. Kuvassa 16. osa näkyy osa struktuurin parametreistä. Säätimen tilaa säädellään puolestaan enumeroinnilla E_CTRL_MODE.

4.4.2 Enumeroinnit

Enumerointeja määriteltiin projektia varten 4 erilaista. Enumeroinneilla tässä projektissa toteutettiin muuttujia jotka muuten olisi toteutettu indeksoinnilla. Tällaisiä muuttujia olivat erilaiset lähdetyypit, sekä laitteiden tai prosessin tila.

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_SourceModel :
(
    Empty := 0,
    Picosolution := 1,
    Picosolid := 2,
    Picohot200 := 3,
    Picohot300 := 4,
    Picogas := 5
);
END_TYPE
```

Kuva 17. Lähteiden enumerointi

Lähdetyyppien enumeroinnin tai indeksoinnin avulla voidaan ohjelmassa määritellä, mitkä ohjelman osat toteutetaan minkälaisenkin lähdetyypin kanssa (kuva 17). Esimerkiksi toteutetaanko lämmityslohkoa tai viilennyslohkoa.

Tässä projektissa käytettiin myös enumerointia määrittämään prosessin kulkunvaiheita, massavirtaussäätimien erilaisia tiloja ja turbomoottorin tiloja.

4.4.3 Toimilohkot

Jotta ohjelmasta saadaan modulaarisempaa, on hyvä luoda toimilohkoja eli function blockeja. Toimilohkot kannattaa toteuttaa aina niille ohjelman osille, jotka muuten toistuisivat ohjelmassa.

Tässä projektissa loin toimilohkot seuraaville:

- ALD-venttiilittiilien ohjaus
- massavirtasäätimien ohjaus
- lämmittimien ohjaus
- jäähdyttimien ohjaus
- lukemien skaalaus
- hälytykset
- digitaalitulojen hälytykset

- turbon ohjaus

Laitteessa on yhteensä 17 venttiiliä, joista yksi on säätöventtiili. Kuitenkin vain lähteiden venttiilien ohjaukselle oli järkevää tehdä omat toimilohkot, niin että jokaisen lähteen venttiilejä ohjattiin yhdestä toimilohkon instanssista.

```
MFC200(
    PurgeSel1:= UI.PurgeSel_B1, //Purge selection line, source 1
    PurgeSel2:= UI.PurgeSel_B2, //Purge selection line, source 2
    PurgeExec:= UI.Purge_Execute, // Purge execute
    FlowMode:= General.FlowMode, // Enumeration for flowmodes
    Measured_Max:= 20000, // Max measured value without scaling
    Raw_In:= AI.MFC200, //Measured value
    Scal_Max:= 200, //Max scaled value
    Manual_SP:= UI.MFC200_SP_Man, //Manual Setpoint
    Deposit_SP:= UI.MFC200_SP_Deposit, //Process setpoint
    Pulse_SP:= , //Setpoint when pulse
    Stabil_SP:= UI.StabLineFlow, //Stabilization flow setpoint
    Purge_SP:= Purge.Flow, // Purge setpoint
    Venting_SP:= UI.Venting_SourceMFC,  //venting setpoint
    PV=> IO.MFC200, //Present value
    SP=>IO.MFC200_SP, //Active setpoint
    RAW_SP=> AO.MFC200); // Scaled Setpoint for the controller
```

Kuva 18. Massavirtasäätimen toimilohkon tulot ja lähdöt.

Massavirtasäätimien ohjaus tehtiin lohkoilla (kuva 18), joissa enumeraation avulla säädeltiin useiden eri asetusarvojen vaihtoa. Tällöin vältyttiin kirjoittamasta ohjelmassa useaan kertaan massavirtasäätien asetusarvoja, vaan voitiin muuttaa enumeroidun muuttujan General.Flowmode:n tilaa. Lisäksi toimilohkon sisällä oli kaksi skaalaustoimilohkoa, joilla saatiin skaalattua luettu virtaus sekä asetusarvo oikeisiin lukemiin.

4.4.4 Prosessin eteneminen

Prosessin aloittamiselle luotiin ensin ehdot joiden tulee täyttyä ennen kuin prosessin aloittaminen sallitaan. Prosessin aloittamisen jälkeen sen vaiheen määrittelee enumeraatio muuttuja Deposit.Status. Muuttujan arvo muuttuu aina tietyn vaiheen siirtymisehtojen täytyttyä (kuva 19).

```
DepositStart(CLK:=Deposit.ProcessRun , Q=>Start_Redge );
IF  Start_Redge THEN
    Deposit.Status := E_DepositStatus.Heating;
END_IF
// Wait for stabilization, If Uv is used this is used as UV light warm up time as well
Wait(IN:=Deposit.AllTemps_OK AND Deposit.Status = E_DepositStatus.Heating, PT:=T#60S , Q=> WaitforStabil, ET
IF WaitforStabil THEN
    Deposit.Status := E_DepositStatus.Stabil;
END_IF
```

Kuva 19. Prosessin aloitus ja siirtyminen lämmittämisestä stabilointivaiheeseen.

Siirtymisehtoina voitiin käyttää esimerkiksi laskevia tai nousevia reunoja sekä ajastimia. Prosessin eteneminen oli kirjoitettu yhteen POU:hun, joka keskustelee muun ohjelman kanssa, missä itse prosessin vaiheet tapahtuvat. Prosessin etenemistä hallinoiva ohjelman osa määrittelee, milloin mitkäkin tapahtumat tapahtuvat prosessissa.

4.4.5 Sarjaliikenne

Sarjaliikennettä varten tarvittiin Functions-paketti TF6340. Paketin avulla saadaan käyttöön TwinCAT:in kirjasto, josta löytyvät sarjaliikkenöintiin tarvittavat valmiit toimilohkot sekä muuttujat. Tässä projektissa sarjaliikenteellä ohjataan turboventtiiliä.

Kirjatosta valittiin ensin taustakommunikointia varten 6inData22B ja 6inData22B-struktuurit, jotka valittiin sarjaliikennekortin EL6002 mukaan. Nämä struktuurit linkitettiiin TwinCAT:issä oikeisiin kortin EL6002 tuloihin ja lähtöihin. Lisäksi luotiin TX-kommunikoinpuskuri datan lähettämiseen ja RX-kommunikointipuskuri datan vastaanottamiseen.

Seuraavaksi valittiin sarjaliikennekirjastosta toimilohko SerialLineControl hoitamaan kommunikointia PLC:n ja sarjaliikenneväylän välillä. Tähän asetettiin edellä mainitut sarjaliikennemuuttujat. Valmis toimilohko näytti seuraavalta (kuva 20.).

```
TurboComm(
    Mode:= SERIALLINEMODE_EL6_22B,
    pComIn:=ADR(Turbo.DataIN),
    pComOut:=ADR(Turbo.DataOUT),
    SizeComIn:= UINT_TO_INT(SIZEOF(Turbo.DataIN)),
    Error=> ERROR,
    ErrorID=> ERROR_ID,
    TxBuffer:= TxBuffer,
    RxBuffer:= RxBuffer);
```

Kuva 20. Projektissa käytetty SerialLineControl toimilohko

Seuraavaksi tarvittiin vielä toimilohkot sanomien lähettämiseen ja vastaanottamiseen. Vastaanottamiseen valittiin toimilohko ReceiveString ja lähettämiseen SendString.

ReceiveString-lohkoon (kuva 21.) tuodaan sarjaliikenteeltä saatu sanoma RX-kommukointipuskurilla ja saatu string-muotoinen viesti siirretään muuttujaan ReceivedString. Prefix tunnistaa viestin alun ja Suffix puolestaan tunnistaa viestin lopun. Viestit jotka eivät ala samoilla merkeillä kuin Prefix, jätetään huomiotta ja viestiä luetaan niin kauan, kunnes merkkijono vastaa samaa kuin Suffix. [21.]

```
Receive(
    Prefix:= 'Stx' ,
    Suffix:= 'Etx',
    Timeout:= ,
    Reset:= ,
    StringReceived=> Message_received,
    Busy=> Busy1,
    Error=> Error1,
    RxTimeout=> ,
    ReceivedString:= Received_String,
    RXbuffer:= RxBuffer );
```

Kuva 21. Projektissa käytetty ReceiveString toimilohko

SendString-toimilohko (kuva 22.) puolestaan kirjoittaa TX-kommukointipuskuriin sanoman, joka syötetään toimilohkon sisääntuloon SendString.

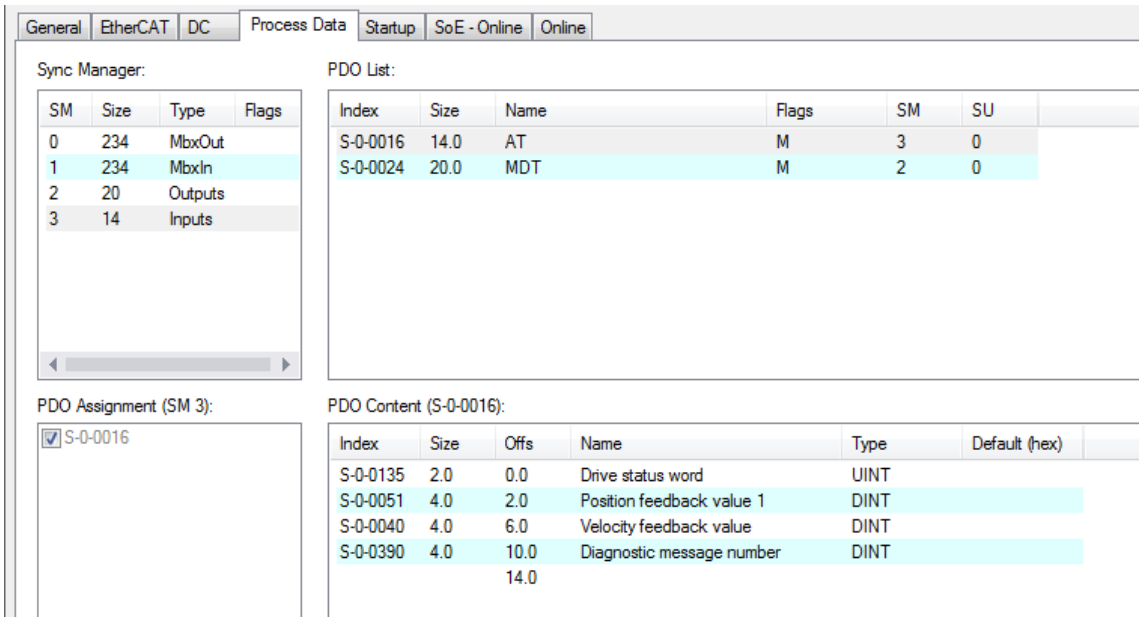
```
Send(SendString:= Send_String,
    Busy=>Busy2,
    Error=> Error2,
    TXbuffer:= TxBuffer);
```

Kuva 22. Projektissa käytetty SendString-toimilohko

Lähtettävien viestien eteen tuli myös lisätä Prefix ja Suffix. Sarjaliikenteen kommunikaatioperiaatteen mukaan komentojen lähettämisen lisäksi SendString-toimilohkoa käytetään myös kyselyjen tekemiseen. Kyselyjen avulla saadaan tietoja laitteelta, esimerkiksi tässä voidaan tiedustella turbomoottorin tilaa tai kierroslukuja.

4.4.6 EtherCAT-laitteet

XML-muotoisten ESI-tiedostojen ansioista voidaan TwinCAT:ssa nähdä, millaista dataa voidaan kirjoittaa ja vastaanottaa EtherCAT-laitteelta. Tarkastellessa laitetta projektin konfiguraatiossa päästään näkemään laitteen tila ja se, millaisia parametrejä valittiin laitteen ohjaamiseen. Process Data -välilehdellä voidaan valita, mitä laitteen tietoja halutaan lukea sekä kirjoittaa.

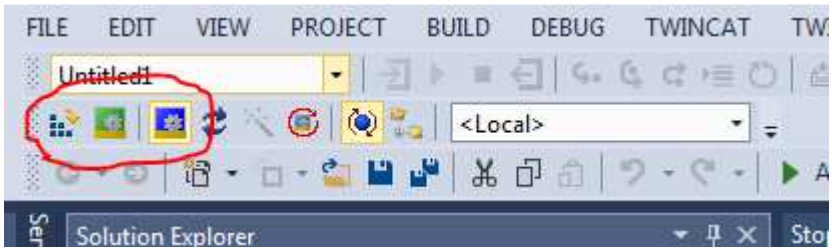


Kuva 23. Process Data -välilehti, jossa voidaan valita tarvittavat parametrit

Servomoottorin parametrilista oli hyvin pitkä ja parametrit valittiin koodin mukaan. Esimerkiksi paikkakomentoja moottorille oli useita erilaisia, joista piti valita oikeanlainen halutulle ohjaustyypille.

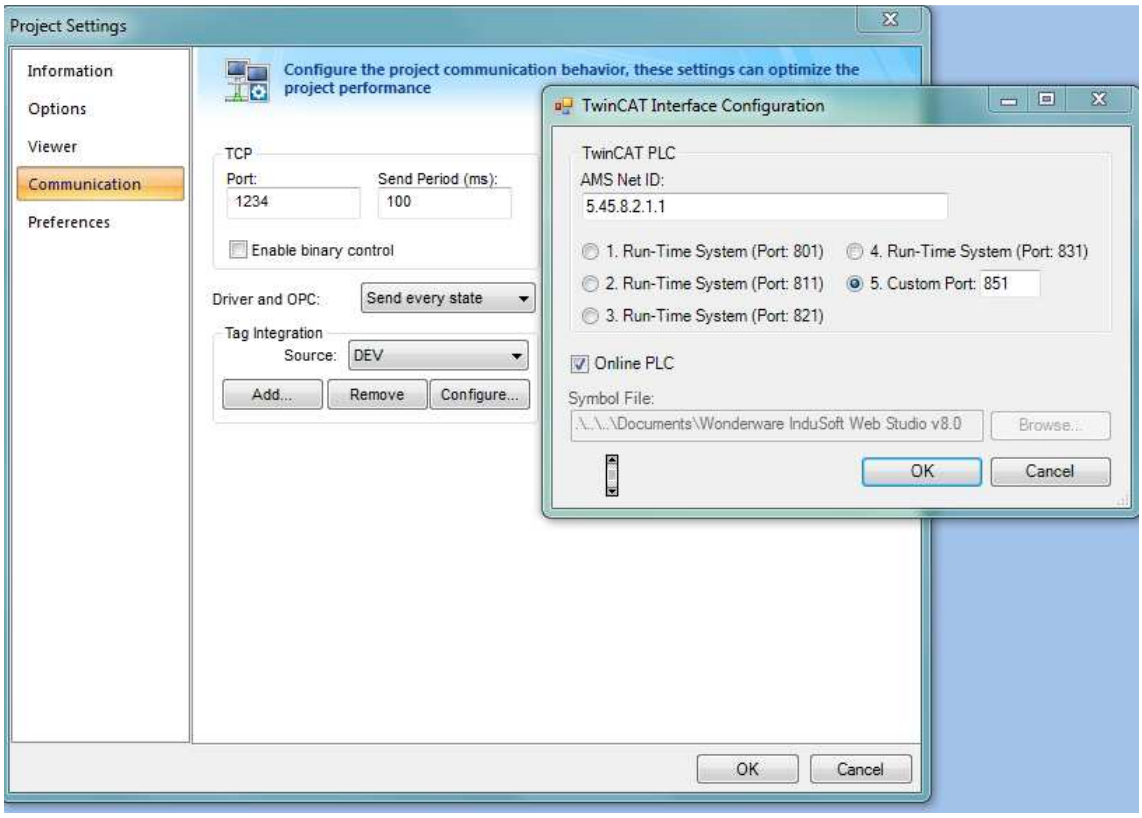
4.4.7 TwinCAT-muuttujien integrointi käyttöliittymään

Käyttöliittymä, johon TwinCAT:in muuttujat liitettiin, oli tehty Indusoft Web Studiolla. Käyttöliittymä pyörii samalla logiikalla, eli CX2020 sulautetulla PC:llä kuin TwinCAT. Jotta muuttujat voitiin liittää käyttöliittymään, täytyi TwinCAT:illä tehdyn ohjelman konfiguraation olla aktivoituna logiikalla ja ohjelman Run-tilassa (kuva 24).



Kuva 24. TwinCAT:in työkalurivissä: Aktivoi konfiguraatio, Käynnistä Run-tila ja Käynnistä Konfigurointi-tila

Indusoft Web Studiossa lisätään käyttöliittymäprojektiin TwinCAT Driver Sheet, joka huolehtii kommunikaatiosta ohjelmien välillä. Tämän jälkeen IWS:ssa avattiin Project Settings -valikon alta Communication-välilehti. Välilehdeltä löytyy Tag Integration, jossa voidaan lisätä ja sen jälkeen konfiguroida muuttujien lähteitä. TwinCAT:istä tuli tarkastaa sen AMS Net ID, joka voidaan katsoa joko ohjelmointitietokoneelta tai suoraan logiikalta, sekä portti jota käytetään.



Kuva 25. Indusoft Web Studion Commucations -välilehti, jossa voidaan konfiguroida TwinCAT:in muuttujia käyttöliittymäprojektiin.

Kun muuttujien lähde on konfiguroitu, voidaan IWS:n Object Finderissa, josta löytyy kaikki käyttöliittymäprojektin muuttujat, löytää kaikki TwinCAT-projektin muuttujat. Nämä muuttujat pitää kuitenkin vielä erikseen valita ja tuoda käyttöliittymäprojektiin Object Finderissa.

5 Yhteenveto

Työn tavoitteena oli suunnitella ohjelma UV-ALD-laitteelle TwinCAT 3 -ohjelmointiympäristössä. Lisäksi työhön kuului projektin automaatiojärjestelmän suunnittelu. Tavoitteena oli myös tutustua ohjelmointiympäristöön ja logiikkaan.

Työssä ohjelma saatiin suunniteltua toivotulla tavalla ja lisäksi ohjelmointimpäristöön päästiin tutustumaan intensiivisesti. Mielestäni onnistuin kehittämään ohjelmaa verrattaessa vanhaan, ja sain yhdisteltyä toimintoja niin, että ohjelma on modulaarisempaa ja selkeämpää. Käyttöönotossa vielä hiottiin hieman ohjelmaa, jotta saatiin kaikki toiminnallisuudet toimimaan juuri halutulla tavalla. Esimerkiksi

lämmönsäädön parametrejä pystyttiin säätämään vasta käyttöönotossa, kun pystyttiin näkemään, millaisen vasteen lämmittimet antoivat säätimelle. Käyttöönotto on opinnäytetyön kirjoitushetkellä projektissa kesken.

Uuteen ohjelmointiympäristöön tutustuminen vie aikaa, ja sellaisella toteutettu projekti vie kauemmin. Lisäksi ohjelmoija ei vältämättä osaa optimoida kaikkia käytettävissä olevia työkaluja, jos ne ovat sellaisia, mitä aiemmissa ohjelmointiympäristöissä ei ole ollut. Esimekiksi metodeita ei tässä työssä käytetty, ja niiden avulla saataisiin ohjelmaa vielä kehitettyä edistyneempään suuntaan.

Työn ollessa tekijälleen ensimmäinen ohjelmointiprojekti, joka toteuttettin täysin alusta loppuun, näkyy tämä myös toteutuksessa. Ohjelmaa tehdessä saatiin paljon ideoita, joista osa taas todettiin myös heti toimimattomaksi, jolloin ohjelmassa jotkin osiot tehtiin välillä täysin uudestaan.

Koska suunnittelu osittain pohjautui käytössä olevaan pohjaohjelmaan, eikä varsinaisesti laitteen toiminnallisiin kuvauksiin, suunnittelun eteneminen perustui ohjelmallisiin kokonaisuuksiin, eikä varsinaisesti toiminnallisuuksiin. Tällaisellä suunnittelun etenemisellä ohjelmasta ei välttämättä tule rakenteeltaan yhtä eheä, kuin jos ohjelmaa rakennettaisiin toiminnallisuuksien mukaan.

Jos pohditaan, millaista lisäarvoa Picosun Oy voisi saada Beckhoffin logiikkaan ja TwinCAT 3 -ohjelmointiympäristöön siirtymisestä, eräs tärkeä tekijä tässä on EtherCAT-väylä. Tätä työtä tehdessä koin, että laitteiden liitettävyys ohjelmaan ja kommunikointi helpotti ohjelmoijaa erittäin paljon, esimerkiksi verrattaessa sarjaliikenteeseen. Lisäksi väylän avulla saadaan kaapelointia vähennettyä, erityisesti jos tulevaisuudessa laitevalmistajilta löytyy EtherCAT P -liitäntämahdollisuus. EtherCAT P -kaapelointiratkaisussa tuodaan yhdellä kaapelilla väylä sekä virta laitteelle.

Lähteet

1 Picosun in brief. Verkkodokumentti. Picosun Oy. <http://www.picosun.com/en/about+us/picosun+in+brief/> Luettu 3.6.2016

2 Lehdistötiedote, verkkodokumentti. Picosun Oy. http://www.picosun.com/binary/file/-/id/41/fid/723/ Luettu 14.7.2016

3 What is ALD. Verkkodokumentti. Picosun Oy. <http://www.picosun.com/en/technology/what+is+ald/> Luettu 3.6.2016

4 Malm Jari, ”Vacuum Line Chemistry In Selected Atomic Layer Deposition Processes” Thesis for the degree of Master of Science in Technology. Espoo, 17 June 2005

5 Products. Verkkodokumentti. Picosun Oy. <http://www.picosun.com/en/products/> Luettu 8.7.2016

6 Suntola Tuomo, “Atomic Layer Epitaxy”, Microchemistry Ltd. (15.4.1989)

7 R. L. Puurunen, ”Surface chemistry of atomic layer deposition: a case study for the trimethylaluminum/water process”

8 H. B. Profijt, S. E. Potts, M. C. M. van de Sanden, and W. M. M. Kessels “Plasma-Assisted Atomic Layer Deposition: Basics, Opportunities, and Challenges” Department of Applied Physics, Eidhoven Univeristy of Technology (18.8.2011)

9 Steven M. George, “Atomic Layer Deposition: An Overview” , Chem. Rev. 2010,110(1), pp 111-131, DOI: 10.1021/cr900056b

10 Manuaali. Picosun Oy. “ALD R200 Advanced: Safety, Operating and Maintenance Manual”

11 R. L. Puurunen, ”A Short history of Atomic Layer Deposition: Tuomo Suntola’s Atomic Layer Epitaxy”, Chemical Vapor Deposition 2014, DOI 10.1002/cvde.201402012 Essay

12 Verkkodokumentti. Picosun Oy. <http://www.picosun.com/en/markets/> Luettu 30.8.2016

13 Verkkodokumentti. Beckhoff Automation. <http://infosys.beckhoff.com/index_en.htm> Luettu 5.8.2016

14 Manuaali, verkkodokumentti. Beckhoff Automation. <https://download.beckhoff.com/download/document/ipc/embedded-pc/embedded-pc-cx/cx2000_hwen.pdf> Luettu 14.9.2016

15 Verkkodokumentti. Beckhoff Automation. <https://download.beckhoff.com/download/document/catalog/Beckhoff_TwinCAT3_042012_e.pdf > Luettu 14.7.2016

16 Verkkodokumentti. Beckhoff Automation. <https://infosys.beckhoff.com/english.php?content=../content/1033/tc3_plc_intro/18014398645791371.html&id=> Luettu 26.4.2017

17 Verkkodokumentti. Beckhoff Automation. <http://www.beckhoff.fi> Luettu 5.8.2016

18 Verkkodokumentti. Microsoft. <https://www.microsoft.com/windowsembedded/en-us/products-solutions-overview.aspx> Luettu 20.9.2016

19 Verkkodokumentti. EtherCAT Technology Group. <https://www.ether-cat.org/en/technology.html#1.1> Luettu 2.4.2017

20 Hopponen Ville, 2016. Key Account Manager, Beckhoff Oy. Tapaaminen 16.6 ja 23.11.

21 Verkkodokumentti. Beckhoff Automation. <https://download.beckhoff.com/download/document/automation/twincat3/TF6340_TC3_Serial_Communication_EN.pdf> Luettu 16.4.2017

IO-lista

I/O list

IPC: CX2020

Rack 1(E-Bus):

1. CX2100-004 power supply unit			
2. TC: EL3318			
I/O	Name	Description	
1	TE1	Reaction chamber bottom plate	
2	TE2	Reactor chamber top plate	
3	TE11	Bottom plate heater temperature	
4	TE12	Bottom plate heater temperature	
5	TE13	Bottom plate heater temperature	
6	TE14	Bottom plate heater temperature	
7	TE15	Bottom plate heater temperature	
8	TE16	Bottom plate heater temperature	
3. TC: EL3318			
I/O	Name	Description	
1	TE17	Bottom plate heater temperature	
2	TE18	Bottom plate heater temperature	
3	TE21	Top plate heater temperature	
4	TE22	Top plate heater temperature	
5	TE23	Top plate heater temperature	
6	TE24	Top plate heater temperature	
7	-	-	
8	-	-	
4. TC: EL3318			
I/O	Name	Description	
1	TE110	Source A1 bottle temperature	
2	TE112	Source A1 valve block temperature	
3	TE113	Source A1 valve block heater temperature	
4	TE114	Source A1 neck heater temperature	
5	TE115	Source A1 valve block heater temperature	
6	TE116	Source A1 bottle heater temperature	
7	TE117	Source A1 bottle heater temperature	
8	TE120	Source A2 bottle temperature	
5. TC: EL3318			
I/O	Name	Description	
1	TE220	Source B2 bottle temperature	
2	-	-	
3	-	-	
4	-	-	
5	-	-	

6	-	-	
7	-	-	
8	-	-	

6. AI: EL3164			
I/O	Name	Description	
1	PT1	Main line pressure	0-10 V
2	PT2	Pump line pressure	0-10 V
3	PT11	PT11	0-10 V
4	PT12	PT12	0-10 V
7. AI: EL3164			
I/O	Name	Description	
1	PT100	Line 1 pressure	0-10 V
2	PT200	Line 2 pressure	0-10 V
3	PT221	Source B1 pressure	0-10 V
4	PT300	Line 3 pressure	0-10 V
8. AI: EL3164			
I/O	Name	Description	
1	-	-	
2	-	-	
3	-	-	
4	-	-	

9. DI: EL1808			
I/O	Name	Description	
1	-	-	
2	-	-	
3	-	-	
4		Lamp alarm	
5		Lamp status	
6	N11	Position 1	
7	N12	Position 2	
8	TS1	Reactor temperature switch	
10. DI: EL1808			
I/O	Name	Description	
1	VS1	Vacuum switch	
2	-	-	
3	OK2	Source heaters contactor	
4		L110 1.lowest level	
5		L110 2.→	
6		L110 3. →	
7		L110 4. highest level	
8	-	-	

11. DO: EL2808			
I/O	Name	Description	

1	PU110	Source A1 cooler	
2	PU120	Source A2 cooler	
3	PU220	Source B2 cooler	
4		Lamp ON/OFF	
5	-	-	
6	-	-	
7	-	-	
8	-	-	
	12. DO: EL2809		
1	V1	Main pumping valve	
2	V2	Soft pumping valve	
3	V3	Carrier & purge nitrogen inlet valve	
4	V7	Pump line selection	
5	V11	Direct vent valve	
6	V13	Carrier & purge argon inlet valve	
7	V14	Turbo protection flow	
8	VSH1	Shutter on	
9	VSH2	Shutter off	
10	VU1	Lamp purge N_2	
11	-		
12	-		
13	-		
14	-		
15	-		
16	-		
	13. DO: EL2809		
1	V110	Source A1 pulsing valve	
2	V111	Source A1 booster valve	
3	V120	Source A2 pulsing valve	
4	V210	Source B1 pulsing valve	
5	V212	Source B1 safety valve	
6	V220	Source B2 pulsing valve	
7			
8	HE113	Source A1 valve block heater	
9	HE114	Source A1 neck heater	
10	HE115	Source A1 valve block heater	
11	HE116	Source A1 bottle heater	
12	HE117	Source A1 bottle heater	
13	-		
14	-		
15	-		
16	-		
	14. EL9410 E-Bus Power Supplier		
	15. DO: EL2809		
1	HE11	Bottom plate heater	
2	HE12	Bottom plate heater	
3	HE13	Bottom plate heater	

4	HE14	Bottom plate heater	
5	HE15	Bottom plate heater	
6	HE16	Bottom plate heater	
7	HE17	Bottom plate heater	
8	HE18	Bottom plate heater	
9	HE21	Top plate heater	
10	HE22	Top plate heater	
11	HE23	Top plate heater	
12	HE24	Top plate heater	
13	-	-	
14	-	-	
15	-	-	
16	-	-	
	16. DI : EL1889		
1	FS1	Flow switch	
2		CDA pressure alarm	
3		CDA pressure warning	
4	-	-	
5	-	-	
6	-	-	
7	-	-	
8	-	-	
9	-	-	
10	-	-	
11	-	-	
12	-	-	
13	-	-	
14	-	-	
15	-	-	
16	-	-	
	17. Sarjaliikenne (RS232) EL6002		
1		Eximer lamp	
2		Turbo pump	Edwards STPH301C
	18. EL6070 License key terminal		
	19. EtherCAT extension terminal EK1110		

EtherCAT:iin kytketään:

-MFC:t (Bronkhorst EL-Flow Select F-201CV)

-V6 (VAT 65040-PA [G][I])

-Servo Drive (Rexroth MSM031B)

ELI KORTIT

IPC	**CX2020**

Term	Rack 1 (E-bus)
1	CX2100-004
2	EL3318
3	EL3318
4	EL3318
5	EL3318
6	EL3164
7	EL3164
8	EL3164
9	EL1808
10	EL1808
11	EL2808
12	EL2809
13	EL2809
14	EL9410
15	EL2809
16	EL1889
17	EL6002
18	EL6070
19	EK1110
20	