Tampereen ammattikorkeakoulu

Toimitilojen rakennusautomaatio

Suunnittelu, määrittely ja ohjelmointi

Piirto, Ville

OPINNÄYTETYÖ
Toukokuu 2020

Sähkö- ja automaatiotekniikka
Automaatiotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Sähkö- ja automaatiotekniikka
Automaatiotekniikka

Piirto, Ville
Toimitilojen rakennusautomaatio
Suunnittelu, määrittely ja ohjelmointi

Opinnäytetyö 74 sivua, joista liitteitä 9 sivua
Toukokuu 2020

Opinnäytetyön tarkoituksena oli toteuttaa toimiva rakennusautomaatio yhteistyöyrityksen uusiin toimitiloihin. Työhön kuuluivat määrittely, suunnittelu sekä komponenttivalintojen että ohjelmiston suhteen, varsinainen ohjelmointi sekä käyttöliittymän toteutus.

Ohjelman tarkoituksena oli tehdä toimitiloissa työskentelystä mahdollisimman vaivatonta hallitsemalla ja ohjaamalla valaistusta, lämmitystä, ilmanvaihtoa ja lukuisia muita rakennusautomaatioon liittyviä komponentteja automaattisesti. Yrityksen edellisissä toimitiloissa oli lukuisia kohteita, jotka vaativat manuaalista sammutusta. Nyt nämä kohteet sammutettaisiin automaattisesti, kun viimeinen työntekijä poistuisi työpaikalta, ja edelleen laitettaisiin päälle aamulla ensimmäisen saavuttua paikalle.

Vaikka työn osuus kokonaisuutena arvioiden oli onnistunut, sen laajuuden vuoksi monia asioita ja mahdollisuuksia jäi tutkimatta, ja mm. DALI-väylällä toteutettu valaistus olisi hyvinkin voinut yhtenä kokonaisuutena olla kokonaisen lopputyön täyttävä suoritus. Nyt asiat toteutettiin helpoimman kautta, joka tulevaisuuden modifiointeja silmällä pitäen voi olla hankalaa. Väylä olisi tarjonnut myös mahdollisuuksia toteuttaa automaattista vianhakua, jolloin käyttöliittymään olisi voinut tulla tietoa vaikkapa viallisista valaisimista. Toisaalta tällaiset vialliset valaisimet voidaan verrattain pienessä tilassa havaita myös visuaalisesti, eikä työn tilaajalla ollut vaadetta toimintoon.

Huomionarvoisin seikka oli kuitenkin Android-tablettien lopullinen soveltumattomuus HMI:ksi. Androidista on – ymmärrettävistä syistä – tehty hyvin suljettu järjestelmä. Tämä estää yhtäältä pahantahtoisten laitteen kaappaajien toiminnan, mutta toisaalta laitteen modifioinnin teollisuuteen sopivaksi HMI:ksi. Erityisesti virransäästöominaisuudet ja niiden muokkaamisen estäminen tekevät tabletin käytöstä hyvin hankalaa. Ja jos tabletti saataisiinkin toimimaan halutun kaltaisesti, ei mikään lupaa, etteikö jonkin tulevaisuuden Androidin ohjelmistopäivityksen yhteydessä näitä käytettyjä metodeita otettaisi pois toiminnasta.

Asiasanat: rakennusautomaatio, HMI, PLC, Beckhoff, automaatiosuunnittelu

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Automation and Electrical Engineering
Automation Engineering

Ville Piirto:
Building Automation of Premises
Design, Specification and Programming

Bachelor's thesis 74 pages, of which appendices are 9 pages
May 2019

The purpose of this thesis was to make a working building automation to the premises of the co-operative company. The task consisted of specification, design, programming and the making of the user interface to control the automation.

The purpose of the program was to make working in the building as effortless as possible by automatically controlling the lighting, heating, air conditioning and numerous other components of the building automation. In the previous premises of the company there were many targets that had to be manually turned off. Now these targetts would be automatically turned off when the last employee would leave the premises, and in turn be turned on when the first one arrived in the morning.

Although the task in general was successful, the magnitude of it left many possibilities unprobed, and e.g. the lighting which was made with the DALI bus could've easily been a subject of a thesis alone. Now everything was executed as easily as possible, which by taking possible future modifications into account may bring difficulties. The bus would've also made automatic diagnostics possible, which would've in turn be seen on the alarm list of the HMI. On the other hand, the relatively small size of the premises makes it entirely possible to search for faulty light sources visually, and the employer of the thesis didn't demand for such feature.

The most noteworthy aspect was the unfittedness of Android tablets as HMI terminals. Android is, for understandable reasons, a very closed system. On the other hand, it prevents malicious hijackers from tampering with the device, but on the other hand it also prevents the modification of the device as a suitable device for an industry HMI. Especially the power saving options make the usage of the tablet very difficult. And even in the case that the tablet could be configured sufficiently, there's nothing that would prevent future Android software updates from disabling the configurations made.

Key words: building automation, PLC, HMI, Beckhoff, automation design

SISÄLLYS

LYHENTEET JA TERMIT

kernel	Käyttöjärjestelmän ydin. Toimii suojatussa muistiavaruudessa, joka estää sen muiden ohjelmien ylikirjoituksen.
PLC	*Programmable Logic Controller*. Logiikkaohjain, joka toteuttaa tehtyä ohjelmaa syklisesti.
HMI	*Human Machine Interface*. Käyttöliittymä ihmisen ja PLC:n välillä.
I/O-piste	Järjestelmän fyysinen mitta- tai ohjauspiste.
struktuuri	Tietotyyppi, joka sisältää joukon muuttujia.
enumeraatio	Tietotyyppi, joka muuttaa kokonaisluvut selkokielisiksi muuttujiksi.

1 JOHDANTO

Opinnäytetyön tarkoituksena oli toteuttaa toimiva rakennusautomaatio yhteistyöyrityksen, Metecno Oy:n uusiin toimitiloihin. Työhön kuuluivat määrittely, suunnittelu sekä komponenttivalintojen että ohjelmiston suhteen, varsinainen ohjelmointi sekä käyttöliittymän toteutus.

Ohjelman tarkoituksena oli tehdä toimitiloissa työskentelystä mahdollisimman vaivatonta hallitsemalla ja ohjaamalla valaistusta, lämmitystä, ilmanvaihtoa ja lukuisia muita rakennusautomaatioon liittyviä komponentteja. Pyrkimyksenä oli, ettei esimerkiksi valaistusta tarvitsisi edes miettiä, vaan se toimisi automaattisesti. Myös joistakin asioista, kuten kompressorien sammutuksista pois lähtiessä, jotka edellisissä toimitiloissa oli jouduttu tekemään manuaalisesti, päästäisiin eroon. Tällöin mm. paloturvallisuus parantuu, eikä viimeisenä työpaikalta lähteneen työntekijän tarvitse miettiä, onko hän muistanut sammuttaa kaiken mahdollisen.

Sekondaarisena tarkoituksena oli tuottaa yritykselle lisätietoa yrityksen omasta PID-säätimestä ja Beckhoffin uudehkosta, HTML5-pohjaisesta HMI-järjestelmästä sekä siitä, sopisivatko Android-tabletit käyttöliittymän päätteiksi.

Työ toteutettiin Beckhoffin komponenteilla, joista I/O-terminaalit olivat KL-sarjaa. Valaistus toteutettiin DALI-väylällä sen ohjattavuuden monipuolisuuden vuoksi, ja energiamittareiden lukuun käytettiin M-Bus -väylää.

Opinnäytetyö on laajuutensa vuoksi vahvasti suoritteellisuutta painottava, eikä sisällä teoriaa syvemmällä tasolla esim. PLC:n ja HMI:n kommunikointikeinoista taikka DALI- tai M-Bus -väylistä.

2 TEORIA

2.1 Beckhoff PLC

Beckhoffin PLC:t ovat Windows-pohjaisia tietokoneita, joihin asennetaan logiikkaohjelmaa suorittava TwinCAT. Toimenpide asentaa Windows-pohjaiseen koneeseen oman kernelinsä, joka varaa prosessorilta ytimen itselleen (Beckhoff, 481). Tietotekniikassa kernelillä tarkoitetaan käyttöjärjestelmän keskeistä osaa (Yuichi, 13). Se on tietyllä tapaa nimensä mukaisesti käyttöjärjestelmän ydin, joka toimii välittäjänä sovellusten ja fyysisten komponenttien välillä (Roch,1). Koska PLC:t ovat Windows-pohjaisia, ohjelmointiympäristön mukana asentuvaa TwinCAT-kerneliä voidaan pyörittää myös ohjelmoivalla PC:llä. PLC-ohjelmaa voidaan siis suorittaa ilman PLC:tä, kunhan kohdelaitteeksi on määritelty *<Local>*.

2.2 Beckhoff HMI

Vaikka Beckhoffin HMI on aikaisemminkin PC-pohjaisuutensa vuoksi mahdollistanut tavallisen näytön käytön käyttöliittymän päätteenä, on ongelmana ollut sen paikallisuus. Etäkäyttöohjelmat ovat toki vaihtoehto, mutta ne eivät ole useinkaan tietoturvallisia ja voivat olla hankalia käyttää. Beckhoff yrittää päästä tästä eroon siirtymällä HTML5-protokollaan pohjautuvaan järjestelmään. Mikä hyvänsä tällä pohjalla tehty HMI-sovellus aukeaa millä tahansa web-selaimella, jolla vain on käyttöoikeus siihen verkkoon, jossa HMI-serveri on käynnissä.

Teoriassa halvinkin tablettitietokone soveltuu HMI-päätteeksi, jos se vain kestää ympäristön rasitukset. Teollisuusympäristöissä olosuhteet määräävät pitkälti, millaisia laitteita sinne voidaan ottaa käyttöön. Kuitenkin esimerkiksi tässä kyseisessä työssä toimiston pöydällä lojuva tabletti, jolla silloin tällöin laitetaan valoja päälle ja pois, tai toisinaan säädetään huonelämpötilaa, ei vaadi suuren luokan kestävyyttä.

2.3 Rakennusautomaatio yleisesti

Rakennusautomaatio juontaa juurensa 1980-luvun loppupuolelle, jolloin markkinoille ilmestyi mikroprosessoriohjaimia (Sands, Verhappen, 532). Näitä ohjaimia käytetään rakennusautomaatiossa edelleen yleisesti, sillä ne ovat käyttötarkoitukseensa spesifisti suunniteltuja, mikä tekee niistä halpoja. Tässä työssä tällaisten käyttöön ei kuitenkaan päädytty, sillä ne eivät mahdollista monimutkaista laskentaa. Tätä haluttiin joka tapauksessa mm. energiamittausten suhteen, ja koska Beckhoff tarjoaa rakennusautomaatioon sopivia komponentteja, laskettiin sen tulevan halvemmaksi.

Rakennusautomaatiota käytetään pääasiassa neljästä syystä: ylläpitokustannusten lasku, kommunikaatioväylät informaation vaihtoon, ihmisten mukavuus ja joustavuus. Kustannusten lasku koostuu pääasiassa valaistuksen ja lämmityksen säädöstä. Kustannusten laskua voidaan perustella myös tutkimuksilla, joiden mukaan ihmisten tuottavuus tippuu huomattavasti, mikäli työympäristö ei ole optimaalinen, eli lämpöä on joko liikaa tai liian vähän. Joustavuutta tuo ohjelmoitavuus, jolloin vaikkapa toimistokäyttöön tarkoitetun rakennuksen pohjapiirustusta voidaan muuttaa. Toimistoja voidaan siis yhdistää tai erottaa. (Merz et al, 2-3.)

2.4 Digital Addressable Lighting Interface, DALI

DALI on rakennusautomaatiossa käytetty valaistuksen ohjausväylä, joka on määritelty standardissa IEC 60929. Se on isäntä-renki -tyyppinen järjestelmä, jossa kommunikointi isäntä- ja renkilaitteen välillä tapahtuu molempaan suuntaan kahta johdinta pitkin. Järjestelmä on digitaalinen ja sen looginen 1 on yli 9,5 V (yläraja 22,5 V) sekä looginen 0 alle 6,5 V (alaraja -6,5 V). (Yuejun at al, 1106-1107.)

DALI-väylään liitettäviä valaisimia voidaan kontrolloida joko valaisin kerrallaan, tai ne voidaan jakaa ryhmiin. Koska käskyt valaisimille menevät digitaalisena viestinä, katkaisijoille ei tarvitse asentaa 230 V sähköä. Tämä säästää jo asennuskuluissa, sillä valaisimille voidaan vain viedä sähköt, eikä asentajan

tarvitse miettiä, miten ja millaisissa ryhmissä valaisimien tulee toimia. DALI-valaisimilta saadaan myös diagnostiikkatietoa takaisin – mm. onko valaisin päällä, millä voimakkuudella, tai onko se vioittunut. (Hein, 901.)

3 SUUNNITTELU

3.1 Hallin sähköinen rakenne

Toimitilan sähkökeskukset oli jaettu sähkösuunnitteluvaiheessa viiteen eri keskukseen: kolmeen alakerrassa, yhteen yläkerrassa ja yhteen hallin ilmanvaihtoparvella. Näistä alakerrassa olivat pääkeskus (PK), alakerran taukotilojen ja pukuhuoneiden jakokeskus (JK102) ja yksi teknisen tilan jakokeskus (JK101). Yläkerrassa sijaitsivat yläkerran toimistojen ja neuvottelutilojen sähköjen jakokeskus (JK201), hallin ilmanvaihtoparvella hallin ilmanvaihdon sähköt (IVK). Jokaiseen sähkökeskukseen tulisi sen alueen tarvittavat automaation komponentit. Keskukset oli yhdistetty ethernet-verkolla.

3.2 Määrittely

Tarvittavien I/O-pisteiden määrittely aloitettiin määrittelemällä sähkösuunnittelijan tekemistä pää- ja piirikaavioista. Määriteltiin kuinka monta ja minkä tyyppistä ulos- ja sisääntuloa tarvittaisiin mihinkin keskukseen ja mitä jänniteluokkaa ohjaukset olisivat. I/O-taulukko on esitetty liitteessä 1.

3.2.1 Pääkeskus PK

Pääkeskuksessa binääristä ohjattavaa oli kaksi sähköauton latauspistettä, hallin paineilmajärjestelmän kompressori, kaksi hallin nosto-ovea, hallin siltanosturi ja etuoven betonilaatan ja vesirännien sulanapitolämmitys. Näille varattiin jokaiselle sekä yksi binäärinen ulostulo (DO-piste) että yksi binäärinen sisääntulo (DI-piste) vikamonitoroinnin vuoksi. Pääkeskuksesta ohjattiin myös hallin kuutta ohjattavaa lattialämmityspiiriä. Lisäksi pääkeskuksesta ohjattaisiin hallin ja ulkoseinien valoja DALI-väylän kautta.

Pääkeskukseen tarvittiin siis 7 kpl 24 V DI-pistettä, 13 kpl 24 V DO-pistettä ja DALI-ohjainkortti. Yhdessä DALI Master -laitteessa voi olla kytkettynä 64

toimilaitetta kanavaa kohden (Beckhoff, 8), joten koska ohjattavien valojen kokonaismäärä jäi tämän alle, voitiin tyytyä yhteen DALI-korttiin.

3.2.2 Hallin ilmanvaihtokeskus IVK

IVK:hon kytkettynä oli kaksi hallin ilmanvaihtokonetta: tulo- ja poistoilmalle. Näistä kummankin puhaltimet tehtiin taajuusmuuttajaohjatuiksi, jotta ilmanvaihdon tehokkuutta voitaisiin säätää. Kummallekin pumpulle tarvittiin siis yksi DO-piste taajuusmuuttajan aktivointisignaalia varten, yksi DI-piste taajuusmuuttajan vikasignaalille ja yksi analoginen ulostulo (AO-piste) puhaltimien kierrosmäärän kontrollointia varten.

Kummassakin koneessa oli myös venttiili ilmatien sulkemiseen, joilta saatiin potentiaalivapaa kärkitieto niiden auki-/kiinni-tilasta. Näille venttiileille tarvittiin siis yhteensä neljä DI-pistettä. Venttiilit avautuivat 230 VAC -jännitteellä ja sulkeutuivat jousipalautteisesti, joten niiden aukaisuun tarvittiin yhteensä kaksi DO-pistettä.

Tuloilman koneella oli myös lämmityspatteri, jolla oli mahdollista lämmittää sisääntuloilmaa, ja poistoilman koneella lämmön talteenoton järjestelmä. Molemmille oli omat pumppunsa, joille tarvittiin kummallekin yksi DI- ja yksi DO-piste. Lisäksi tulevaa ilmaa mitattiin NTC10K-anturilla, jotta järjestelmä voitaisiin sammuttaa jäätymisen estämiseksi, sekä suodatinvahti, joka mittaisi paine-eroa suodattimen tulo- ja lähtöpuolen välillä.

IVK:hon tarvittiin siis yhteensä 8 kpl 24 V DI-pistettä, 4 kpl 24 V DO-pistettä, 2 kpl AO-pistettä, 2 kpl potentiaalivapaata DO-pistettä, 1 kpl vastusanturin tulopistettä ja 2 kpl sisääntulopistettä ilmanpaineelle.

3.2.3 Teknisen tilan jakokeskus JK101

JK101 sijaitsi hallin teknisessä tilassa, johon oli sijoitettu lämminvesivaraaja, maalämpöpumppu ja alakerran taukotilojen ilmanvaihtokone sekä energiamittarit. Myös lattialämmityksen vesikierron pumput sijaitsivat teknisessä tilassa.

Maalämpöpumpulla oli oma valmistajan toimittama ohjaimensa, joten sille tarvittiin vain neljä DI-pistettä: yleinen vika, kaksi paisuntaventtiilien ohjainten vikaa ja lisälämpöpyyntö, jonka perusteella ohjattaisiin lämminvesivaraajan sähköllä toimivia lämpövastuksia siinä tapauksessa, ettei maalämpö itsessään riittäisi. Vastuksia ohjaavalta kontaktorilta otettiin myös käyntitieto, joten tarvittiin yksi DI-piste. Maalämpöön liittyy myös maakylmä, jolla voitaisiin viilentää ala- ja yläkerran tiloja. Näitä varten tarvittiin kolme DO-pistettä: päämaaliuospumpulle, sekä ylä- ja alakerran viilennyspiirin pumpuille. Viilennyspiirien pumpuilta haluttiin myös käyntitieto, joten tarvittiin kaksi DI-pistettä. Maalämpöpumpun lauhdevesipumpuille tarvittiin myös kaksi DI-pistettä niiden vikasignaaleille.

Lattialämmityspiirit oli jaettu kahteen osaan: hallin piireihin sekä ala- ja yläkerran tilojen piireihin. Lattialämmityspiireille oli siis kaksi pumppua, joiden vikasignaaleille tarvittiin kaksi DI-pistettä. JK101:een oli johdotettu myös kahden pukuhuoneen ohjattavat lattialämmityspiirit, joille tarvittiin kaksi DO-pistettä.

Ilmanvaihtokone alakerrassa poikkesi hallin ilmanvaihdosta. Sille oli vain yksi kone, johon oli valmiiksi ohjelmoituna neljä eri nopeutta. Näiden nopeuksien ohjaus annettiin koneelle binäärisenä tietona. Potentiaalien erottamiseksi nämä ohjaukset päätettiin antaa potentiaalivapaana kärkitietona. Koska ensimmäinen nopeus olisi aina päällä koneen saadessa sähkönsyötön, tarvittiin kolme potentiaalivapaata DO-pistettä. Hälytyksiä koneelta saatiin kaksi: huoltovaroitukset ja kriittiset, koneen pysäyttävät hälytykset. Näille tarvittiin siis kaksi DI-pistettä. Kaikkien IV-koneiden hätäseis sijaitsi myös teknisessä tilassa, ja sen tilan tarkkailua varten varattiin yksi DI-piste.

Keskukselle oli johdotettu myös potentiaalivapaat tilatiedot rikos-/paloilmoitinjärjestelmän päällä olosta, vikatilasta ja hälytyksestä. Myös

kulunvalvontajärjestelmän tila- ja vikatiedot tulivat kyseiselle keskukselle. Näille tarvittiin siis neljä DI-pistettä. Hallin syöttöveden sulkuventtiiliä ohjattiin rikosilmoitinjärjestelmän tilatiedon avulla, joten sille tarvittiin yksi potentiaalivapaa DO-piste.

Kaikki hallin kuusi energiamittaria sijaitsivat teknisessä tilassa. Kaikki mittarit kommunikoivat M-Bus -väylällä, joten sitä varten tarvittiin yksi M-Bus -kommunikaatiopiste. Standardin mukaan yhteen M-Bus -linjaan voi kytkeä 250 toimilaitetta (Beckhoff, 4), mutta yhteen M-Bus -masterkorttiin korkeintaan 40 (Beckhoff, 5). Koska määrä jäi kuitenkin selvästi jälkimmäisenkin luvun alle, voitiin tyytyä yhteen terminaaliin.

JK101:een tarvittiin siis 16 kpl 24 V DI-pistettä, 6 kpl 24 V DO-pistettä, 4 kpl potentiaalivapaata DO-pistettä ja M-Bus -kommunikaatiopiste.

3.2.4 Alakerran jakokeskus JK102

Alakerran ohjattavien lämmityspiirien venttiilit oli johdotettu tähän keskukseen. Niitä varten tarvittiin viisi DO-pistettä. Kaikki alakerran, halli mukaan lukien, huoneiden lämpötila-anturit oli johdotettu myös kyseiseen keskukseen, joten niitä varten tarvittiin kahdeksan vastusanturitulopistettä.

Jäähdytyskonvektoreita alakerrassa oli kolme: yksi työnjohdon huoneessa ja kaksi keittiössä. Näiden tilatietoon maaliuospumpun ohjaamista varten tarvittiin kolme 230 V DI-pistettä. Alakerran taukotilojen valaistusta ohjattiin DALI-väylällä, joten tarvittiin yksi DALI-kommunikaatiokortti.

JK102:een tarvittiin siis 6 kpl 24 V DO-pistettä, 8 kpl vastusanturipistettä, 3 kpl 230 V DI-pistettä ja DALI-kortti.

3.2.5 Yläkerran jakokeskus JK201

Yläkerran ohjattavien lattialämmityspiirien venttiileille tarvittiin kuusi DO-pistettä. Huonelämpötilan vastusantureille tarvittiin neljä vastusanturin tulopistettä. Myös ulkolämpötilan mittaus suoritettiin NTC10-tyypin anturilla, joten vastusanturin tulopisteitä tarvittiin viisi. Toimistojen tietokoneet oli kaikki kytketty UPS-järjestelmään, jonka hälytys- ja tilatiedoille tarvittiin kaksi DI-pistettä. Myös valvontakamerajärjestelmän vikatieto oli johdotettu kyseiseen tilaan, ja tarvitsi yhden DI-pisteen.

Yläkerrassa jäähdytyskonvektoreita oli viisi kappaletta, joiden tilatietoihin tarvittiin viisi 230 V DI-pistettä. Kaikki yläkerran valot ohjautuivat DALI-väylän kautta, joten tarvittiin yksi DALI-kommunikaatiopiste. Ulkovaloisuuden mittarille tarvittiin yksi 0…10 V AI-piste. Yläkerran ilmastointi suoritettiin samanlaisella koneella kuin alakerran taukotilojen, joten sille tarvittiin samat pisteet: kolme potentiaalivapaata DO-pistettä ja kaksi DI-pistettä. Näiden pisteiden tarkoitukselle on annettu tarkempi selvitys luvussa 2.2.3.

JK201:een tarvittiin siis 6 kpl 24 V DO-pistettä, 5 kpl 24 V DI-pistettä, 5 kpl vastusanturin tulopistettä, 5 kpl 230 V DI-pistettä, 3 kpl potentiaalivapaata DO-pistettä, 1 kpl 0…10 V AI-piste sekä M-Bus- ja DALI-kommunikaatiopisteet.

3.3 Komponenttivalinnat

Koska yritys, jolle työ tehtiin, suosii monissa kohteissa Beckhoffin logiikoita, työ tehtiin Beckhoffin laitteistolla. Toinen Beckhoffia puoltava tekijä oli KL-sarjasta suoraan löytyvät DALI- ja M-Bus -ohjainterminaalit. Täten DALI-väylällä kommunikoivia laitteita varten ei tarvittu erillistä rajapintaa PLC:n ja DALI-ohjaimen välillä. M-Busia on mahdollista ohjata Modbusin kautta, mutta Beckhoffin M-Bus -kirjastosta löytyvät valmiit toimilohkot helpottivat ja nopeuttivat työtä merkittävästi. Näiden toimilohkojen kanssa on käytettävä Beckhoffin omaa M-Bus -ohjainterminaalia. Beckhoffin valikoimasta löytyy myös kattava valikoima erilaisiin tarkoituksiin sopivia I/O-yksiköitä. Sisääntuloterminaaleista löytyy versioita, joissa on jokaiselle sisääntulopisteelle omat liittimensä 24 VDC ja 0 V, mikä vähentää riviliittimien tarvetta sähkö- ja ohjauskeskuksissa. Beckhoff tarjoaa myös releterminaaleja potentiaalivapaiden ulostulojen aikaansaamiseksi.

Teoriassa fyysisessä sovelluksessa ei välttämättä tarvita muita kuin Beckhoffin omia komponentteja, joskin kontaktorin korvaaminen kahdella (kaksi kanavaa per terminaali) Beckhoffin terminaalilla tulee hintavaksi.

Työn teettäjä valitsi järjestelmän pääohjainlaitteeksi sulautetun teollisuus-PC:n CX5130.

Koska toimitilojen sähkökeskukset on hajautettu viiteen eri paikkaan hallia, tarvittiin niiden välille verkko. Lisäksi koska kyseisessä työssä käytettiin nimenomaisesti Beckhoffin KL-sarjaa, johtuen DALI- ja M-Bus -väylistä, täytyi verkko toteuttaa ethernet-protokollaa käyttäen. Beckhoff tarjoaa tähän päätteitä BK9XXX-sarjasta. Kolmeen keskukseen (JK101, JK102, IVK) valittiin yksikanavainen BK9050. Näille verkko yhdistettiin JK201:een sijoitetulla ethernet-kytkimellä CU2008. Keskusten sijoittelusta johtuen PK päätettiin varustaa päätteellä BK9100, jossa on sisäinen ethernet-kytkin. PK:lta verkkoa siis jatkettiin IVK:lle tätä kytkintä apuna käyttäen.

Suurin osa järjestelmän binäärisistä sisääntuloista oli potentiaalivapaita tilatietoja, joten DI-kortiksi valittiin KL1808, joka on kahdeksankanavainen kortti, jonka 16:sta liittimestä kahdeksan ovat sisääntulopisteitä ja kahdeksan +24 VDC -liittimiä. Myös binäärisistä ohjauksista suurin osa oli joko potentiaalivapaita ohjauksia tai lattialämmityspiirien venttiilien ohjauksia, joten DO-kortiksi valikoitui KL2808, joka tarjoaa kaikille kahdeksalle ulostulokanavalle myös liittimen 0 V.

Potentiaalivapaisiin binääriohjauksiin valittiin relekortti KL2622, joka sopii myös 230 V ohjauksiin. Korttia valitessa tuli olla tarkkana, sillä Beckhoff tarjoaa myös kortteja moottoriohjauksien suunnanvaihtoihin, joiden kanavat ovat ’mutually locked’, kuten KL2732. Tässä kortissa vain yksi kanava voi olla päällä yhtäaikaisesti, joten esimerkiksi tämän työn taajuusmuuttajaohjauksiin kyseinen kortti ei sopinut. 230 V sisääntuloihin valittiin nelikanavainen KL1704, jolta saa myös syötön 230 VAC. Sähkön syöttökortiksi näille valittiin KL9150.

Analogiakorteista vastusantureille valittiin kahdeksankanavainen KL3208-0010, joka osaa tulkita NTC10-antureita. Ulkovaloisuuden anturi antoi signaalinaan 0…10 VDC, ja siihen liittyi myös saman signaalinen lämpötilasignaali, joten

valittiin kortiksi kaksikanavainen KL3062. Hallin ilmanvaihtokoneiden puhaltimien pyörittämiseen valittiin kahden taajuusmuuttajan takia kaksikanavainen KL4002.

DALI-kommunikaatiokortiksi valikoitui KL6821, M-Bus -väylän kommunikointiin valittiin KL6781. IVK:ssa suodatinvahdin ilmanpainetulojen mittakortiksi valittiin kaksikanavainen KM3712. Tieto siirtyy KL-sarjassa BK9XXX-päätteiden tai CXXXX-päätteiden ja terminaalien välillä K-Bus -väylää pitkin. Jokainen korttirivi tarvitsee myös K-Bus -väylän päättökortin, joka KL-sarjassa on KL9010.

4 OHJELMOINTI

4.1 Hyvä ohjelmointitapa

Mitä tahansa ohjelmaa tehdessä tulisi muistaa, että ohjelma voi päätyä aikojen saatossa kenen tahansa käsiin. Erityisesti puhtaan ohjelmistotekniikan alalla lähdekoodin tekijä ei välttämättä ole sen käyttäjä tai ylläpitäjä. Myös koneita ohjelmoidessa tulisi muistaa, että kone voidaan myydä minne ja kelle vain, ja sen kunnossapidosta vastaavilla henkilöillä voi olla tarvetta tehdä ohjelmaan muutoksia tai vaikkapa testata sisään- ja ulostulojen toimintaa, joka on helpointa suoraan ohjelmasta käsin. Myös tässä työssä tehdyssä rakennusautomaatiossa oli otettava huomioon, ettei ohjelman tekijä välttämättä ole sen ylläpitäjä koko rakennuksen elinkaarta.

Ohjelman tulee olla siis kaikilta osiltaan mahdollisimman selkeä, jotta siinä tapauksessa, että jonkun muun tarvitsee saada siitä selvää, sen toiminnan selvittämiseen ei mene turhan paljon aikaa. Tämän toteuttaminen alkaa ohjelmarakenteen suunnittelusta. Minkä hyvänsä kokonaisuuden voi jakaa aina pienempiin toiminnallisiin osiin, jotka voi jälleen jakaa pienempiin toiminnallisiin osiin, kunnes ollaan pienimmässä tekijässä, jota ei enää voida jakaa.

Tässä työssä kokonaisuus on ensin jaettu seuraaviin toiminnallisiin osiin: ilmastointi, DALI-väyläkommunikaatio ja valaistus, lämmitys, ulkovaloisuuden mittaus, energiamittareiden kommunikointi M-Bus -väylällä, turvatoiminnot, persistentti data, HMI-kommunikaatio sekä hälytykset. Näistä taas esimerkiksi M-Bus -kommunikaatio on jaettu seuraaviin osiin: kommunikaatio, mittareiden luvun sekvenssi, tarvittavien arvojen laskenta ja niiden kirjoitus taulukoihin sekä tarvittavien arvojen kirjoitus HMI-muuttujiin niiden loppukäyttäjälle näyttämiseksi.

Koska verrattain pienissäkin kohteissa voi olla tuhansia muuttujia, tulisi globaaleja muuttujia käyttää harkiten. Ei ole mitään syytä, miksi vaikkapa jossakin lohkossa laskennan väliaikaisten arvojen säilytystä varten tehdyn *rTemp*-muuttujan tulisi näkyä globaalisti koko ohjelmassa. Sen määrittäminen paikalliseksi muuttujaksi mahdollistaa myös sen, että saman nimistä muuttujaa

voidaan käyttää myös muualla ohjelmassa muussa laskennassa paikallisena väliaikaismuuttujana.

Muuttujien tulisi olla myös mahdollisimman kuvaavia nimeltään. Muuttuja nimeltä *iiTEA105_1* ei kerro juuri mitään kellekään ohjelmakokonaisuuteen vihkiytymättömälle. *iiTemperatureSensorChangingRoom* sen sijaan kertoo välittömästi, mitä arvoa ohjelmaa tarkasteleva henkilö katselee. Nykyisistä ohjelmointiympäristöistä jokaisesta löytyy ns. SmartCoding-ominaisuus, joka osaa tarjota ohjelmoijalle näppäillyn alun mukaisia muuttujia. Pitkääkään muuttujanimeä ei siis tarvitse kirjoittaa kuin kerran muuttujalistaan. Tämän jälkeen ohjelmointiympäristö osaa ehdottaa sitä jo muutaman näppäimen painalluksen jälkeen.

Muuttujanimiin on myös syytä käyttää kuvaavia etuliitteitä. Etuliitteitä ei ole standardoitu, mutta PLCOpen suosittaa käyttämään etuliitteenä vähintään muuttujan datatyyppiä. Fyysisiin sisään- ja ulostuloihin on ehkä syytä selvyyden vuoksi käyttää myös etuliitettä *i* sisääntulolle ja *q* ulostulolle. Datatyypeistä mainittakoon boolean-muuttuja, jonka etuliite on *x*, kokonaisluku *i*, etumerkitön kokonaisluku *ui*, reaaliluku *r* ja string-tyypin muuttuja *str*. Datatyypin tunnistaminen suoraan muuttujanimestä on hyödyllistä etenkin kokonaislukujen kohdalla, jotta etumerkittömään kokonaislukuun ei yritetä kirjoittaa negatiivista kokonaisluvun arvoa, jolloin säästytään turhilta ohjelman kaatumisilta. Tässä työssä hyväksi käytännöksi havaittiin myös käyttää etuliitettä *ton* vetohidasteisten ajastimien kohdalla, *rt* nousevien reunojen ja *ft* laskevien reunojen tunnistusten kohdalla. Etuliitteiden tulee olla pienillä kirjaimilla (PLCopen, 22).

On otettava huomioon myös se seikka, ettei ohjelmaa tule välttämättä käyttämään, korjaamaan tai ylläpitämään ohjelmoijan äidinkieltä puhuva henkilö. Globaali de facto -standardi on englannin kieli. Huonostikin artikuloitu ja käytetty englanti on vaikkapa unkarilaiselle tai intialaiselle huomattavasti hyödyllisempää, kuin suomenkielisillä muuttujanimillä ja kommenteilla varustettu ohjelma.

Ohjelman kommentointi on myös tärkeää. Vaikka muuttujat olisi nimetty hyvinkin kuvaavasti, on tärkeää kommentoida erityisesti pitkiä sekvenssejä kohta kohdalta. Vaikka kyseessä olisi itse tekemä ohjelma, on varsinkin laajoissa

kokonaisuuksissa joskus vaikea muistaa mitä, tai miksi on tehty jossakin tietyssä kohdassa ohjelmaa. Kommentointi helpottaa näin ohjelman analysointia esimerkiksi vianhakutilanteissa.

Luettavuus parantuu myös käyttämällä sisennyksiä. Erityisesti sekvenssien kohdalla ohjelmassa saattaa alkaa esiintyä useita sisäkkäisiä lauseita, joiden erottaminen toisistaan on merkittävästi helpompaa sisennyksiä käyttämällä. Työssä käytetyssä structured text -ohjelmointikielessä myös suurten toimilohkojen kutsumiseen on syytä suhtautua kuin aliohjelmakutsuun. Esimerkki tässä luvussa esitettyjen käytäntöjen toimeenpanosta on esitetty kuvassa 1.

```
272  CASE iState2 OF
273
274      1:
275          IF NOT xReadingValues THEN
276              tonError2(IN := TRUE, PT := T#15S);
277
278              fbGroundSourceHeatPumpEnergyMeter1P1(
279                  //inputs
280                  usiAddress := 1,
281                  eBaudrate := e1P1Baud,
282                  bSND_NKE := TRUE,
283                  bReadInit := FALSE,
284                  tMinSendTime := T#0S,
285                  bDisabled := x1P1Disabled,
286                  stCom := stCom,
287                  bStart := TRUE,
288                  //outputs
289                  bReady => x1P1Ready
290                  );
291
292              IF tonError2.Q THEN
293                  iState := 50;
294              END_IF
295
296              IF x1P1Ready THEN
297                  iState2 := 2;
298              END_IF
299
300          ELSIF xReadingValues THEN
301              iState2 := 0;
302          END_IF
303
304      2:
305          tonError2(IN := FALSE);
306
```

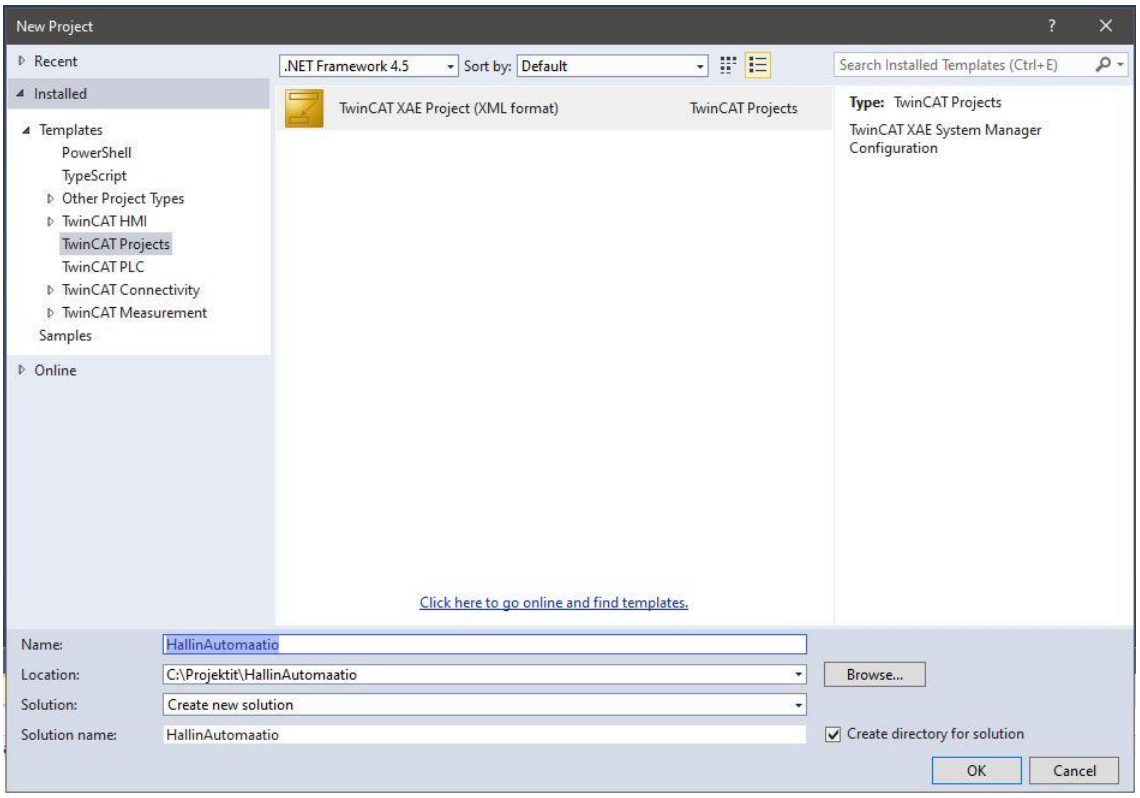
KUVA 1. Hyvän ohjelmointitavan implementointi

4.2 TwinCAT 3

4.2.1 PLC:n konfigurointi

PLC:n konfigurointi aloitettiin antamalla PLC:lle IP-osoite, jonka kautta sen kanssa kommunikoitaisiin. Tämä osoite oli samassa aliverkossa, kuin missä myös HMI-client -tabletit toimisivat. Tässä noudatettiin Windows 10:n IP-osoitteen asettamisen normaaleja toimintatapoja.

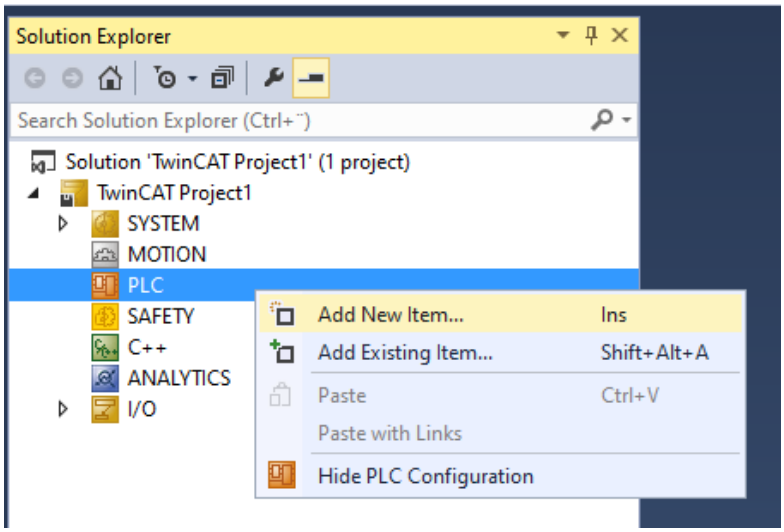
PLC:tä piti konfiguroida myös TwinCATissa. Tämä aloitettiin luomalla uusi projekti. Uusi projekti luodaan valitsemalla TwinCAT 3:n valikosta *File -> New -> Project*. Avautuvasta ikkunasta (kuva 2) valitaan malli *TwinCAT Projects* ja *TwinCAT XAE Project*. Valitaan projektille nimi ja tallennuskansio.



KUVA 2. Uusi projekti

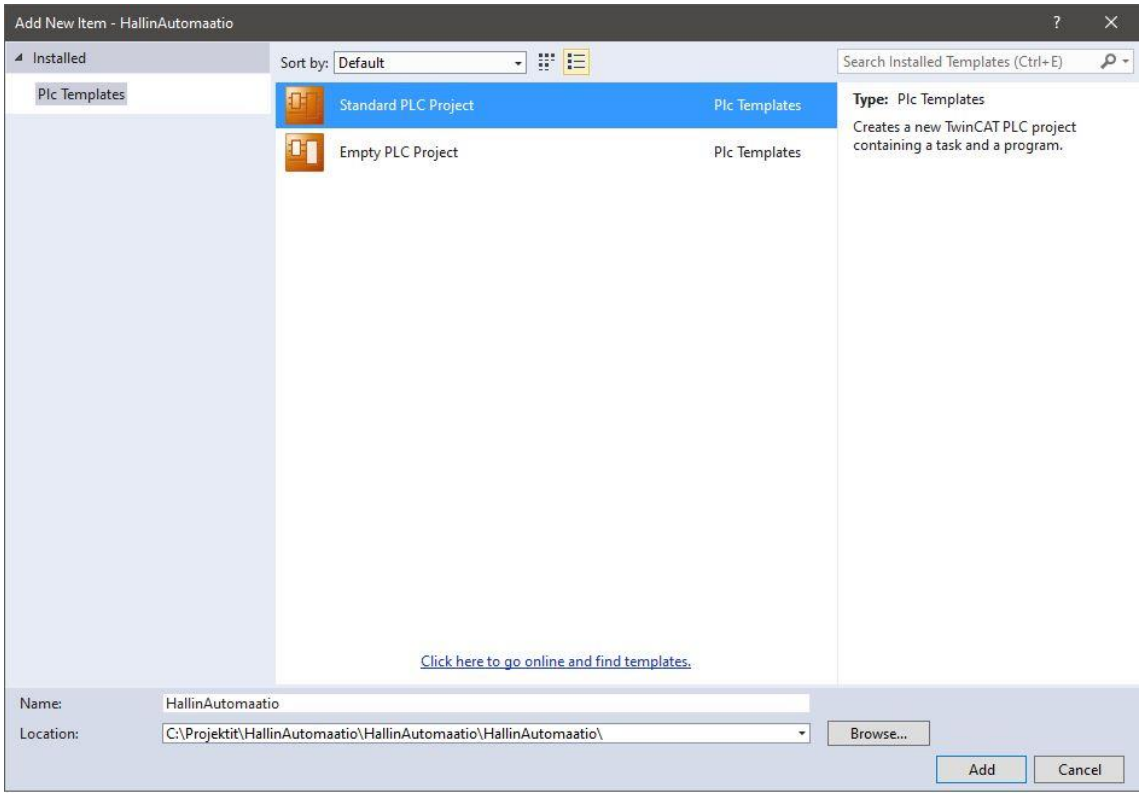
Uuden projektin avauduttua voidaan nähdä projektin rakenne ohjelmointiympäristön vasemmalla puolella sijaitsevasta *Solution Explorerista*.

Koska tekeillä olevaan projektiin ei tullut turvalogiikkaa eikä servo-ohjauksia, voitiin valita kohdan *PLC* päältä *Add New Item* (kuva 3).



KUVA 3. Add New Item

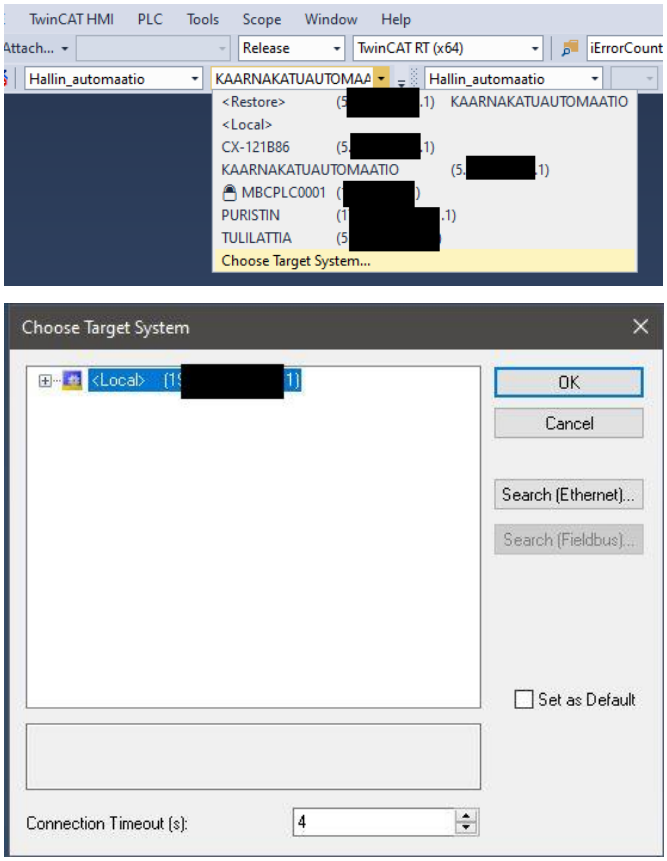
Avautuvasta ikkunasta valitaan *Standard PLC Project* ja annetaan ohjelmalle nimi (kuva 4).



KUVA 4. Standard PLC Project

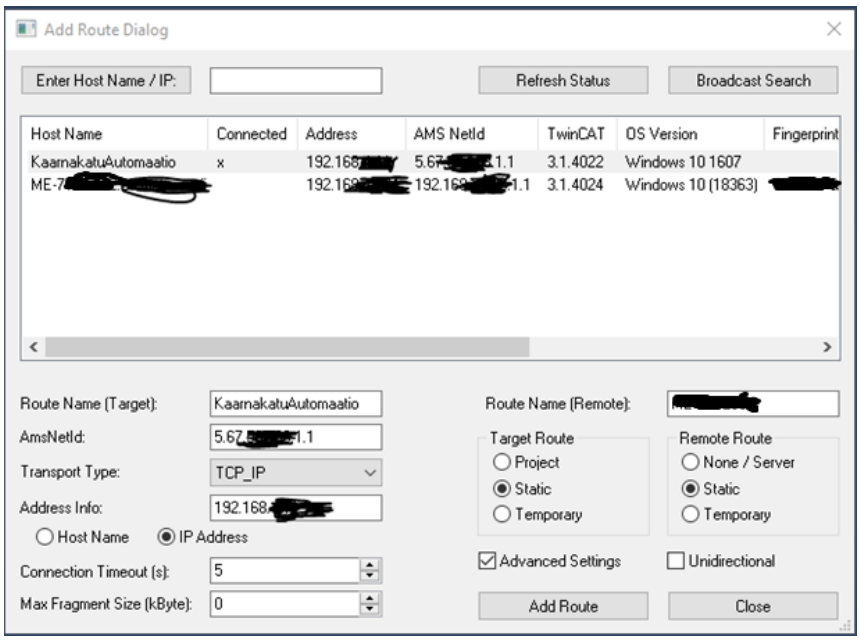
Tämän jälkeen projektissa on valmius alkaa ohjelmointi. Jos ohjelmointia ei haluta suorittaa paikallisesti ohjelmointiympäristössä, voidaan ohjelma siirtää

PLC:lle jo heti aluksi, jolloin tarvitsee rakentaa yhteys PLC:hen. Valitaan ylävalikosta keskimmäisestä alasvetovalikosta *Choose Target System*, josta valitaan *Search (Ethernet)* (kuva 5).



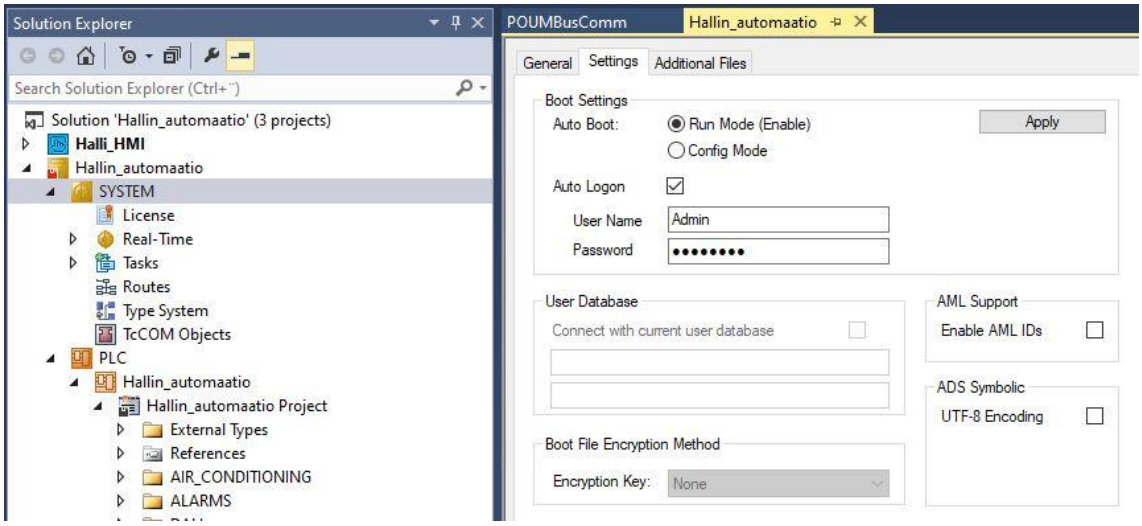
KUVA 5. Kohdelaitteen etsintä

Avautuvasta ikkunasta valitaan *Advanced Settings* päälle ja painetaan *Broadcast Search*. Valitaan asianmukainen verkkosovitin, josta etsitään. Löydetyistä laitteista valitaan haluttu, ja asetuksista valitaan *IP Address*, jonka jälkeen painetaan *Add Route* (kuva 6). Onnistuneen väylän luomisen jälkeen laitteen nimen viereen ilmestyy merkki X.



KUVA 6. Kohdelaitteen valinta

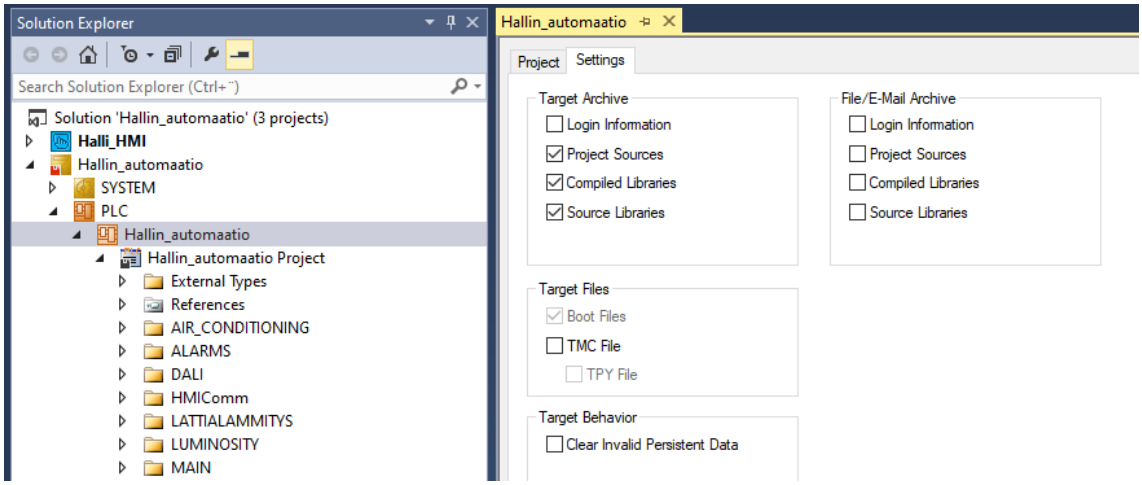
PLC halutaan run-tilaan aina, kun ohjelmaa ladataan, tai kun PLC käynnistyy. Kaksoisnapauttamalla *Solution Explorerista SYSTEM* ja valitsemalla *Settings*, päästään kuvan 7 esittämään valikkoon. Valitaan *Run Mode*, *Auto Logon* ja annetaan valittu käyttäjänimi ja salasana. Ne siis luodaan tässä, ja ladatessa ohjelma PLC:lle, ne jäävät sille talteen.



KUVA 7. SYSTEM Settings

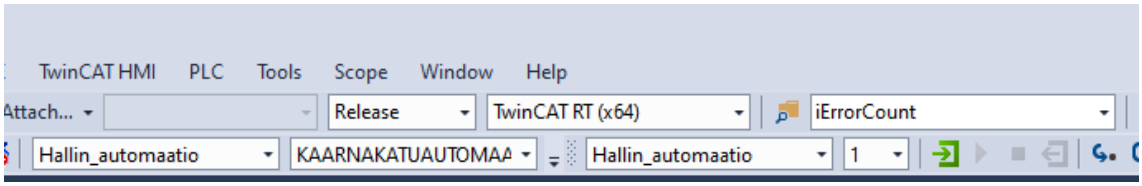
Mikäli asiakkaan kanssa on sovittu, ettei ohjelman lähdekoodi kuulu kauppaan, voidaan ohjelma ladata PLC:lle vain binäärisessä muodossa, jolloin sen lataaminen PLC:ltä TwinCATiin ei onnistu. Kaksoisnapauttamalla projektin nimeä ja valitsemalla avautuvasta ikkunasta *Settings*, päästään kuvan 8 esittämälle

sivulle. Kohdasta *Target Archive* poistetaan valinnat kohdista *Project Sources*, *Compiled Libraries* ja *Source Libraries*.



KUVA 8. Project Settings

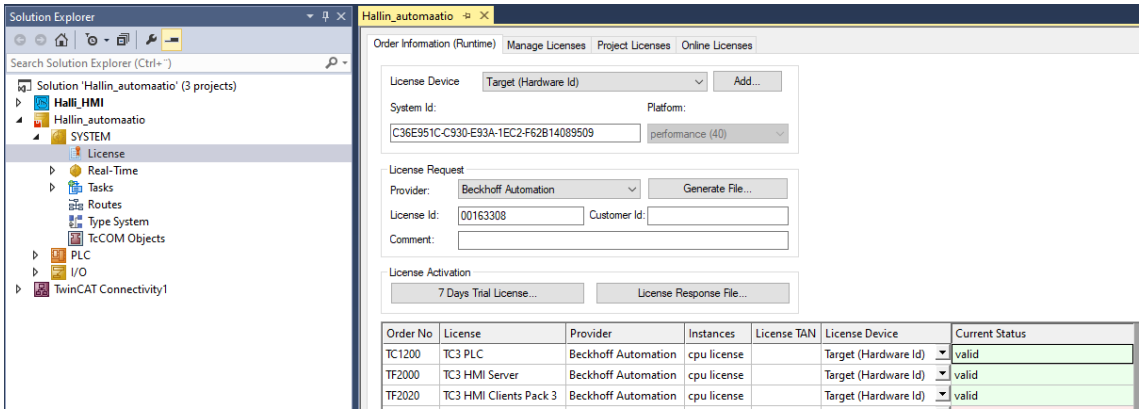
Tämän jälkeen, kun PLC:hen otetaan ylävalikosta vihreän nuolen osoittama *Login*-yhteys (kuva 9), saadaan ohjelma pyörimään PLC:llä ohjelmointiympäristöä pyörittävän PC:n sijaan.



KUVA 9. Login-yhteys

Beckhoffin PLC:tä voidaan pitää käynnissä pelkillä trial-lisensseilläkin, mutta tämä johtaa tilanteeseen, jossa ne on uusittava seitsemän päivän välein. Ei ole tarkoituksenmukaista, että asiakkaan täytyy ottaa yhteys PLC:hen joka viikko ja aktivoida uudet trial-lisenssit. *Solution Explorerin* alta kohdasta *SYSTEM -> License* voidaan löytää lisenssien hallinta. Lisenssit on projektin suunnitteluvaiheessa määriteltävä ja ostettava Beckhoffilta. Jokaiseen Beckhoffin PLC:hen sisältyy peruslisenssi *TC1200*, joka mahdollistaa perusmuotoisten ohjelmien ja I/O:n käytön. Tähän projektiin ostettiin myös *TF2000*, joka on HTML5-pohjaisen HMI-sovelluksen lisenssi, *TF2020*, joka mahdollistaa useampien yhtäaikaisten HMI-asiakaspäätteiden käytön, ja *TF6420* tulevaa tietokantaintegraatiota varten.

Välilehdeltä *Manage Licences* voidaan projektiin lisätä manuaalisesti tarvittavia lisenssejä. Kun kaikki tilatut lisenssit on valittu projektiin, välilehdeltä *Order Information* (kuva 10) painetaan *Generate File*.



KUVA 10. Lisenssien hallinta

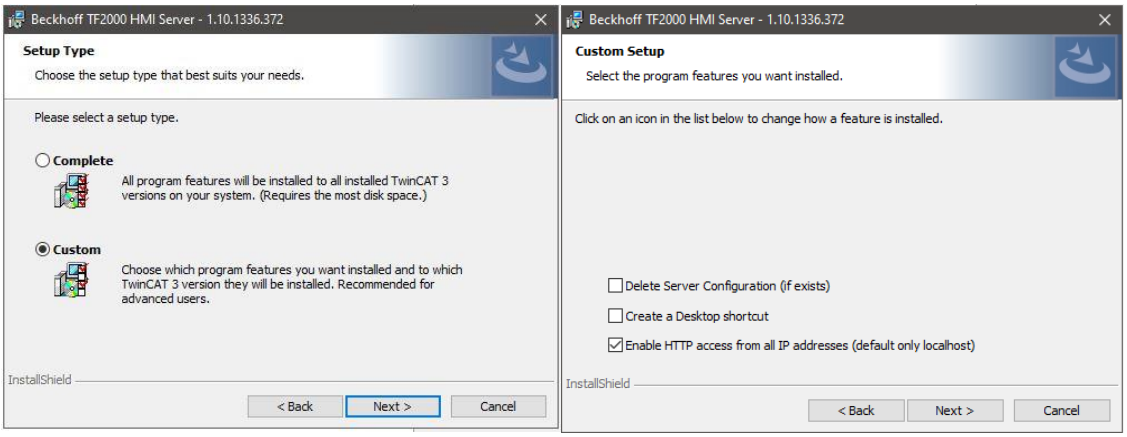
Toimenpide luo *TwinCAT License Request Filen*, joka on tclrq-päätteinen. Tämä tiedosto tallennetaan ja lähetetään sähköpostilla osoitteeseen tclicense@beckhoff.com. Automatisoitu vastaus kertoo joko ettei kaikkia pyydettyjä lisenssejä ole lunastettu, tai lähettää tclrs-päätteisen *License Response Filen*. On oltava huolellinen, että TwinCATilla on yhteys haluttuun PLC:hen, sillä kun painetaan *License Response File*, TwinCAT liittää lisenssit siihen PLC:hen, johon sillä on yhteys. Tätä yhteyttä ei tiettävästi saa purettua ainakaan ilman soittoa Beckhoffin tekniseen tukeen. Lisenssien onnistuneen käyttöönoton jälkeen niiden statukseksi muuttuu vihreälle taustalle ilmoitusteksti *valid*.

4.2.2 HMI:n konfigurointi

HMI-projektin luonti vaatii *TE2000 HMI Engineering* -lisäosan, jonka asennuspaketti sisällyttää sen TwinCATiin. Paketti on vapaasti ladattavissa Beckhoffin web-sivuilta.

HMI-serveriä pyörittävälle laitteelle taas on asennettava itse HMI-serveri, jonka paketti *TF2000 TC3 HMI Server* on myös ladattavissa Beckhoffin web-sivuilta. Asennuksessa on otettava muutama seikka huomioon. Oletusasetuksilla

asennettaessa HMI-serveri ei anna ottaa serveriin yhteyttä lokaalin laitteen, tässä tapauksessa PLC:n ulkopuolelta. On siis valittava asennuksessa *Custom*, jonka jälkeen on valittava kohta *Enable HTTP access from all IP addresses* (kuva 11).

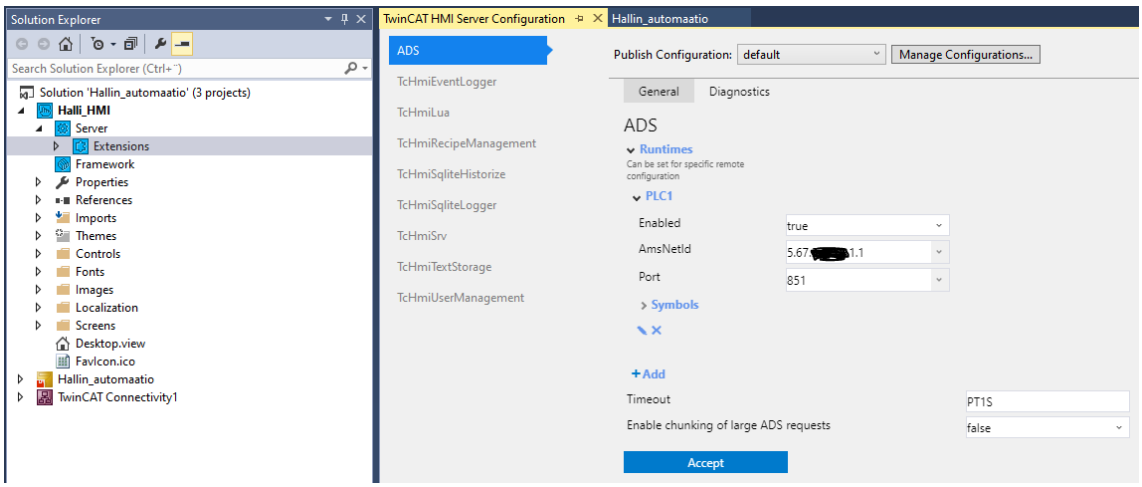


KUVA 11. Lokaalin ulkopuolelta tulevat yhteydet salliva asetus

HMI-serveri ei käynnisty automaattisesti PLC:n käynnistyessä. Jotta HMI-serveri saataisiin käynnistymään automaattisesti, tulee sille luoda pikakuvake PLC:n kansioon *C:\TwinCAT\3.1\Target\StartUp*.

TwinCATissa uuden tyyppinen visualisointiprojekti voidaan luoda joko kokonaan omana sovelluksenaan, tai liittää projekti osaksi PLC-sovellusta. Koska tämän työn HMI-projektia olisi vaikea käyttää esimerkki- tai pohjaprojektina muissa, päätettiin sisällyttää HMI-projekti PLC-sovellukseen. Tämä tapahtuu valitsemalla *File -> Add -> New Project* ja sen jälkeen valitsemalla *TwinCAT HMI Project*. Jälleen projektille annetaan nimi ja tallennuskansio.

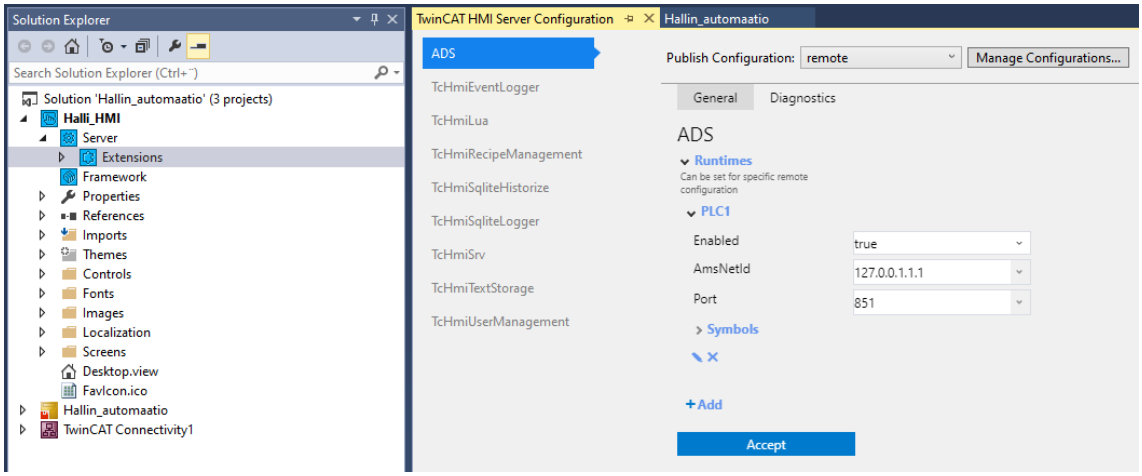
HMI-serveri täytyy konfiguroida niin, että oletuskonfiguraatio osoittaa siihen laitteeseen, jonka kanssa HMI-serveri keskustelee, ts. siihen, jolla logiikkaohjelma on käynnissä (kuva 12). Tätä osoitetta TwinCAT käyttää myös *LiveView*issä, jolla voidaan seurata reaaliaikaisesti tehtyjä muutoksia, joka on merkittävästi nopeampaa, kuin projektin julkaiseminen HMI-serverille.



KUVA 12. HMI-serverin default-konfiguraatio

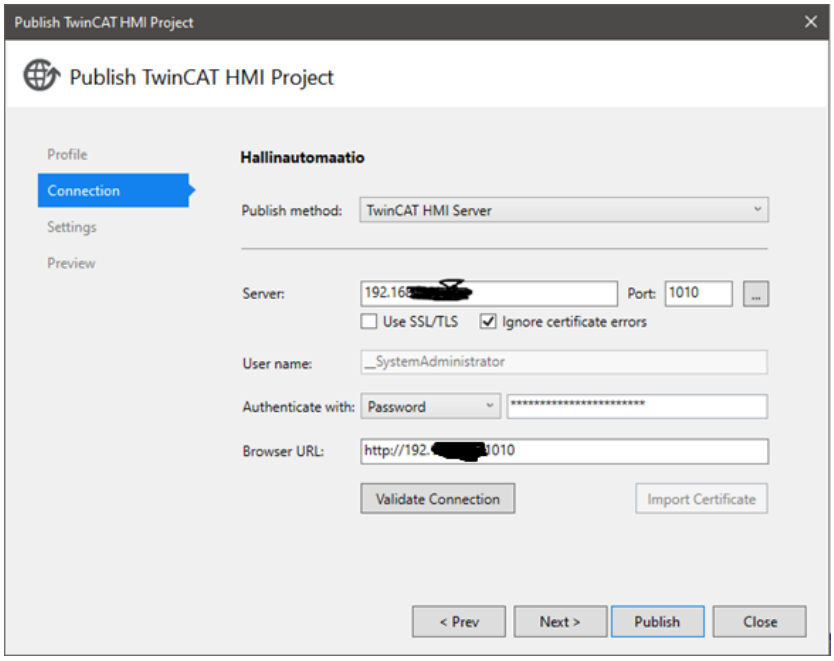
Kuten kuvasta 12 voidaan havaita, HMI ja PLC keskustelevat ADS-protokollalla. Tämä on Beckhoffin kehittämä protokolla, joka toimii TwinCATin sisäisessä kommunikaatiossa. Se on kehitetty ohjelmamoduulien keskinäiseen kommunikaatioon. Mikäli tietoa tarvitsee siirtää useamman PC:n välillä, se voi käyttää myös TCP/IP:tä pohjanaan (Beckhoff). ADS:n vaatima AmsNetId johdetaan IP-osoitteesta, jonka perään lisätään .1.1 (Beckhoff).

Remote-konfiguraatio tarkoittaa sitä konfiguraatiota, jolla HMI-ohjelma osoittaa sitä pyörittävään laitteeseen, eli HMI-serveriin. Tämän työn kontekstissa HMI-serveri on toiminnassa samaisella PLC:llä, kuin objektikoodi ja I/O-hallintakin, joten sille annetaan lokaali osoite 127.0.0.1.1.1 (kuva 13).



KUVA 13. HMI-serverin remote-konfiguraatio

Seuraavaksi luodaan profiili, jolla TwinCAT kommunikoi HMI-serverin kanssa. Valitaan *TwinCAT HMI -> Publish to TwinCAT HMI Server*. Profiilille annetaan kuvaava nimi. *Connection*-välilehdeltä asetetaan kohtaan *Server* sen laitteen IP-osoite, jolla HMI-serveri on toiminnassa (kuva 14). Portti salaamattomalle http-liikenteelle on 1010, salatulle 1020. Kun painetaan ensimmäistä kertaa *Validate Connection*, serveri pyytää luomaan salasanan käyttäjälle __SystemAdministrator. On huomionarvoista, ettei tämä ole sama käyttäjä ja salasana, jotka on luotu PLC:lle.



KUVA 14. Publish-asetukset

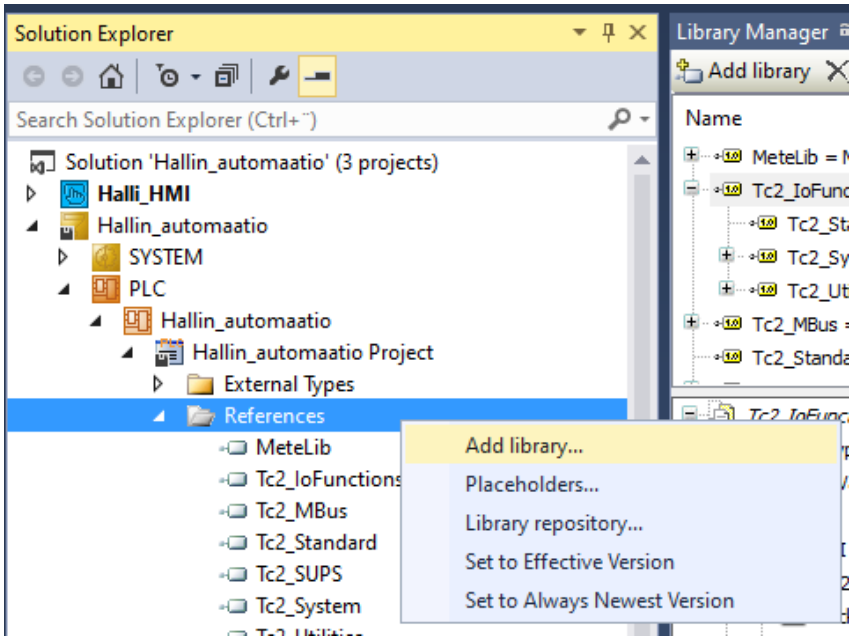
4.3 Ohjelmakokonaisuus

4.3.1 Lattialämmitys, kirjastojen lisäys ja muuttujien linkitys fyysiseen I/O:hon

Koska varsinaisen ohjelmointityön alkaessa muutto uusiin toimitiloihin oli jo tehty, aloitettiin ohjelmointi lattialämmityksestä. Seinien vieressä kulkevia lattialämmityspiirejä lukuun ottamatta kaikki piirit olivat säädettävissä termostaattien avulla. Säätöön valittiin yrityksen itse ohjelmoima PID-säädin, joka löytyi yrityksen omasta TwinCAT-kirjastosta *MeteLib*. Termostaatteja olisi voitu

ohjata myös analogisella jänniteviestillä, mutta pitkien välimatkojen ja binääristen ulostulokorttien hintaeron analogisiin verrattuna vuoksi ohjaustavaksi valittiin PWM-ohjaus. Tämä ulostulo oli valmiina säätimessä.

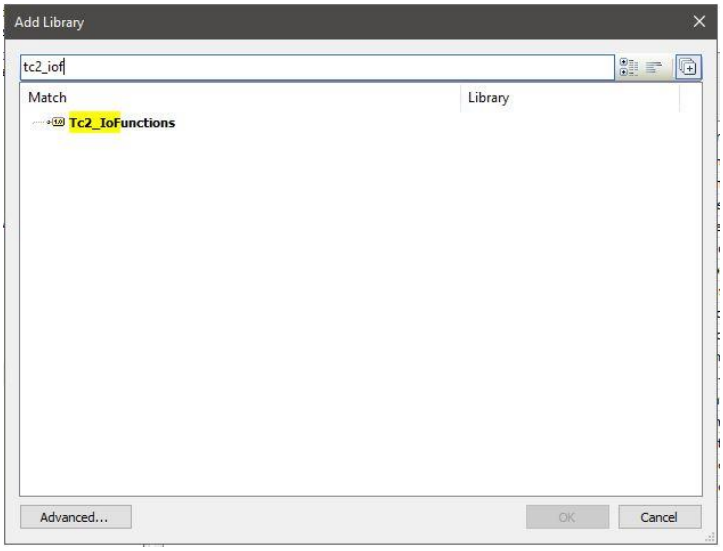
Mittaustieto säätimelle saatiin huoneisiin sijoitetuista NTC10K-tyypin vastusantureista, jotka oli yhdistetty KL3208-sisääntulokorttiin. KL3208 osaa ottaa vastaan vastustiedon useilta anturityypeiltä: Pt1000, Ni1000 ja eri tyypin NTC-anturit.



KUVA 15. Add library

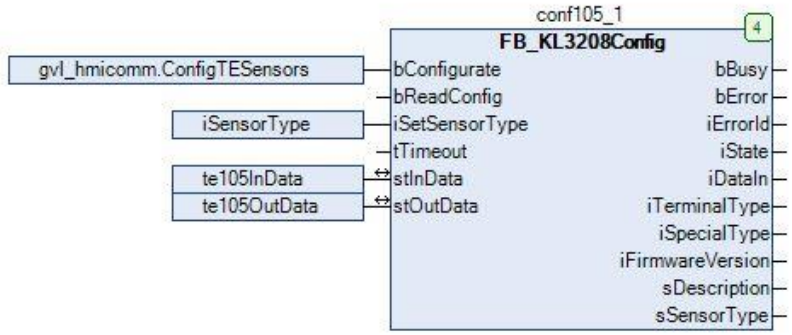
Koska KL-sarja käyttää kommunikointiin ethernetiä eikä EtherCATia kuten EL-sarja, ei konfigurointia voitu suorittaa suoraan TwinCATista. Konfigurointia varten Beckhoffin *Tc2_IoFunctions*-kirjastosta löytyy toimilohko *FB_KL3208Config*, jonka avulla anturityyppi voidaan kirjoittaa kortille. Kirjasto voidaan lisätä projektiin valitsemalla hiiren oikealla painikkeella *Solution Explorerista* kohdasta *References* ja sen jälkeen *Add Library* (kuva 15).

Tämän jälkeen avautuvasta ikkunasta etsitään ja valitaan haluttu kirjasto, tässä tapauksessa *Tc2_IoFunctions* (kuva 16).



KUVA 16. Lisättävän kirjaston valinta

KL3208 mahdollistaa jokaisen kanavan anturityypin valinnan erikseen. Tämä tarkoittaa tietysti myös sitä, että mikäli jokaiseen kanavaan tulee eri anturityyppi kuin oletus, eli Pt1000, tulee jokaiselle kanavalle tehdä oma instanssinsa toimilohkosta. Ryhmäkonfigurointia ei tunneta.



KUVA 17. Toimilohko anturityypin konfigurointiin

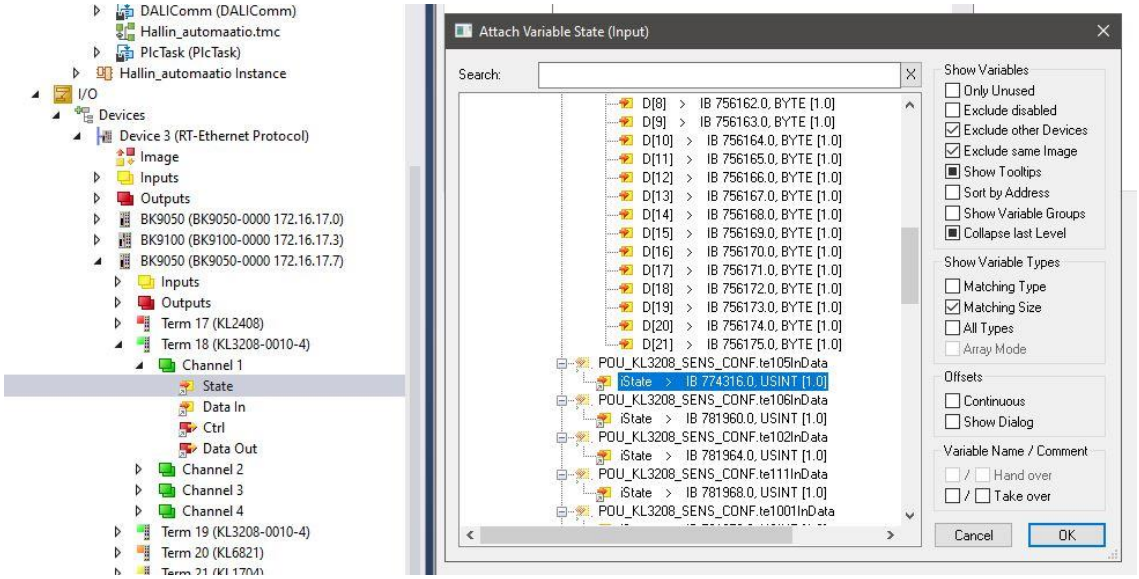
Toimilohko kommunikoi kortin kanssa in-out -muuttujien *stInData* ja *stOutData* avulla. *StInData*:an liitetään instanssi *Tc2_IoFunctions*-kirjastosta löytyvästä struktuurista *ST_KL3208InData* ja *stOutData*:an instanssi struktuurista *ST_KL3208OutData*, jotka tulee määritellä aliohjelman muuttujalistassa (kuva 18). Nämä struktuurit muuttujineen ovat suoraan yhteydessä sisään- ja ulostulojen prosessikuvaan. Näistä *ST_KL3208InData* kommunikoi kortin sisääntulon kanssa, joten sille täytyy varata muistipaikka logiikan sisääntulomuuttujille varatulta muistialueelta, ja *ST_KL3208OutData*-muuttujalle muistipaikka ulostulojen muistialueelta. TwinCATin tapa määritellä muuttujat on

esitetty kuvassa 18. Kuten kuvasta ilmenee, muiden modernien CoDeSys-pohjaisten ohjelmointiympäristöjen tapaan muuttujille ei tarvitse itse varata mitään tiettyä muistipaikkaa, vaan kääntäjä osaa tehdä sen itse.

```
te105InData AT %I* : ST_KL3208InData;
te105OutData AT %Q* : ST_KL3208OutData;
```

KUVA 18. Sisään- ja ulostulomuuttujien määrittely

Nämä muuttujat tulee linkittää kortin kanaviin. Tämä tapahtui navigoimalla kyseiseen korttiin *Solution Explorerin* I/O-listassa. Jokaiselle kortin kanavalle löytyvät sisääntulot *State* ja *Data In*, joihin linkitetään *ST_KL3208InData*-struktuurin muuttujat *iState* (kuva 19) ja *iDataIn*. Ulostuloille *Ctrl* ja *Data Out* linkitetään struktuurin *ST_KL3208DataOut* muuttujat *iCtrl* ja *iDataOut*.



KUVA 19. Muuttujien linkitys

Toimilohkon sisääntulomuuttujaan *iSetSensorType* annetaan anturin tyyppi kokonaislukuna. NTC10K-tyypin anturille tämä luku on 8 (liite 2). Tämä kokonaisluku kirjoitetaan kortille sisääntulon *bConfigurate* nousevalla reunalla. Toimilohko mahdollistaa myös konfiguraation lukemisen, jolloin sisääntulon *bReadConfig* nousevalla reunalla kortilta luetaan mm. laiteohjelmiston versio ja konfiguroitu anturityyppi, jotka näkyvät toimilohkon ulostuloissa. Huomattavaa on, että anturin tyyppi näkyy string-tyypin muuttujassa *sSensorType* luettavassa

muodossa, eikä kokonaislukuna, kuten toimilohkon sisääntuloon annettu anturityyppi.

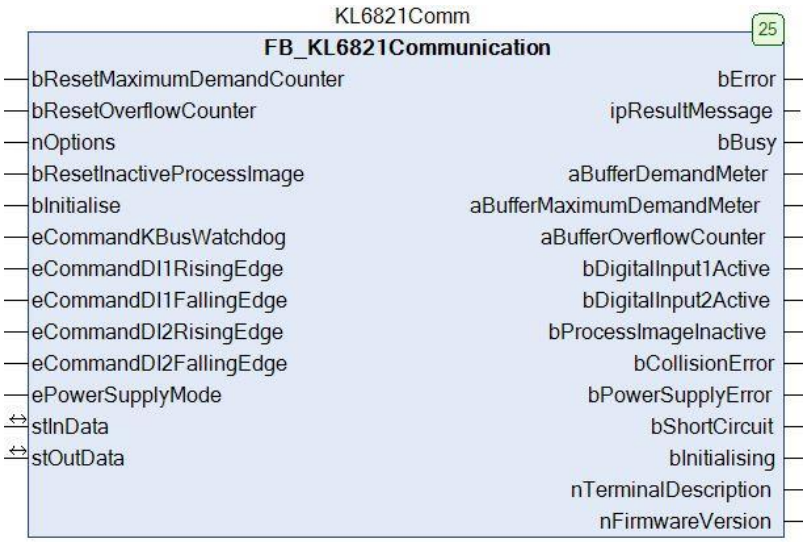
Kun anturien mittaustieto oli saatu kulkemaan kortilta muuttujaan, tuli tämä vielä jakaa kymmenellä, jotta saatiin muuttujaan oikea lämpötila-arvo Celsius-asteina. Kortti osaa siis anturityypin konfiguraation jälkeen tehdä itse skaalauksen vastusarvosta lämpötilaksi yhden desimaalin tarkkuudella. Koska tarkkuus on vain yhden desimaalin luokkaa, suoritettiin yhdessä ohjelmamoduulissa 120:n pisteen liukuvan keskiarvon laskenta. Jokaisesta huonelämpötilasta otetaan siis lukema puolen minuutin välein, ja viimeisen tunnin ajalta lasketaan keskiarvo. Tämän katsottiin helpottavan säätöä, ja näkyikin säätimen ulostulon vaihtelun vakaantumisena. Näistä lukemista tehtiin huonekohtaiset taulukkomuuttujat, joista tehtiin myös persistenttejä sähkökatkosten aiheuttamien taulukoiden nollausten välttämiseksi.

Ulkolämpötilan perusteella ohjattiin etuoven edustan betonilaatan ja rännien sulatusta. Tätä varten otettiin ulkolämpötila-anturin lukema kerran tunnissa ja tehtiin näistä arvoista 24-soluinen taulukkomuuttuja. Ohjauslogiikassa päädyttiin siihen, että jos viimeisen vuorokauden aikana lämpötila on ollut sekä plus- että miinusasteita, lämmitys olisi päällä. Jäätä ei kerry ränneihin, jos lämpötila on koko ajan joko pakkasen puolella tai koko ajan on plusasteita. Jäätä sen sijaan kertyy juuri silloin, kun ensin on sulaa, ja sen jälkeen taas pakkasta. Tällaisessa tilanteessa lämpötilan taas muuttuessa pakkaselta lämpöasteiksi, on myös mahdollista, etteivät rännit ehdi sulaa ennen kuin katolta tuleva sulamisvesi tarvitsisi tilaa.

Ohjelmamoduuleita lämmityksen ohjaukseen tehtiin kuusi: hallin lämmitys, ensimmäisen kerroksen tilojen lämmitys, toimistojen lämmitys, etuoven ja rännien sulanapito, KL3208:n konfigurointi sekä anturien mittaustietojen luenta, niiden taulukointi ja niistä keskiarvojen laskenta.

4.3.2 Valaistus ja tehtävien (*task*) luonti

Toimitilojen koko valaistus toteutettiin DALI-väylällä. Jokaisessa valaisimessa on DALI-käskyjä totteleva toimilaite, *control gear*. Toimilaitteille käskyjä antavat toimilohkot eivät suoraan kommunikoi KL6821:n prosessikuvan kanssa, vaan tähän kommunikointiin tarvitaan toimilohko *FB_KL6821Communication*, joka varastoi kaikki annetut käskyt kolmeen eri prioriteetin perusteella jaoteltuun puskuriin, *high*, *middle* ja *low*, jotka määritetään käskytoimilohkokohtaisesti parametrilla *eCommandPriority*. Näin estetään usean toimilohkon pääsy prosessikuvaan yhtä aikaa. *FB_KL6821Communication*-lohko (kuva 20) löytyy kirjastosta *Tc3_DALI*. Kirjasto lisätään projektiin luvussa 3.3.1 esitetyllä tavalla.

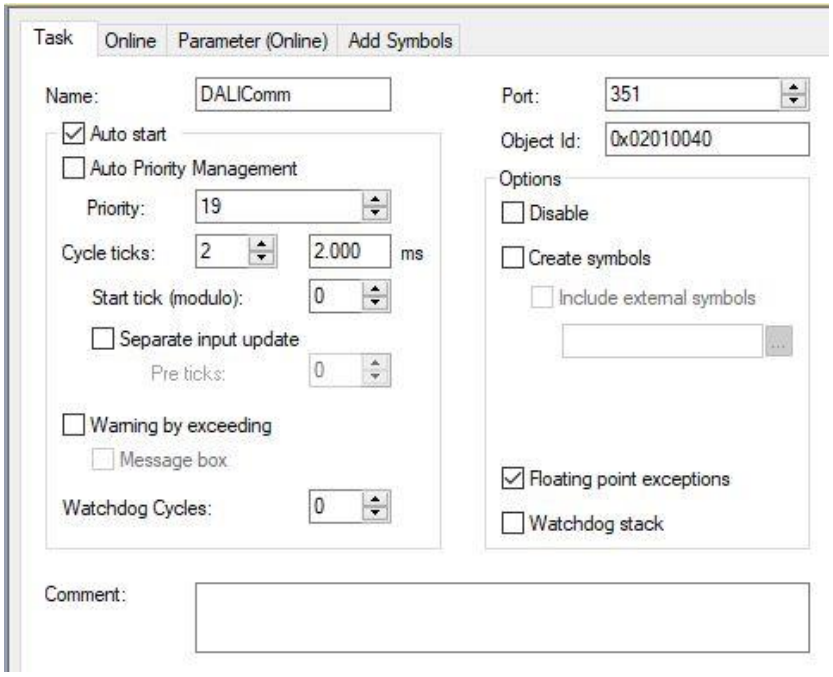


KUVA 20. FB_KL6821Communication

Toimilohkojen toimilaitteiden kanssa kommunikointiin tämän työn mittakaavassa ei käytännössä tarvita muuta kuin kommunikaatiolohkon linkitys kortin sisään- ja ulostulojen prosessikuvaan in-out -muuttujien *stInData* ja *stOutData* avulla. Näihin tuloihin liitetään instanssit struktuureista *ST_KL6821InData* ja *ST_KL6821OutData*, jotka ovat sisällytettyinä kirjastoon *Tc3_DALI*. Toimilohkosta löytyy kuitenkin suurempia sovelluksia varten hyödyllisiä ominaisuuksia, kuten puskurien käyttöaste *aBufferDemandMeter*, datan törmäystunnistus *bCollisionError* sekä DALI-kortin virransyötön puuttumisen tai virransyötön oikosulun ilmaiseva ulostulo *bShortCircuit*.

Koska kyseinen toimilohko lähettää väylälle puskuristaan yhden käskyn yhdelle toimilaitteelle kerrallaan, tulee se puskurien täyttymisen mahdollisimman tehokkaaksi estämiseksi sijoittaa nopeampaan tehtävään. Uusi tehtävä voidaan TwinCATissa luoda seuraavasti: valitaan *Solution Explorerista* kohdan *SYSTEM* alta *Tasks -> Add New Item*.

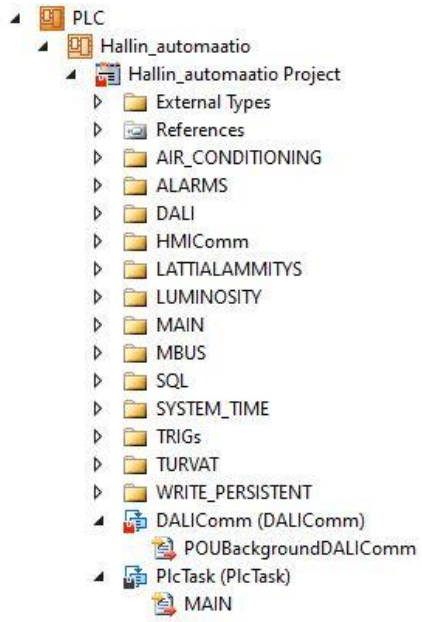
Annetaan tehtävälle nimi ja painetaan ok. Tämän jälkeen tehtävän asetuksista (kuva 21) voidaan asettaa mm. sen prioriteetti ja sykliaika. Tässä työssä sykliajaksi asetettiin 2 ms. Watchdog-toiminnolla voidaan myös asettaa varoitus tehtävän sykliajan ylittymisestä. Kommunikaatioportin TwinCAT asettaa useimmissa tapauksissa automaattisesti.



KUVA 21. Task Settings

Halutut ohjelmamoduulit voidaan liittää haluttuun tehtävään valitsemalla PLC-projektin alta halutusta tehtävästä *Add -> Existing item* kuvan 22 osoittamalla tavalla.

Jokainen KL6821-kortti vaatii oman instanssinsa toimilohkosta *FB_KL6821Communication*. Näitä kortteja oli kyseisen työn määrittelyn mukaan kolme, joten instanssejakin luotiin kolme.



KUVA 22. Ohjelmamoduulin liittäminen tehtävään

DALI-toimilaitteet ovat aluksi osoitteettomia. Jotta käskyjä antavat toimilohkot voisivat kommunikoida niiden kanssa, täytyi niille antaa ns. lyhyt osoite, eli toimilaitekohtainen osoite toimilohkolla *FB_DALI102Addressing*. Kuten aiemmin mainittiin, tämä, kuten muutkin DALI-toimilaitteille käskyjä lähettävät toimilohkot keskustelevat kommunikaatiotoimilohkon, eivätkä suoraan toimilaitteiden kanssa. Tämä tekee käskytoimilohkojen esittelystä (*declaration*) hieman normaalista poikkeavaa. Ohjelmalohkon muuttujalistassa muuttujat on esiteltävä kuvan 23 esittämällä tavalla. Lohkolle annetaan siis normaalisti instanssinimi, mutta niiden tyypin parametreissä on viitattava siihen instanssiin toimilohkosta *FB_KL6821Communication*, jonka kautta käsky toimilaitteelle halutaan lähettää.

```
VAR
    GiveAddresses : FB_DALI102Addressing(POUBackgroundDALIComm.KL6821Comm); //1st floor addressing
    GiveAddresses2 : FB_DALI102Addressing(POUBackgroundDALIComm.KL6821Comm2); // Shop floor addressing
    GiveAddresses3 : FB_DALI102Addressing(POUBackgroundDALIComm.KL6821Comm3); // 2nd floor addressing
END_VAR
```

KUVA 23. DALI-käskylohkon esittelyperiaate

Koska kaikki DALI-laitteet olivat uusia ja osoitteettomia, osoitteet antava toimilohko *FB_DALI102Addressing* (kuva 24) asetettiin antamaan osoitteet asetuksella *OpticalFeedback* sisääntuloon *nOptions*. Tämä asetus himmentää jokaisen valaisimen valotehon sille asetettuun minimiin, joka on oletusarvoisena juuri ja juuri näkyvä. Sitä mukaan, kun toimilaite saa osoitteen, se muuttuu jälleen kirkkaaksi. Näin voidaan helposti visuaalisesti todeta, että kukin yksiköistä on

saanut osoitteen. Osoitteistus toimii täysin satunnaisesti, eikä käyttäjällä ole kontrollia siihen, minkä osoitteen kukin päätelaitteista saa.

KUVA 24. Osoitteistustoimilohko

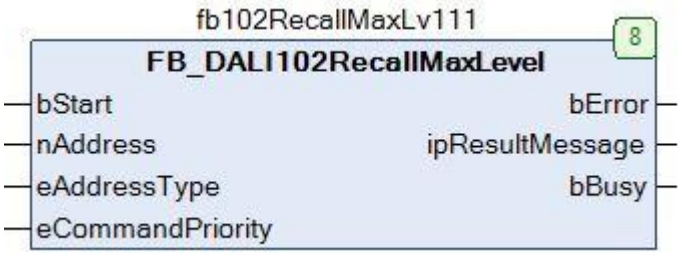
Jotta valoja olisi helpompi ohjata, niistä jokaiselle annettiin ryhmä. Ryhmiä voi olla yhden KL6821-kortin kontrollialueella 16. Jokainen toimilaite voi kuulua useampaan ryhmään. Ryhmänumero annetaan toimilaitteelle toimilohkolla *FB_DALI102AddToGroup* (kuva 25). Sisääntuloon *nAddress* annetaan osoite, joka halutaan liittää johonkin ryhmään. Tämä osoite voi olla toimilaitteen osoite, tai DALI-ryhmän numero. Tämä määritellään sisääntulolla *eAddressType*, joka ottaa vastaan *Tc3_DALI*-kirjastosta löytyvän *E_DALIAddressType*-enumeraatiomuuttujan (liite 3) mukaisia parametreja. Vaihtoehtoja ovat *Short*, *Group*, *Broadcast* ja *BroadcastUnaddr*. Näistä *Broadcast* lähettää käskyn kaikille toimilaitteille ja *BroadcastUnaddr* käskyn kaikille osoitteettomille toimilaitteille. Sisääntuloon *nGroup* annetaan sen ryhmän numero, johon toimilaite tai ryhmä halutaan liittää. Ryhmänumerot annettiin valaisimille huonekohtaisesti, jolloin niitä tarpeen mukaan voitaisiin myös sammuttaa.

KUVA 25. Ryhmäosoitinlohko

Valaistusta suunnitellessa on ollut tavoitteena, ettei fyysisiä kytkimiä tarvittaisi lainkaan. Toimitilojen jokaisessa osassa olisi todennäköisesti aina joku normaalina työaikana, joten liiketunnistimille tai kytkimille ei arvioitu olevan tarvetta. Normaalin työajan ulkopuolella toimitiloissa arveltiin olevan hyvin

harvoin ketään, joten kaikkien sisätilojen valojen päätettiin menevän päälle ja pois rikosilmoitusjärjestelmän tilatiedosta.

Valojen päälle laittoon käytettiin toimilohkoa *FB_DALI102RecallMaxLevel* (kuva 26). Kyseinen lohko asettaa valaisimen sille annettuun valotehon maksimiarvoon, joka oletuksena on 254. Valaisin menee päälle, mikäli se on käskyn saadessaan poissa päältä. DALIssa valaisimien valotehoa kontrolloidaan yhdellä tavulla, joten oletuksena kaikki valaisimet ovat toiminnassa 100 % teholla.



KUVA 26. RecallMaxLevel-lohko

FB_DALI102RecallMaxLevel-lohkolla saadaan joko jokin päätelaitteista tai päätelaitteista koostuva ryhmä asettumaan niille yksilöllisesti asetettuihin maksimiarvoihin. Jos siis on olemassa ryhmä 4, johon kuuluu 10 valaisinta, ja näistä vaikkapa yhdeksän valaisinta, joiden maksimiarvoiksi on annettu 254 ja yksi, jolle on annettu maksimiarvoksi 100, asettuvat yhdeksän valaisinta valotehon arvoon 254, ja yksi arvoon 100.

Valaisimet voidaan sammuttaa lohkolla *FB_DALI102Off*. Tälle lohkolle, kuten maksimiarvon kutsulohkollekin annetaan sisääntulona aloitusbitti, osoite, osoitetyyppi ja käskyprioriteetti, eli mihin kolmesta puskurista käsky menee odottamaan vuoroaan.

Vakuutussyistä toimitilojen ulkopuolella on oltava riittävästi valaistusta. Tämän valaistuksen tulee sopimusehtojen mukaan olla päällä aina, kun on riittävän hämärää. Koska Suomessa valoisuuden määrä vaihtelee suuresti vuodenajasta riippuen, päätettiin niiden kontrolli suorittaa ulkovaloisuusmittauksen mukaisesti. Katolle sijoitettu valoisuusanturi oli ulostulosignaaliltaan 0…10 VDC, jota logiikka käsittelee 16-bittisenä kokonaislukuna. Tälle mittaukselle suoritettiin yksinkertainen skaalaus 0…1000 lux, sekä 15 sekunnin välein suoritetusta

mittauksesta 120 pisteen liukuva keskiarvo. Keskiarvosuodatuksella vältyttiin mittauksen heittelyn aiheuttamalta valojen välkynnältä.

Koska ulkovalaistus piti olla pimeän aikaan vain riittävä, ei olisi ollut järkevää hukata sähköä täydellä teholla palaviin lamppuihin. Etupihan ja länsipuolen seinän valot haluttiin kuitenkin palavan täydellä teholla silloin, kun rikosilmoitin ei ollut päällä, jolloin ne toimivat pihan työvalaistuksena. Työvalaistukselle ei toisaalta ollut tarvetta enää silloin, kun kukaan ei olisi töissä. Etupihan ja länsiseinän valoille annettiin siis valaistuksen minimiarvo lohkolla *FB_DALI102RecallMinLevel*. Kuten maksimiarvon kutsulohko, myös tämä lohko antaa valaisimelle valotehon arvon ja laittaa valaisimen päälle, jos se ei ole jo ollut.

Maksimi- ja minimiarvot annettiin valaisimille lohkoilla *FB_DALI102SetMaxLevel* ja *FB_DALI102SetMinLevel*. Kuten muillekin kontrollilohkoille, myös näille lohkoille annetaan sisääntulona aloitusbitti, osoite, osoitetyyppi ja kutsuprioriteetti. Näille lohkoille spesifisti annetaan myös luonnollisesti maksimi- ja miniarvot.

Valaisinkohtaiset maksimiarvot etsittiin valotasoa mittaavan mittarin avulla. Standardi SFS-EN 12464-1 määrittää, että kun työ sisältää tietojenkäsittelyä tai jakokeskusten kokoonpanoa (liite 4), tulee työskentelypisteessä valoisuuden määrän olla vähintään 500 lx. Myös rakennuksen kaikki muu valaistus päätettiin asettaa tälle tasolle, vaikka standardin mukaan vähempikin taso riittäisi.

Maksimi- ja minimiarvojen kutsulohkoilla toteutettiin myös hallin ovikello. Hallissa on kovaäänisiä laitteita, kuten sorveja, metallisirkkeleitä ja CNC-jyrsin, joten pelkkä ääni on usein liian tehoton tapa ilmoittaa jonkun olevan ovella. Tätä tarkoitusta varten päätettiin vilkuttaa hallin valoja. Valot jaettiin kahteen uuteen ryhmään, joita ovikelloa käytettäessä laitettaisiin minimi- ja maksimiarvoihinsa vuorotellen. Tämä päätettiin tehdä myös taukotiloihin, jolloin kaikkien kahvitauolla ollessa ovikellon käyttö tulisi myös ilmi. Vilkutus toteutettiin yksinkertaisella kahden vetohidasteisen ajastimen vuorokytkennällä.

Neuvotteluhuoneeseen katsottiin tarvittavan valaistusta varten himmentimen. Tämä toteutettiin lohkoilla *FB_DALI102StepUp* ja *FB_DALI102StepDown*. Nämä lohkot joko lisäävät tai vähentävät yhden bitin verran valaistustehon tavusta. Näille lohkoille annettiin kaikkien muiden kontrollilohkojen tavoin sisääntulona aloitusbitti, osoite, osoitetyyppi ja käskyprioriteetti. Himmennyksen portaistus suoritettiin 30 ms:n ajastimella, joka antoi riippuen siitä, mitä nappia painettiin, joko ryhmälle käskyn mennä kirkkaammaksi tai himmeämmäksi. Valotehon koko skaalan läpikäynnin aika oli siis 7,65 s, joka todettiin hyväksi käyttää.

Ohjelmamoduuleita tarvittiin valaistusta varten useita. Moduulit luotiin toimintakohtaisesti. Moduuleita tuli siis omansa osoitteistukselle, ryhmitykselle, minimi- ja maksimiarvojen asettamiselle, himmentämiselle, ovikellon vilkutukselle ja valojen päälle ja pois asettamiselle. Myös kortin kanssa kommunikoinnille luotiin oma moduulinsa, joka siis toimii eri PLC-tehtävässä. Ulkovaloisuuden mittaukselle ja suodatukselle luotiin myös omat moduulinsa.

4.3.3 Turvatoiminnot

Hallin eri toimintoihin on jo sähkösuunnitteluvaiheessa päätetty rakentaa rikosilmoitusjärjestelmän päälläolon kärkitiedon perusteella joitakin turvatoimintoja. Useat toiminnot ovat kontaktorien takana, joista saadaan paluutieto. Hallin kahteen suureen nosto-oveen on esimerkiksi olemassa kaukosäätimet, joten niiden aukaiseminen rikosilmoitinjärjestelmän ollessa päällä estettiin katkaisemalla sähkö kontaktoreilta.

Myös päävesisulku laitettiin kiinni, sillä hallissa ei ole mitään vettä tarvitsevaa ihmisten poisollessa. Hallin paineilmakompressorilta ja myöhemmin halliin tulevalta siltanosturilta katkaistiin myös ohjaus.

4.3.4 Ilmastointi

Ilmastointikoneista kaksi, ensimmäisen kerroksen tilojen ja yläkerran toimistojen koneet olivat uusia, joita ohjattiin vain binäärisesti. Niistä löytyi neljä nopeutta,

joista alinta pidettiin aina päällä. Koneiden ohjaukseen tarvittiin siis kolme ulostuloa konetta kohden. Näistä toisiksi alin nopeus oli se, jolle annettiin logiikalta tieto, kun rikosilmoitin otettiin pois päältä. 3. ja 4. nopeus ovat tehostus ja maksimipuhallusnopeus, jotka laitettiin ajastimen taakse. Maksiminopeus voisi olla päällä 30 minuuttia ja tehostettu nopeus 60 minuuttia. Näille ajoille tehtiin muuttujat, joita myöhemmin voisi muuttaa HMI:lta. Huolimatta siitä, millä nopeudella ne kävisivät, rikosilmoitinjärjestelmän mennessä päälle, koneet tiputettaisiin hiljaisimmalle käyntinopeudelle.

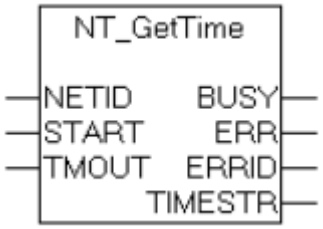
Hallissa oli kaksi konetta: korvaus- ja poistoilmalle, joita ohjattiin taajuusmuuttajilla. Kummassakin koneessa oli ilmatien sulkuventtiilit, jotka avautuivat niiden omilla moottoreillaan ja sulkeutuivat jousipalautteisesti. Takaisinkytkentänä venttiileiltä saatiin kaksi asentotietoa, joiden rajat olivat mekaanisesti säädettävissä.

Säätöpiiriä laitteille ei tehty, vaan niille annettiin kolme ennalta määrättyä nopeutta: alin nopeus, joka olisi päällä ihmisten poissaollessa, normaali paikallaolonopeus ja tehostusnopeus. Myös näille annettiin myöhemmin muutettaessa olevat muuttujat. Poiston ja sisääntulon suhteessa päädyttiin lukuun 21 / 23, poistoilman puhaltimen ollessa alempi luku. Sulkuventtiileistä johtuen puhaltimien käynti ei voinut alkaa heti järjestelmän käynnistyessä. Venttileiltä saatava takaisinkytkennän auki-tieto siis käynnisti puhaltimet.

Lämmön talteenoton kiertovesipumppu, jolla siis otettiin talteen sisäilman lämpöä ja lämmitettiin sisääntulevaa ilmaa käynnistettiin myös puhaltimien kanssa samaan aikaan. Kiertovesipumppu kuitenkin päätettiin sammuttaa ulkolämpötilan noustessa 16 °C:hen. Tällaisessa tilanteessa, jossa pumppu saattaa olla pois päältä puolikin vuotta, sitä päätettiin kuitenkin käyttää neljä minuuttia vuorokaudessa kiinni juuttumisen estämiseksi. Tätä tarkoitusta varten tarvitsi tietää aika, joka luettaisiin järjestelmän ajasta. Tähän tarkoitukseen Beckhoff tarjoaa toimilohkon *NT_GetTime* (kuva 27). Lohko löytyy kirjastosta *Tc2_Utilities*, joka on oletuksena liitettynä jokaiseen luotavaan projektiin.

Lohko lukee siis Windowsin ajan, joko paikallisesta laitteesta, eli PLC:stä itsestään, tai jostakin muusta laitteesta. Tämä laite voidaan määritellä

sisääntulolla *NETID*, joka jätetään tyhjäksi, jos kyse on paikallisen laitteen ajasta. Sisääntulo *START* aloittaa ajan lukemisen nousevalla reunalla. *TMOUT*:lla voidaan määrittää, milloin ajan lukeminen lopetetaan, jos sitä ei ole saatu luettua. Ulostuloista *TIMESTR* on se, johon aika saadaan näkyviin. Ulostulo on struktuurityyppinen, ja sen muuttujiin tulevat kuluvat vuosi, kuukausi, päivä, viikonpäivä (0…6), tunti, minuutti, sekunti ja millisekunti (liite 5). Kaikki ovat tyypiltään sanoja (*word*).



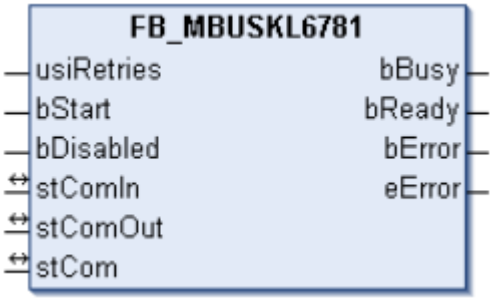
KUVA 27. Järjestelmän ajan lukeva lohko

Hallin ilmastoinnin pysäyttäviä ehtoja olivat jäätymisvahdilta, eli lämpötila-anturilta tuleva liian pieni mittaustulos, suodatinvahdin liian suuri paine-ero, taajuusmuuttajilta tulevat virhesignaalit sekä ilmateiden sulkuventtiilien auki-tiedon poistuminen järjestelmän ollessa käynnissä. Kaikki toteutettiin yksinkertaisilla IF-lauseilla.

Kesää varten rakennukseen, erityisesti toimistojen kattoihin oli sijoitettu jäähdytyskonvektoreita. Näille annettaisiin kaukosäätimellä jokin lämpötila-arvo, ja ne osaisivat itse ohjata itsensä päälle ja pois tarpeellisilla ajan hetkillä. Käynnissä olostaan ne ilmoittivat potentiaalivapaalla kärkitiedolla. Näiden tietojen perusteella ohjattiin maalämpöpumpun pääliuospumppua ja kahta eri jäähdytyspumppua. Ala- ja yläkerran tiloihin oli oma jäähdytyspumppunsa. Kun siis mikä tahansa tai useampi konvektori yläkerrasta menisi päälle, ohjattaisiin päälle sekä pääliuospumppu ja yläkerran jäähdytyspumppu. Alakerran kohdalla ohjattaisiin päälle pääliuospumppu ja alakerran jäähdytyspumppu.

4.3.5 Energian mittaus

Kaikissa kiinteistön energiamittareissa oli optio lukea ne M-Bus -väylällä. M-Bus -kommunikaatioon tarvittavat toimilohkot, struktuurit ja enumeraatiot löytyivät kirjastosta *Tc2_MBus*. Kuten DALI-väylän tapauksessa, mittatietoa lukevat toimilohkot eivät suoraan lukeneet ja kirjoittaneet KL6781-prosessikuvaa, vaan välissä oli oltava kortin kanssa kommunikoiva toimilohko *FB_MBUSKL6781* (kuva 28). In-out -muuttujat *stComIn*, tyyppiä *ST_KL6781inData22B*, ja *stComOut*, tyyppiä *ST_KL6781outData22B* ovat ne, jotka keskustelevat kortin prosessikuvan kanssa. DALI-väylästä poiketen mittatietoa lukevia toimilohkoja ei linkitetä kortin kanssa keskustelevaan lohkoon lohkojen muuttujaesittelyissä (*declaration*), vaan jokaisesta lohkosta löytyy in-out -muuttuja *stCom*, joka on struktuuri tyyppiä *ST_MBUS_Communication*.



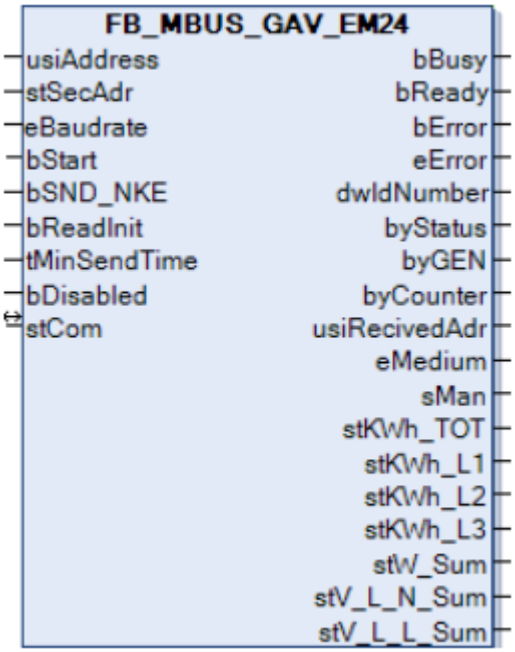
KUVA 28. M-Bus -kommunikaatiolohko

Energiamittareita rakennuksessa oli kuusi: ylä- ja alakerran lämminvesivaraajat, kolmen lämmitysvastuksen yhteiskulutus, maalämpöpumpun kulutus sekä ylä- ja alakerran tilojen yhteinen lattialämmityksen vesienergiamittari ja hallin lattialämmityksen vesienergiamittari.

Neljä ensin mainittua olivat tyypiltään Carlo Gavazzin valmistamia EM24-mallisia energian kulutusmittareita. Näiden lukemiseen Beckhoff tarjoaa valmiin toimilohkon, *FB_MBUS_GAV_EM24* (kuva 29).

Tämän lohkon tarpeelliset sisääntulot ovat *usiAddress*, jolla annetaan mittarin fyysinen osoite, *eBaudrate*, jolla annetaan lohkolle kommunikointinopeus, *bStart*, luvun aloitusbitti, *bSND_NKE*, joka alustaa mittarin jokaisella lukukerralla, *bReadInit*, joka asettamalla tilaan `TRUE` voidaan mittari lukea kerran jokaisen

kerran PLC:n kernelin käynnistyttyä sekä *tMinSendTime*, joka on mittarin lukuintervalli ja joka asettamalla nollaan voidaan mittari lukea antamalla nouseva reuna sisääntuloon *bStart*.



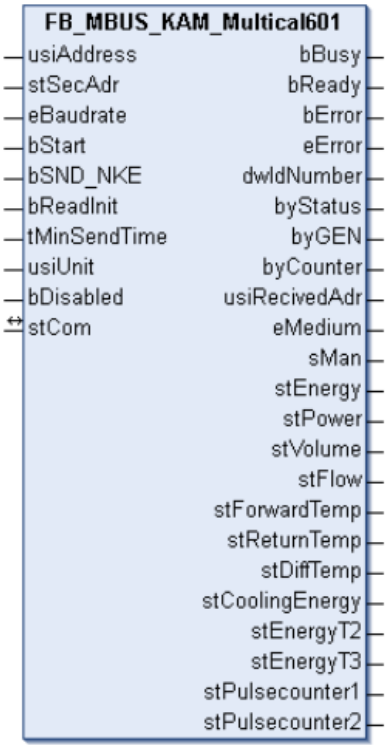
KUVA 29. Carlo Gavazzi -energiamittarin lukulohko

Ulostuloista tarpeelliset tämän työn kontekstissa olivat *bReady*, joka on tilassa `TRUE` yhden ohjelmakierron ajan kun lohko on saanut mittarin luvun valmiiksi, *bError*, joka muuttuu tilaan `TRUE` lukuvirheen sattuessa sekä *stKWh_TOT*, joka on mittarin antama kokonaistehonkulutus. Viimeksi mainitun tyyppi on struktuuri *ST_MBus_Info*, jonka muuttujat ovat *sValue* ja *sUnit*, jotka ovat string-tyypin muuttujia. Näistä *sValue* piti siis muuttaa reaaliluvuksi `STRING_TO_REAL`-komennolla.

Vesienergiamittarit olivat Kamstrupin mallia Multical 603. Beckhoff tarjoaa valmiin lohkon mallin Multical 601, joten sitä päätettiin kokeilla mittarin lukuun oman toimilohkon rakentamisen sijasta. Mittarien luku onnistui, joten lohkoa *FB_MBUS_KAM_Multical601* (kuva 30) käytettiin. Samat toiminnalliset sisään- ja ulostulot otettiin käyttöön kuin CG:nkin tapauksessa lukuunottamatta CG:n ulostuloa *stKWh_TOT*, joka tämän mittarin tapauksessa oli *stEnergy*. Tyypiltään sekin on struktuuri *ST_MBus_Info*.

Vaikka Beckhoff kertoo, että kaikille mittareille yhtä aikaa tuleva lukukäsky käsitellään kommunikaatiolohkon *FB_MBUSKL6781* toimesta niin, että jokainen mittari luetaan yksitellen siinä järjestyksessä, kuin niitä ohjelmassa kutsutaan, osoittautui tämä käytännössä kuitenkin hyvin vikaherkäksi. Mittareiden lukuun tehtiin sekvenssi CASE-rakenteella, jossa jokaisen mittarin lukuun tehtiin oma osuutensa, ja seuraavaan siirryttiin vasta luettavan mittarin *bReady*-ulostulon signaalilla.

Lohkot asetettiin lukemaan mittarien lukemat kerran päivässä keskiyöllä. Maalämpöpumpun tehotiedosta laskettiin, kuinka paljon sähkötehoa se oli päivän aikana käyttänyt. Samoin tehtiin vesienergiamittareiden kohdalla ja vesienergiamittareiden luvut ynnättiin. Jakamalla sisään menevän energian arvo maalämpöpumpun ottamalla energialla päädyttiin saamaan maalämpöpumpun hyötysuhde COP (*Coefficient of Performance*) (Mix, 4). Kaikki arvot kirjoitettiin kuukauden mittaisiin taulukkomuuttujiin, joista tehtiin myös persistenttiä dataa.

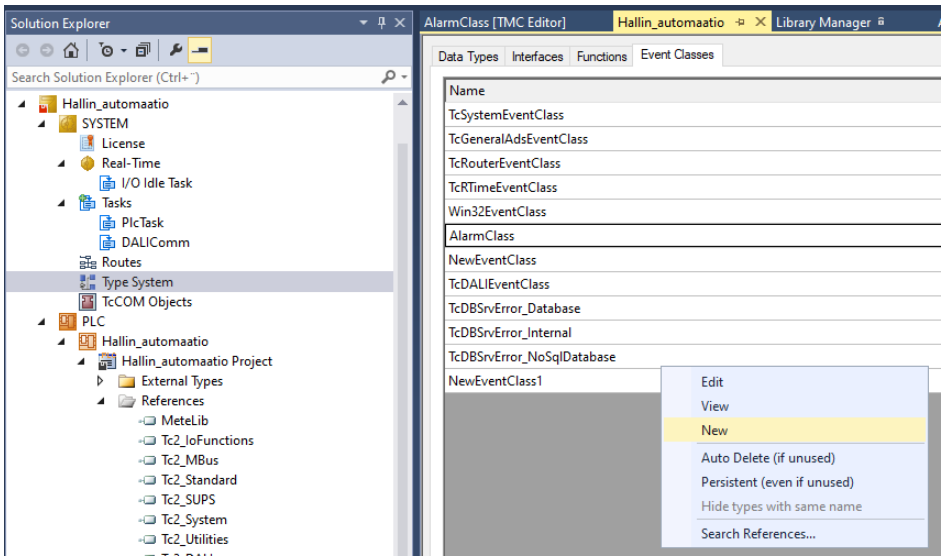


KUVA 30. Kamstrupin Multical 603:n lukemiseen tarkoitettu lohko

4.3.6 Hälytykset

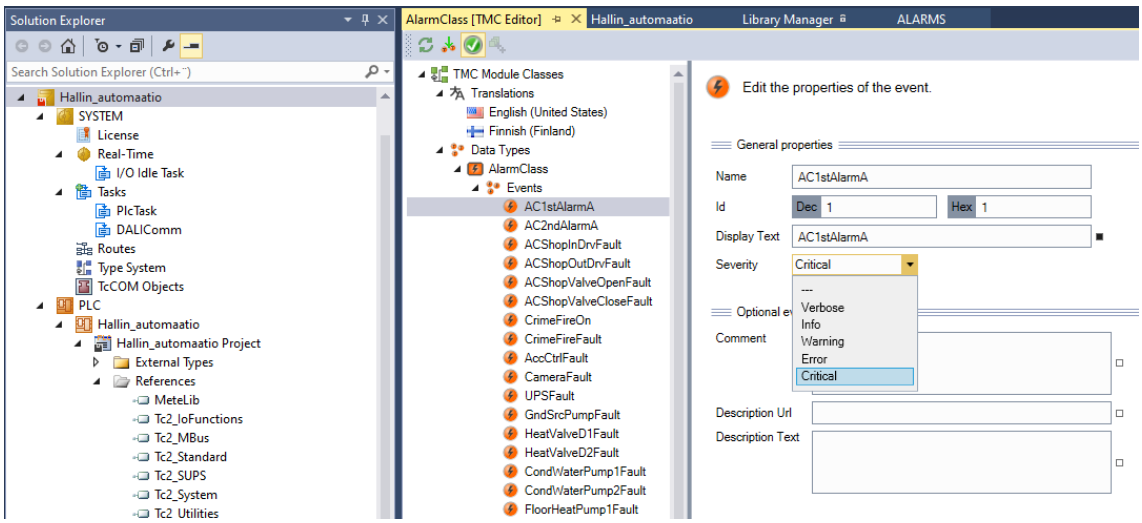
Koska halli on suuri, eikä joitakin asioita voi välttämättä aistein suoraan havaita, päätettiin HMI:ta varten ottaa useista tapahtumista hälytyksiä näkyviin. Näitä olivat mm. ilmastointikoneiden hälytykset, tiedot valvontakamerajärjestelmän vioista, UPS-järjestelmän vioista, useampien pumppujen vioista ja M-Bus -väylän vioista.

Hälytystoiminnot vaativat kirjaston *Tc3_EventLogger* lisäämisen. Kirjaston lisäämisen jälkeen on TwinCATista luotava luokka, joka pitää sisällään hälytykset. Tämän saavuttaakseen on navigoitava *Solution Explorerista* kohtaan *Type System*, kaksoisnapautettava sitä, ja valitsemalla alasivuista *Event Classes*. Tämän jälkeen on luotava uusi luokka valitsemalla hiiren oikealla ja sen jälkeen valitsemalla *New* (kuva 31).



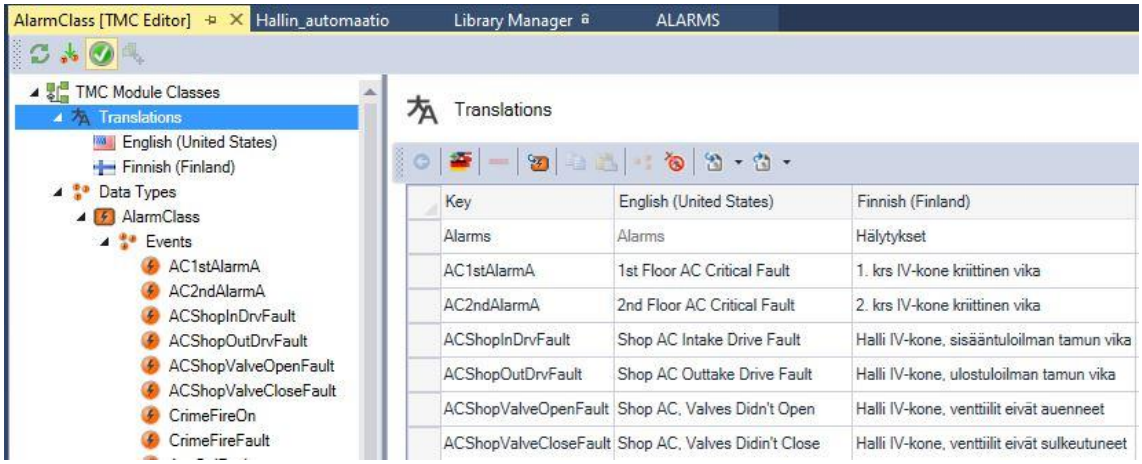
KUVA 31. Event classin luominen

Luokalle annetaan nimi, jonka jälkeen kohdan *Events* päällä valitsemalla hiiren oikealla ja tämän jälkeen *Add new event* voidaan luoda halutut hälytykset (kuva 32). Hälytykselle annetaan nimi ja sen vakavuusaste. Koska HMI tulisi toimimaan sekä englannin että suomen kielillä, kohtaan *Display Text* annetaan hälytyksen avain, jonka jälkeen sen viereisestä pienestä napista valitaan *Create Translation*.



KUVA 32. Hälytyksien luominen

Käännöksissä voidaan nyt antaa ne tekstit, jotka tulisivat näkymään HMI:n hälytyslistassa halutuilla kielillä. Uusia kieliä voidaan lisätä valitsemalla kuvan 33 esittämästä kohdasta *Translations* päältä hiiren oikealla *Add Language*. Kieliin on selkeästi panostettu, sillä niistä löytyvät mm. cherokee sekä jossakin päin Intiaa puhuttava kannada. Huomionarvoista on, ettei suomen kieli ole valinnoista löytyvä *Finnish*, vaan *Finnish (Finland)*.



KUVA 33. Hälytysten kielivalinnat

Hälytykset on nostettava itse ohjelmallisesti. Tähän tarkoitukseen on toimilohko *FB_TcAlarm*, joka löytyy kirjastosta *Tc3_EventLogger*. Jokaiselle hälytykselle on oltava oma instanssinsa kyseisestä lohkosta. PLC:n ensimmäisellä ohjelmakierrolla nämä lohkot alustetaan ja ne liitetään luotuihin hälytyksiin metodilla *CreateEx*. Tämän metodin muuttujiksi annetaan se hälytys, johon

halutaan viitata, halutaanko pakollinen käyttäjän varmistus sekä lähdedatan rajapintaosoitin, joka tämän työn tapauksessa on 0.

Esimerkki hälytystoimilohkon alustuksesta:

```
fbAC1stAlarmA.CreateEx(
            TC_EVENTS.AlarmsClass.AC1stAlarm,
            FALSE,
            0
            );
```

Hälytysehtojen täyttyessä hälytys nostetaan hälytystoimilohkon metodilla *Raise*. Tämän metodin argumentiksi annetaan aikaleima. Mikäli argumentti on 0, käytetään järjestelmän aikaleimaa. Hälytykset voidaan kuitata metodilla *Clear*. Metodin argumenteiksi annetaan jälleen aikaleima sekä boolean-muuttuja *bResetConfirmation*, jonka avulla voidaan manipuloida hälytystoimilohkon enumeraatiomuuttujaa *eConfirmationState*, joka on tyypiltään *TcEventConfirmationState*.

Hälytysten lisäksi luokan sisällä voidaan lähettää myös viestejä. Nämä toteutetaan toimilohkolla *FB_TcMessage*. Toimilohko alustetaan samalla tavoin kuin hälytyksetkin, lukuunottamatta käyttäjän varmistusparametria, joka viestilohkosta puuttuu. Viestejä ei tarvitse kuitata.

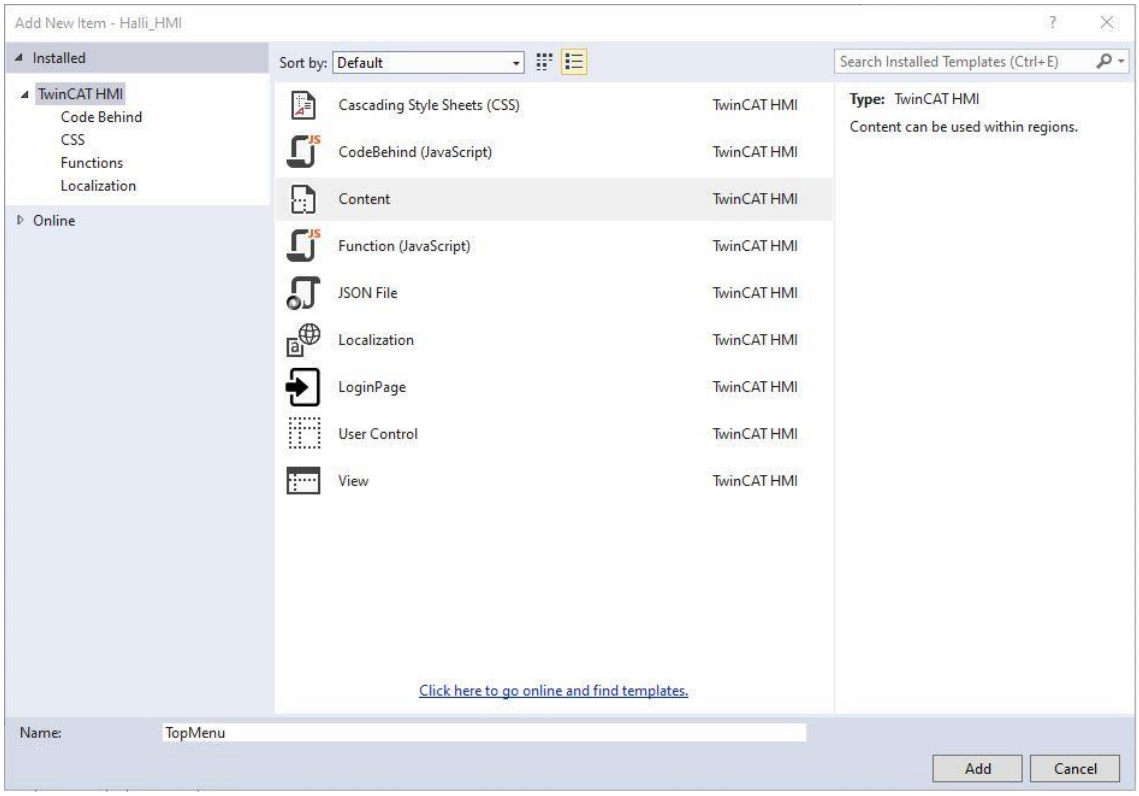
4.4 HMI-toiminnot

Osaa toiminnoista haluttiin kontrolloida käyttäjien toimesta ja osasta mittauksia haluttiin myös kertoa käyttäjälle graafisessa muodossa, joten automaatioon luotiin yksinkertainen HMI. Sovelluksen tekoon valittiin Beckhoffin uusi HTML5-pohjainen HMI, sillä siitä haluttiin yrityksessä kokemuksia muita tulevaisuuden projekteja varten.

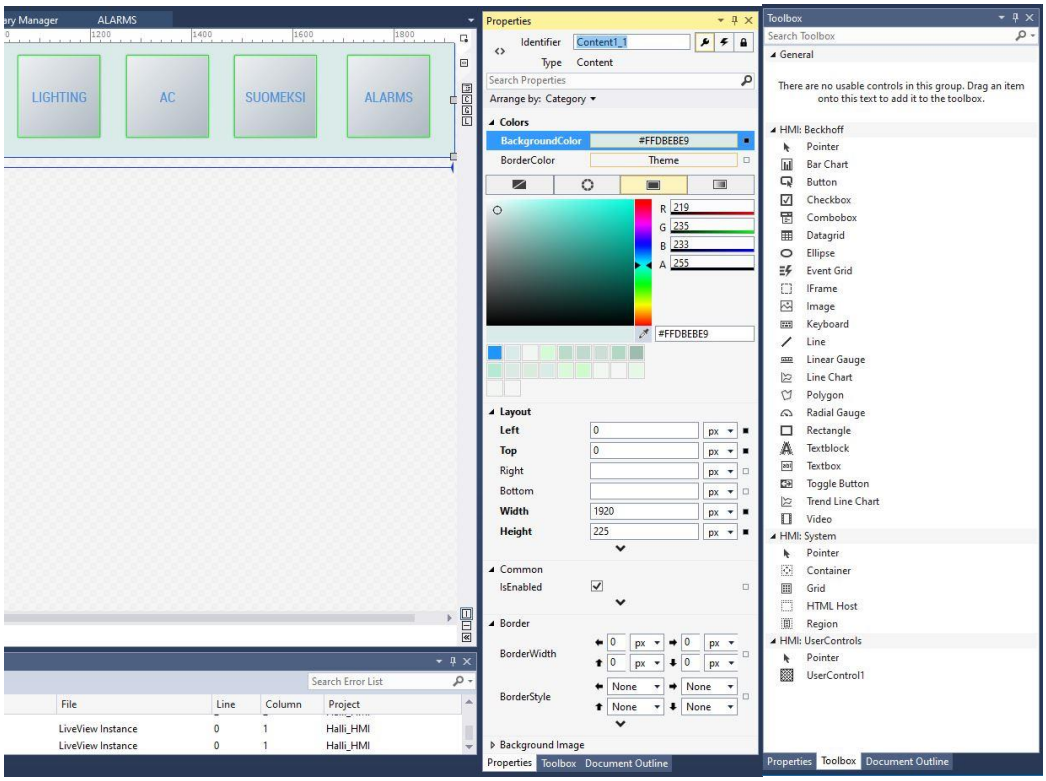
4.4.1 Sivujen luonti ja navigointi

Oletuksena aloitussivuna käytetään *Desktop.view*-nimistä ikkunaa. Tätä voidaan vaihtaa projektin asetuksista kohdasta *General -> Startup View*. Tähän näkymäikkunaan on hyvä sisällyttää kaikki, minkä halutaan näkyvän jokaisessa toimintaikkunassa; tässä työssä päävalikko ja huonelämpötiloja asetettaessa ilmestyvä numeronäppäimistö.

Ensin oli kuitenkin luotava itse päävalikko. Päävalikko, kuten kaikki sivut muutenkin toteutettiin sisältökomponenttina, tyypiltään *Content*. Tällaisen lisääminen onnistuu valitsemalla hiiren oikealla näppäimellä *Add -> New Item*, jonka jälkeen valitaan listasta *Content* (kuva 34). Sisällölle annetaan kuvaava nimi, jonka jälkeen ohjelma luo sisällön tällä nimellä. Luonnin jälkeen sisältö on täysin käyttäjän muokattavissa oikealta sivulta löytyvästä *Properties*-välilehdestä (kuva 35, vasen).



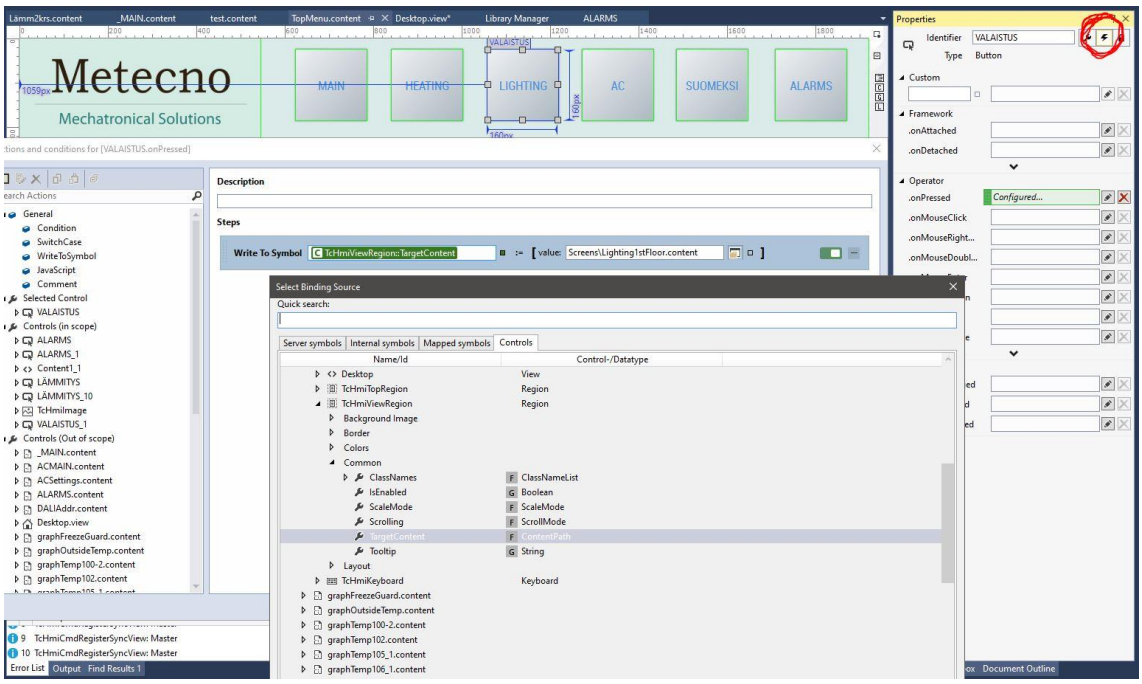
KUVA 34. Sisältösivun luonti



KUVA 35. Properties- ja Toolbox-välilehdet

Kun halutut sisällöt on luotu, voidaan palata *Desktop.view*-näkymään. Jotta haluttu päävalikko saatiin näkymään jokaisella sivulla, luotiin näkymään oikealla puolella sijaitsevan valikon *Toolbox*-välilehdeltä löytyvällä *Region*-työkalulla (kuva 35, oikea) näkymään alue, johon päävalikko sijoitettiin. Toinen alue luotiin lopusta ruutualasta. Päävalikon alueelle annettiin nimi *TcHmiTopRegion* ja loppuruudulle *TcHmiViewRegion*.

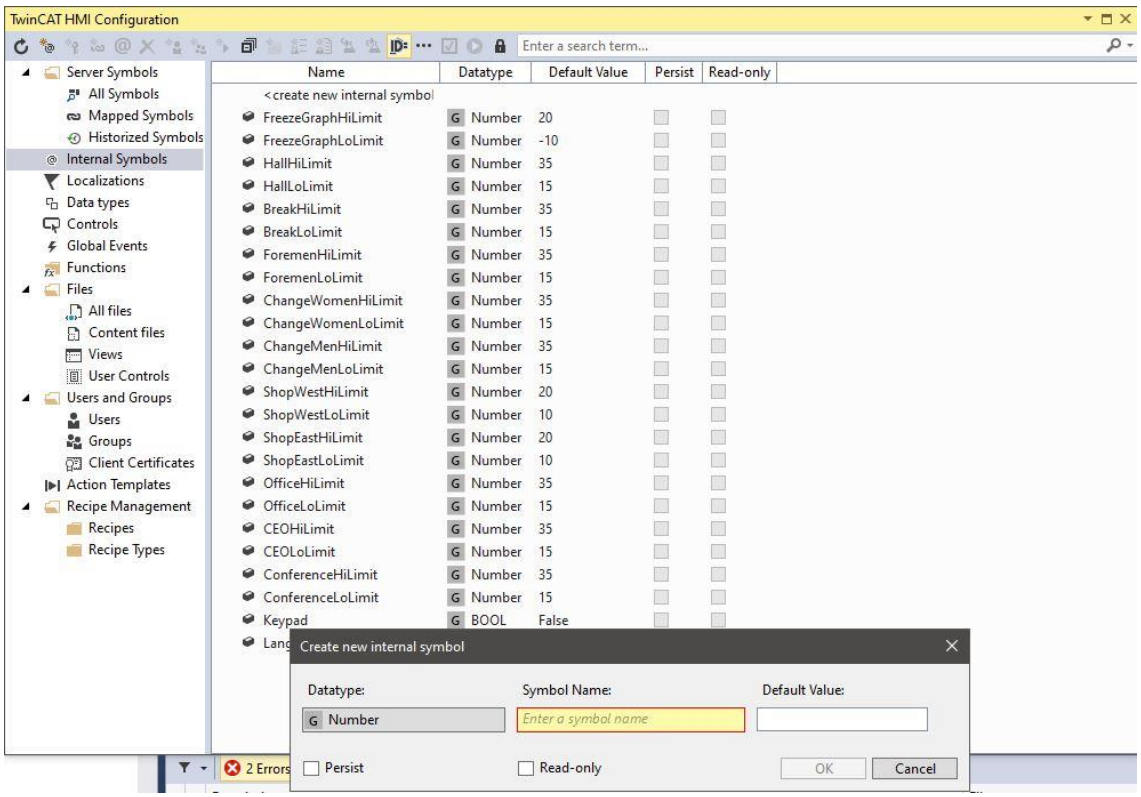
Jotta taas sivulta toiselle siirtyminen olisi mahdollista, pitää loppuruudulle annetun alueen kohdesisältöön kirjoittaa haluttu sisältö. Tämä tapahtuu luomalla painike ja avaamalla sen *Properties*-välilehdeltä *Show Events*. Tästä valikosta voidaan määrittää kaikki painikkeen tapahtumat. Koska painikkeen haluttiin vievän toiselle sivulle sitä painettaessa, valittiin operaattori *.onPressed*. Valittiin vasemmalta toiminto *WriteToSymbol* ja tämän jälkeen oikealla hiiren painikkeella valitsemalla *Create Data Binding*. Etsittiin *Controls*-listasta *TcHmiViewRegion*, jolle haluttu sisältö haluttiin kirjoittaa, ja sen parametri *Target Content*. Tähän parametriin kirjoittamalla haluttu sisältö se saadaan näkyviin ruudulla (kuva 36).



KUVA 36. Target Content

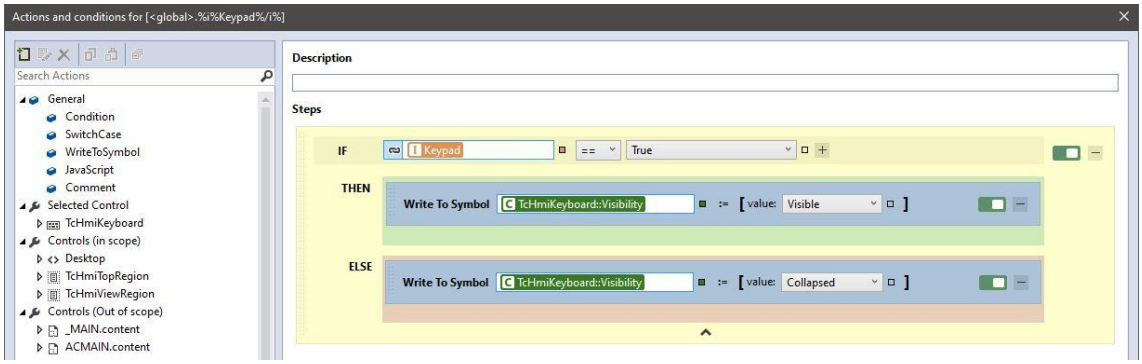
4.4.2 Sisäiset muuttujat

Joissakin tapauksissa HMI-puolella voidaan käyttää sisäisiä muuttujia. Jos HMI:ssa halutaan esimerkiksi vaihtaa kieltä, on turha siirtää tätä tietoa PLC:n ja HMI:n välillä. Objektien näkyvyydet ruudulla ovat myös tällaisia tapauksia, kuten edellä mainittu numeronäppäimistö. Numeronäppäimistöä varten luotiin boolean-tyyppinen muuttuja. Muuttujien luonti onnistuu *TwinCAT HMI Configuration*ista (kuva 37).



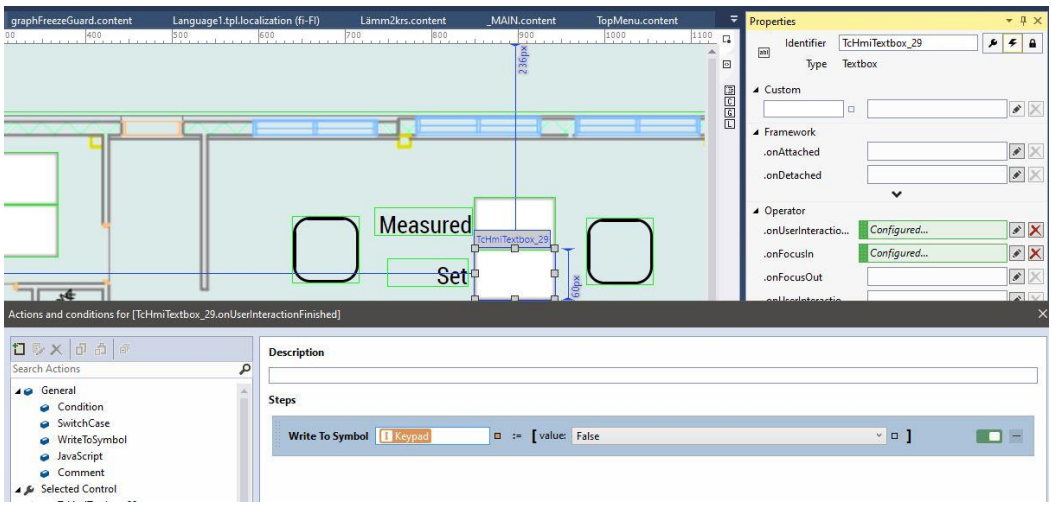
KUVA 37. HMI Configuration

Kuvan 38 osoittamalla tavalla `IF`-lausetta hyväksi käyttäen kirjoitetaan näppäimistön parametriin *Visibilty* sen olevan joko näkyvissä tai näkymätön.



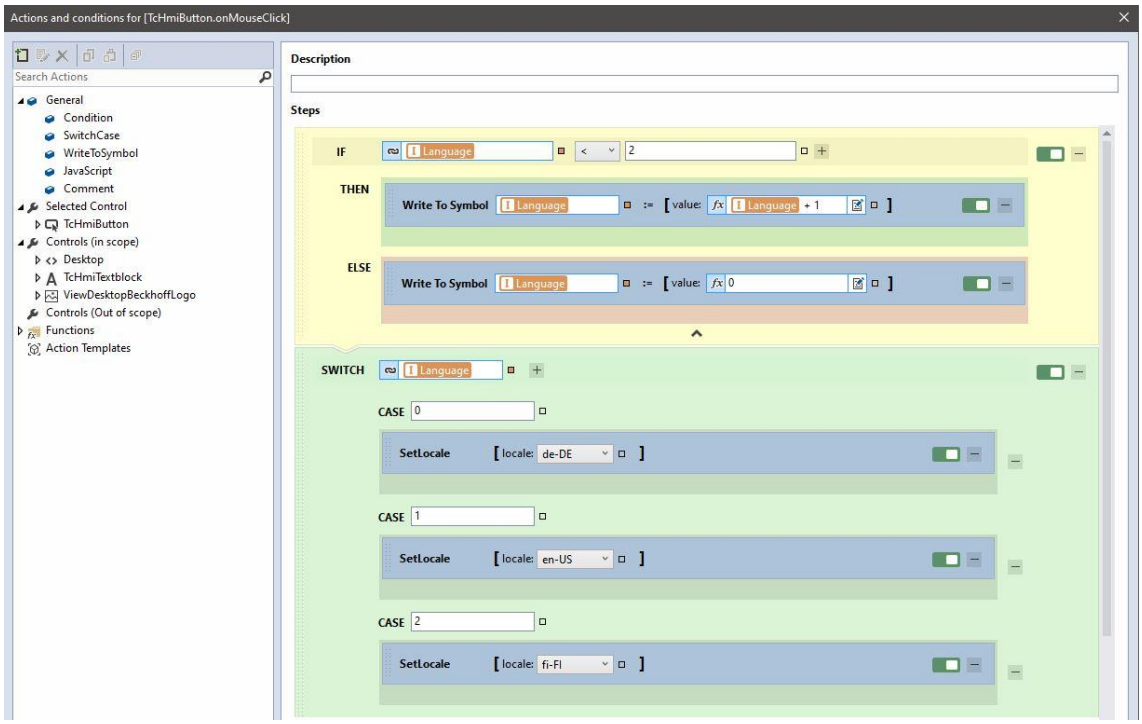
KUVA 38. `IF`-lause HMI-editorissa

Sisäistä muuttujaa *Keypad* manipuloimalla voidaan saada numeronäppäimistö näkyviin ja taas piiloon. Tämä tapahtuu asettamalla huoneenlämpötilan asetusruudut niiden parametrillä *.onFocusIn* kirjoittamaan muuttujaan `TRUE` ja parametrillä *.onUserInteractionFinished* `FALSE` (kuva 39).



KUVA 39. Keypad-muuttujan manipulointi

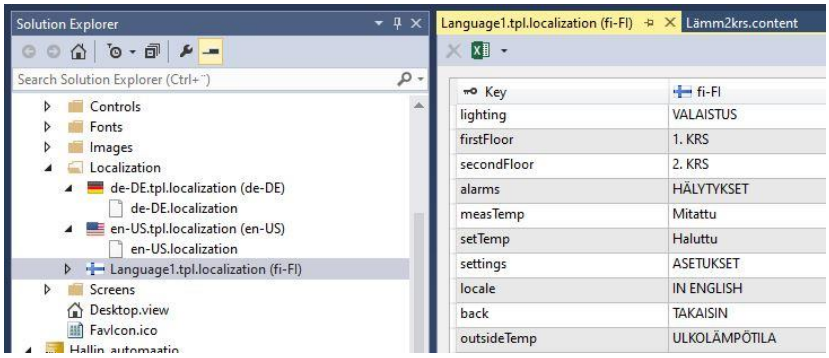
Myös CASE-rakennetta voidaan käyttää. Tällöin vaikkapa napin painallus voi lisätä aina yhden kokonaisluvun muuttujaan, kunnes palataan määritellyn kokonaisluvun jälkeen takaisin alkuarvoon. Jokaiselle kokonaislukumuuttujan arvolle voidaan antaa eri toiminto, vaikkapa kielivalinta. Kielivalinta onnistuu funktiolla *SetLocale* (kuva 40).



KUVA 40. Kielivalinta CASE-rakenteella

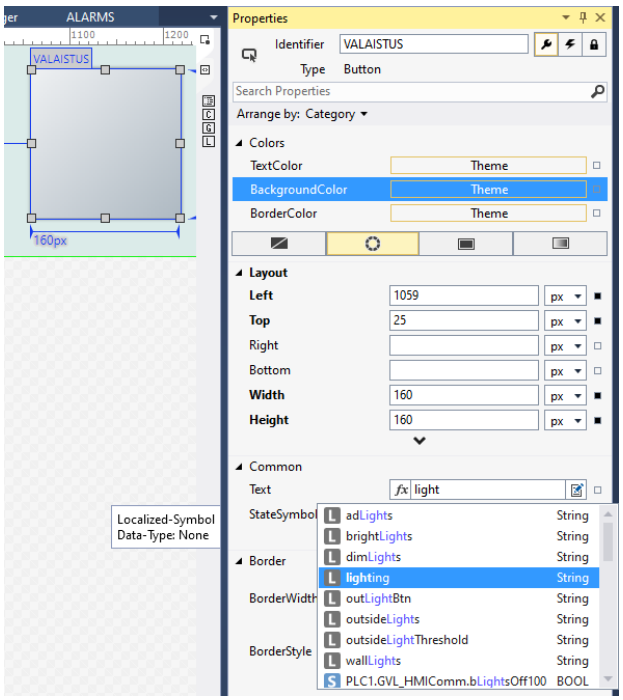
4.4.3 Lokalisaatiot

TE2000 mahdollistaa kaikkien tekstien lokalisoinnin eri kielille. Lokalisaatioita voi lisätä *Solution Explorerista* valitsemalla kansion *Localization* päällä hiiren oikealla ja tämän jälkeen valitsemalla *Add -> New Item -> Localization*. Lokalisaatioissa voidaan luoda lokalisaatiomuuttujia, joiden muuttujanimi toimii niiden avaimena. Tälle avaimelle voidaan tämän jälkeen antaa jokaiselle kielelle oma vastineensa (kuva 41).



KUVA 41. Lokalisaatiotekstien antaminen

Tämä muuttuja linkataan sen jälkeen vaikkapa napin tekstikenttään (kuva 42), joka hakee sitten annetuista lokalisaatiotaulukoista näyttövastineen riippuen voimassa olevasta kielivalinnasta.

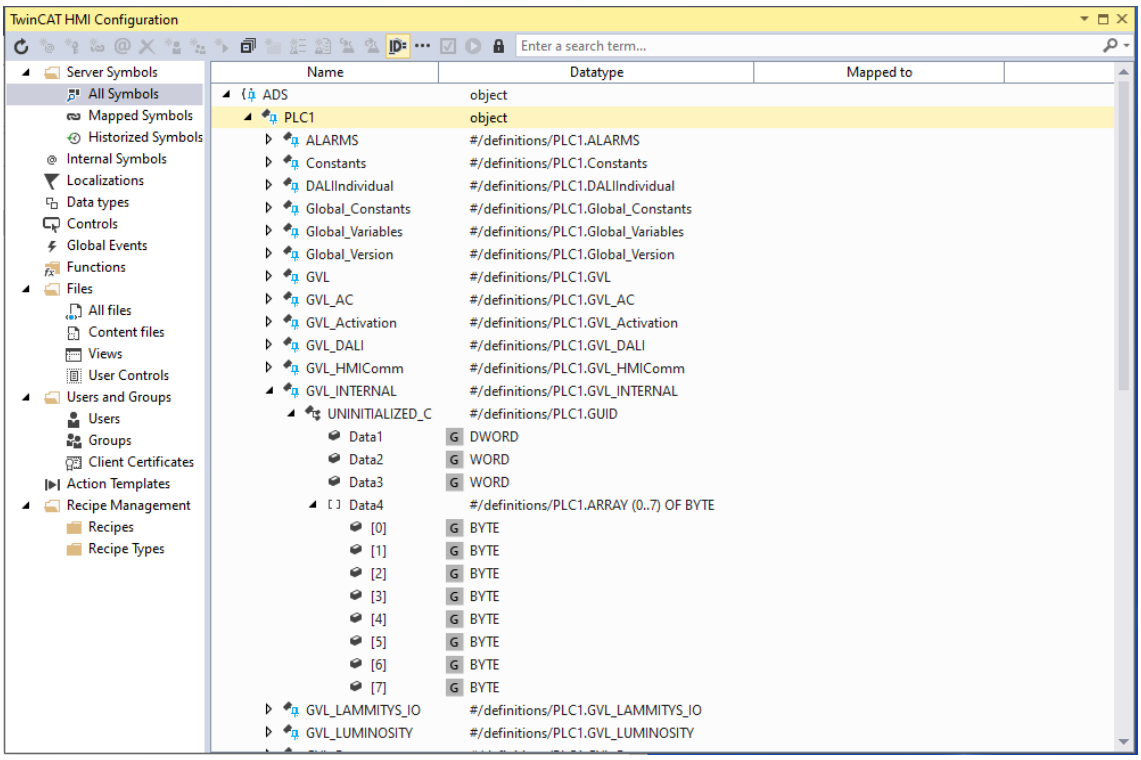


KUVA 42. Lokalisaatiomuuttujan käyttö

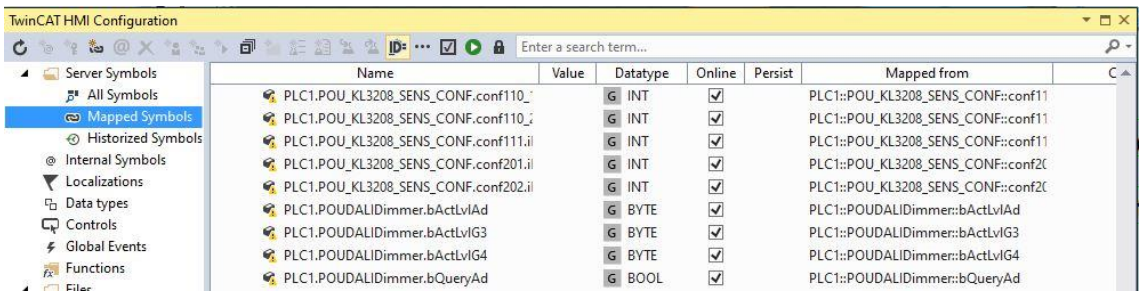
4.4.4 PLC-muuttujien käyttö HMI-projektissa

Jotta PLC-muuttujat saadaan linkitettyä HMI:hin, tulee ajantasaisen muuttujalistan olla PLC:llä. Kuten luvussa 3.2.2 todettiin, PLC- ja HMI-projektit ovat erilliset, ja ne keskustelevat ADS-protokollan kautta keskenään. Mikäli yhteyttä näiden välillä ei ole luotu, ei HMI-projekti pysty näkemään PLC-muuttujia.

Kun yhteys on luotu, voidaan PLC-muuttujat linkata *TwinCAT HMI Configuration*ista navigoimalla *Server Symbols -> All Symbols*, jonka jälkeen nähdään ensimmäisenä projektiin linkatut PLC:t ja näiden jälkeen ohjelmamoduulit ja muuttujalistat. Näistä muuttujia valitsemalla ne siirtyvät kartoitetuiksi muuttujiksi (*mapped symbols*) (kuva 43). Jos jotakin aiemmin kartoitettua muuttujaa ei löydy, sen kuvakkeen päälle tulee keltainen varoituskolmio. Mikäli yhteys PLC:n ja HMI:n välillä katkeaa, tämä kolmio näkyy listassa kaikkien muuttujien kohdalla (kuva 44).

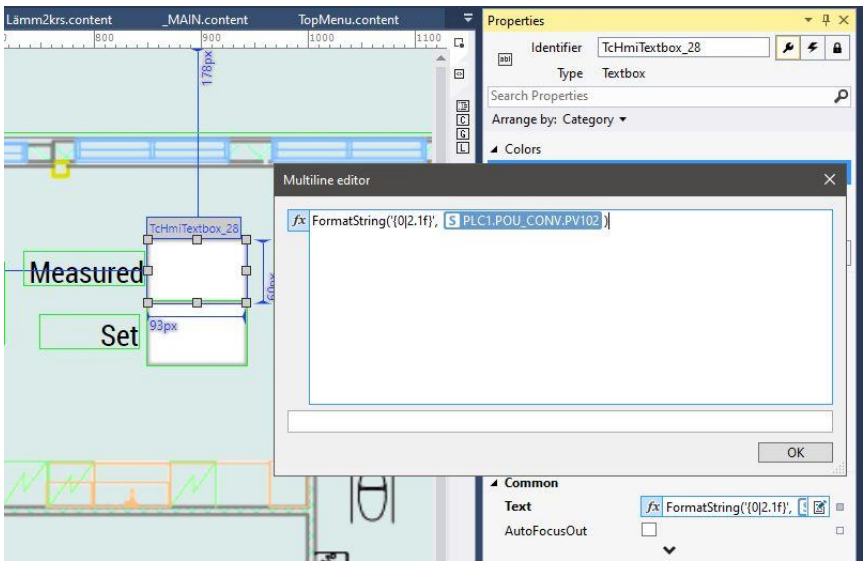


KUVA 43. Mapped Symbols



KUVA 44. Puuttuvat muuttujat

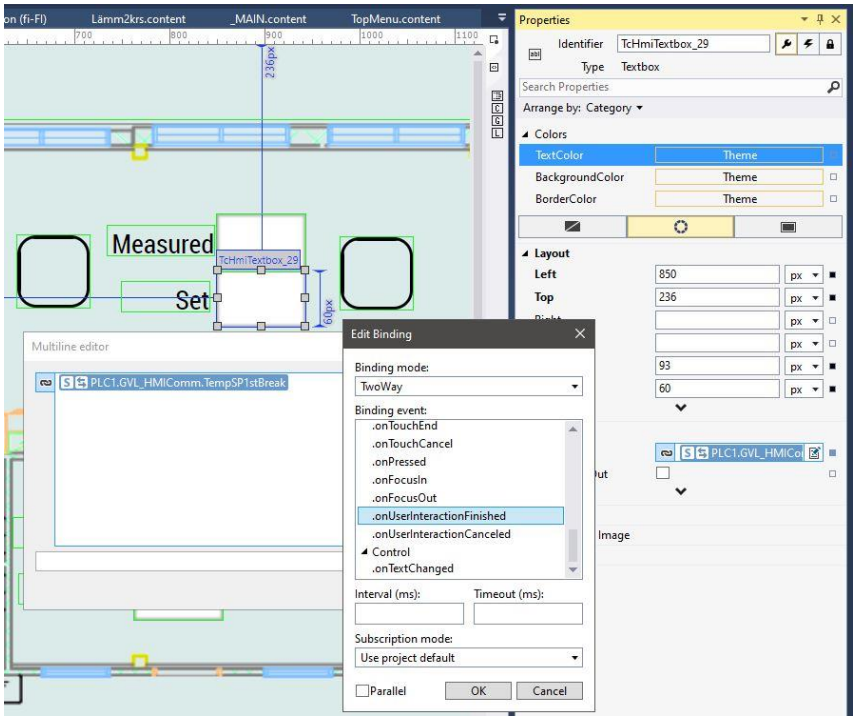
PLC-muuttujia voidaan joko vain monitoroida, tai niistä voidaan tehdä HMI:n kautta manipuloitavia. Monitoroitavia muuttujia voidaan formatoida tiettyyn muottiin. Esimerkiksi reaaliluvut ovat usein sellaisia, joista ei ole tarpeen näyttää useaa desimaalia. Tämä muokkaus tapahtuu funktiolla *FormatString* kuvassa 45 esitetyllä syntaksilla. 0 osoittaa formatointisyntaksin jälkeisen pilkun ensimmäiseen muuttujaan. Näitä muuttujia voidaan erotella pilkuilla useita. 2.1f kertoo, että ennen pilkkua esitetään kaksi numeroa, pilkun jälkeen yksi numero, ja muuttujan tyyppi on reaaliluku.



KUVA 45. FormatString

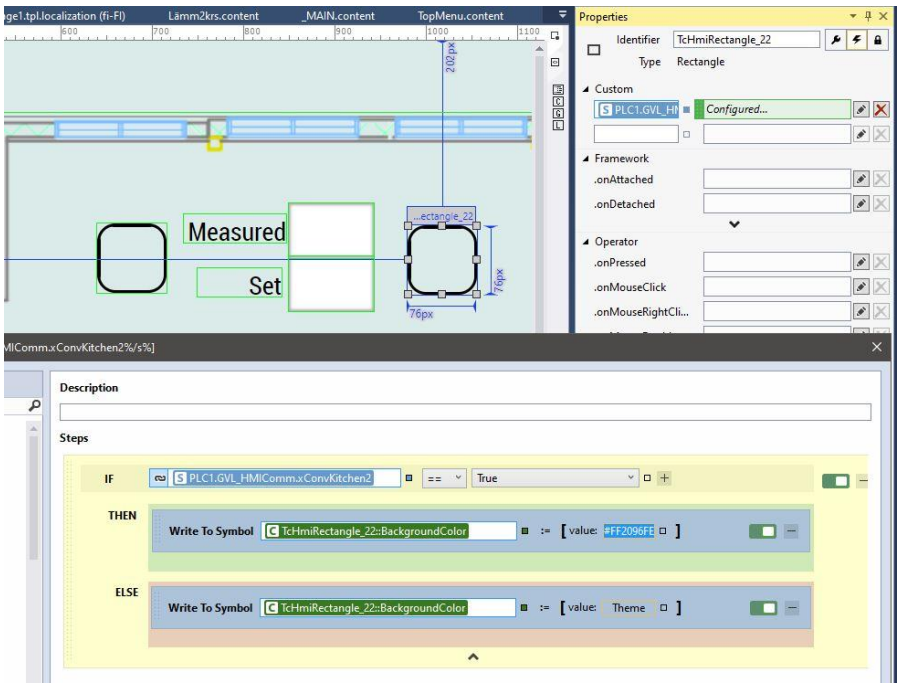
Kuten mainittua, muuttujat voivat olla myös ns. kaksisuuntaisia, eli niitä voidaan manipuloida HMI:lta. Editoimalla muuttujan linkkityypiksi *TwoWay* saadaan aikaan se, että muuttuja sekä näyttää arvoa, että voi muuttaa sen arvoa (kuva 46). Tapahtumaksi, jolla tämä muuttujan arvon muuttuminen tapahtuu, valittiin *.onUserInteractionFinished*. Tällöin arvoa muutetaan vasta, kun käyttäjä on

lopettanut toimintansa, eikä muuttujan arvo muutu jokaisella näppäimen painalluksella.



KUVA 46. Kaksisuuntainen muuttuja.

Ulkoisten PLC- ja sisäisten muuttujien sekaisin käyttöä ei ole millään tavalla estetty. Toimistoissa kattoihin asennettujen jäähdytyskonvektoreiden käyntitilaa voitiin siis monitoroida HMI:lta asettamalla PLC:ltä tuleva konvektorin ohjaustieto kirjoittamaan suorakulmio-objektin taustaväriä (kuva 47).

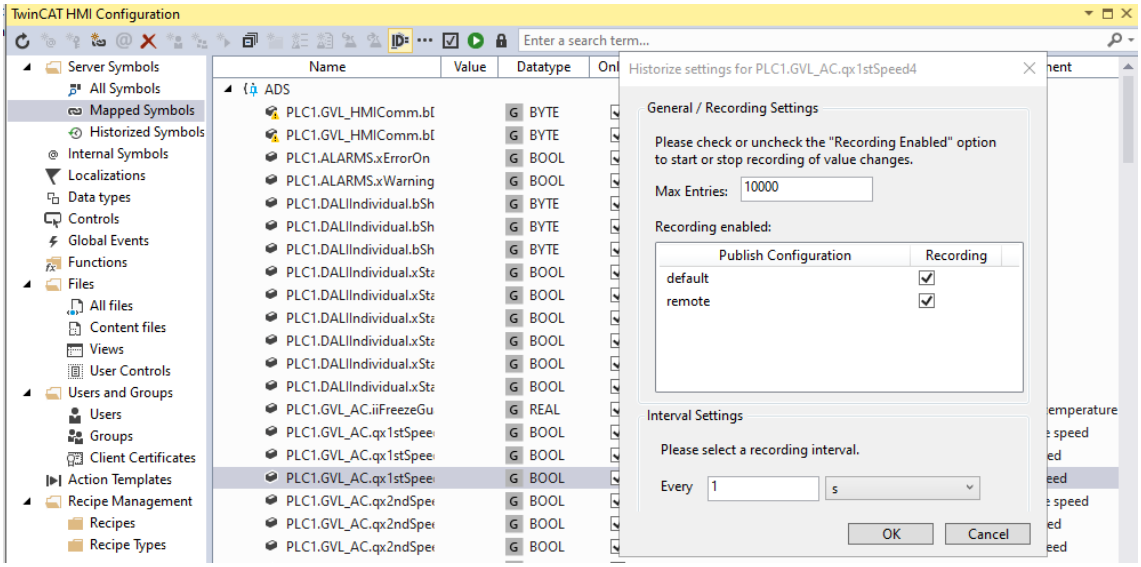


KUVA 47. Ulkoiset ja sisäiset muuttujat yhdessä

4.4.5 Datan historointi

HMI:lle haluttiin tieto joistakin mittausarvoista, kuten ulko- ja huonekohtaisista lämpötiloista, säätimien ulostuloista sekä ulkovaloisuudesta. Tätä tarkoitusta silmällä pitäen HMI-serverille voidaan *Historize*-toiminnolla tallentaa dataa. Historointi on eräänlainen tietokanta, josta käyttäjä ei tosin näe kuin lopputuloksen graafisessa muodossa.

Historointi muuttujaan voidaan tehdä kartoitetusta muuttujasta valitsemalla sen kohdalla hiiren oikealla *Historize Settings*, josta aukeaa kuvan 48 mukainen asetusikkuna. Ikkunasta määritellään miten monta datapistettä siitä tallennetaan sekä tallennusintervalli. Määrittelyn jälkeen historoitu muuttuja näkyy valikossa *Historized Symbols*, josta voidaan lukea myös edellä asetettujen datapistemäärän ja tallennusintervallin lisäksi sen datatyyppi, jonka HMI-serveri tunnistaa automaattisesti.

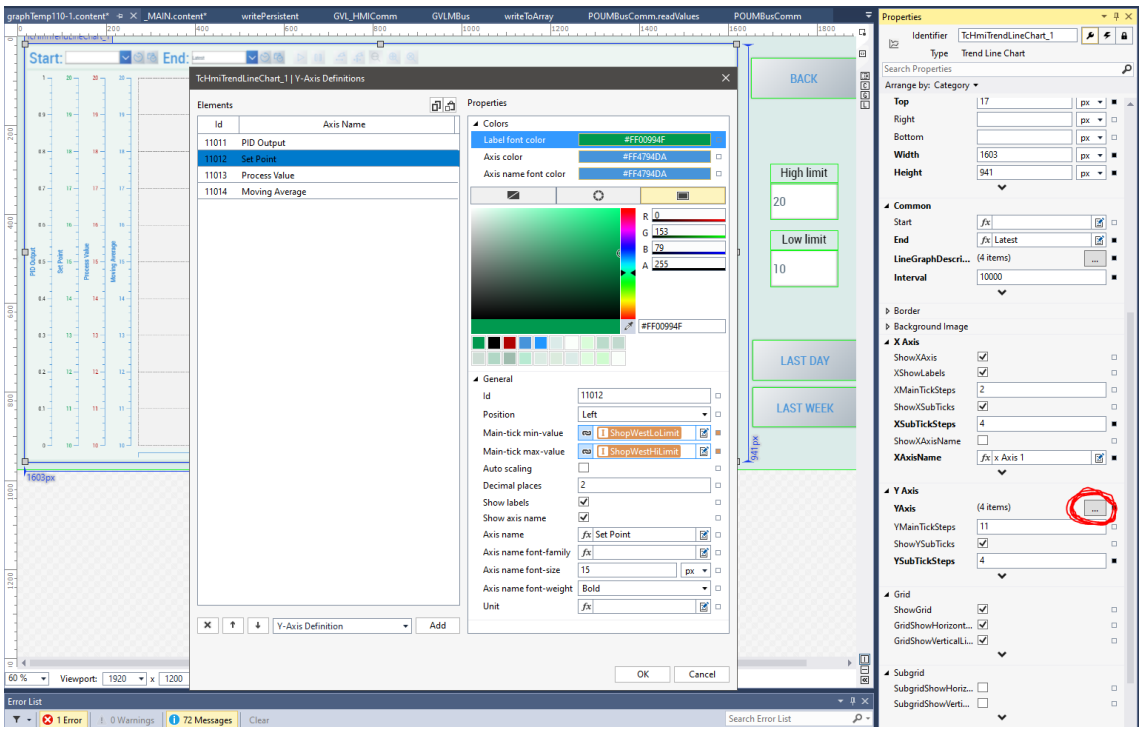


KUVA 48. Historointiasetukset

4.4.6 Graafit

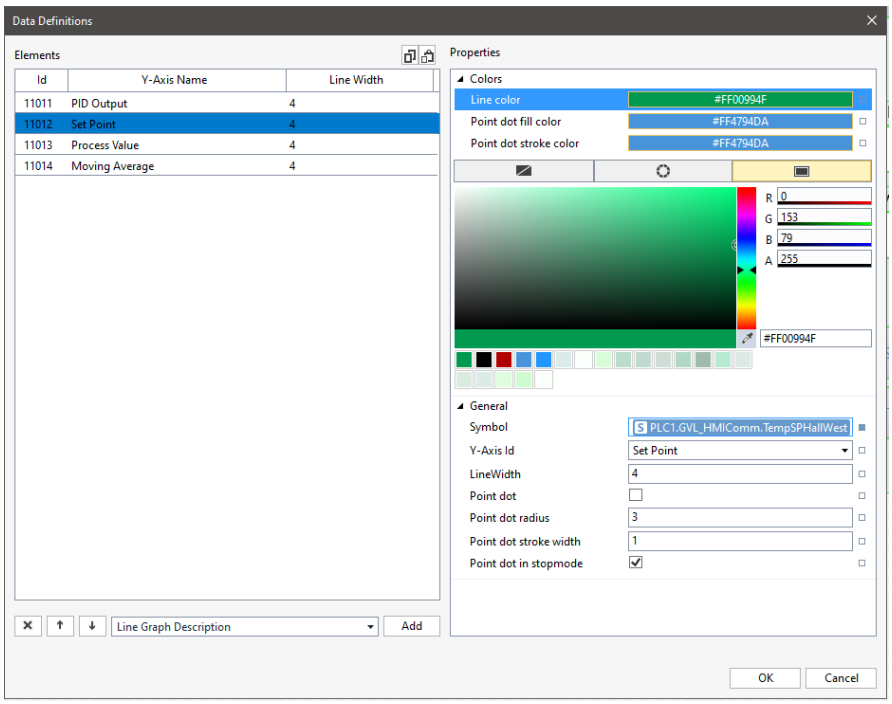
Kuvaajien käyttöön on HMI-editorissa kaksi mahdollisuutta: *Line Chart* ja *Trend Line Chart*, jotka löytyvät *Toolbox*-valikosta. Ensinmainittu on tarkoitettu ensisijaisesti reaaliaikaisen tiedon näyttämiseen, jälkimmäinen on nimenomaisesti tarkoitettu historoidulle datalle.

Jotta useita arvoja voidaan näyttää yhdessä kuvaajassa, tulee sen Y-akseleille antaa ensin muotoiluasetukset (kuva 49). Y-akselin asetuksiin päästään valitsemalla kuvaaja aktiiviseksi ja valitsemalla oikealta *YAxis*. Jokaiselle Y-akselille on annettava ainakin nimi ja Id-numero. Lisäksi akselit voidaan asettaa automaattisesti skaalautuviksi, tai niiden skaalan ylä- ja alarajat voidaan antaa joko suoraan numeroina tai muuttujan avulla (kuva 51).

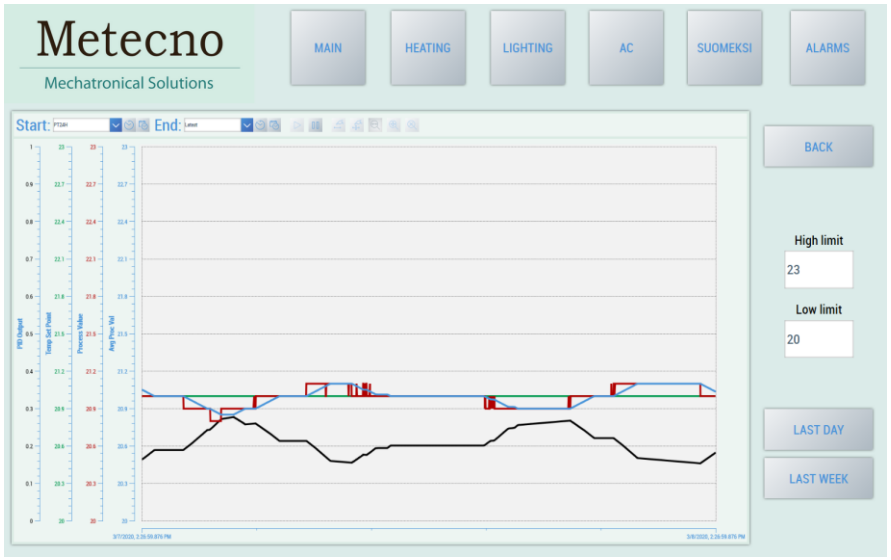


KUVA 49. Y-akseliasetukset

Kun muotoiluasetukset on tehty, täytyy akselit vielä liittää muuttujiin. Tämä tapahtuu asetusvalikosta *LineGraphDescription* (kuva 50), joka löytyy oikealta kohdan *Common* alta. Tässä valikossa kohdasta *Symbol* valitaan se historoitu muuttuja, jota halutaan näyttää. *Y-Axis Id* määrittää, mihin muotoiluasetukseen muuttuja liitetään. Tuloksena on kuvan XX kaltainen kuvaaja. Kuvaajan alkupiste voidaan määrittää kirjoittamalla sen parametriä *[kuvaajan ID] -> Common -> Start*. Alkuliite on aina PT, jonka jälkeen kirjoitetaan numero ja sen perään se aikamääre, joka halutaan. Esimerkiksi millisekunti on MS, sekunti S, tunti H ja päivä D. Jos siis halutaan näyttää viimeinen vuorokausi, voidaan muuttujaan kirjoittaa joko arvo PT24H tai PT1D.



KUVA 50. LineGraphDescription

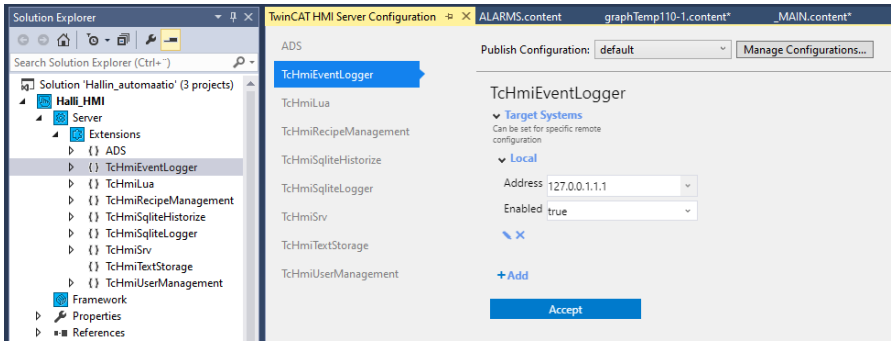


KUVA 51. Kuvaaja

4.4.7 Hälytykset

Kun hälytykset ja viestit on PLC:n puolella määritelty, on ne hyvin helppoa tuoda HMI:lle näkyviin. Ainut määrittely joka tarvitsee tehdä, on määritellä sen laitteen osoite, jonka *Event Loggeriin* halutaan viitata. Tämä tapahtuu määrittelemällä laitteen osoite serverin asetuksissa kuvan XX esittämällä tavalla. Koska tässä

työssä HMI-serveri ja PLC-kernel toimivat samassa laitteessa, annetaan lokaali osoite 127.0.0.1.1.1.



KUVA XX. Event Loggerin osoitteen määrittely

Näiden asetuksien jälkeen voidaan hälytykset näyttää työkalua *Event Grid* apuna käyttäen. Huomionarvoista on, ettei *Event Grid* tarjoa hälytysten nollaamiseen keinoa, vaan sitä tarkoitusta varten on tehtävä erillinen näppäin tai muu toiminto.

4.4.8 Tabletin käyttöönotto

Koska Beckhoffin yksi myyntipuheista on, että HMI:ta voidaan käyttää millä tahansa selaimella, työn yhteydessä ostettiin kolme kappaletta tablettitietokoneita HMI:n käyttöä varten. Tässä oli kuitenkin heti ongelmia: Android-tabletit mm. pitävät itsepintaisesti ylä- ja alavalikoitaan näkyvissä, mikä pienentää HMI:n näyttöön tarkoitetun alueen resoluutiota. Myös HMI:n asettaminen koko näytön tilaan osoittautui hankalaksi. Normaalilla Windows-pohjaisella järjestelmällä toimiva selain voidaan käynnistysasetuksissa asettaa aukeamaan ns. kiosk-tilaan, jolloin se on käynnistymisestään lähtien koko ruudun tilassa. Android-tableteissa asennetut ohjelmat taasen asentuvat sellaiseen kansioon, johon ei päästä käsiksi ilman laitteen root-oikeuksia, joten käynnistysparametreja ei päästä editoimaan. Root-oikeuksia ei myöskään Androidin versiosta 8 eteenpäin saada käyttöön.

Joitakin ominaisuuksia päästiin vaihtamaan työkalulla *Android Debug Bridge*. Tämän työkalun käyttöä varten tarvitsi Androidista ottaa käyttöön sovelluskehittäjän asetukset, joka onnistuu asetuksista navigoimalla ohjelmistotietoihin ja näpyttämällä kohtaa *koontiversio (build number)* kahdeksan

kertaa. Tämän jälkeen otettiin käyttöön *USB-vianetsintä (USB Debugging)*, jotta saatiin yhteys USB-johdon kautta tablettiin.

Ylä- ja alavalikot saatiin piiloon antamalla adb:lla tabletille käsky `adb shell settings put global policy_control immersive.full=*`. Myös muita toimintoja yritettiin saada pois, kuten virransäästötoimintoa, joka kadottaa verkkoyhteyden lähes välittömästi näytön mentyä pois päältä. Tämä parametri löytyy edelleen listasta, jota voidaan manipuloida, mutta sen muuttamisella ei ole mitään vaikutusta. Oletetaan, että se on suljettu root-oikeuksien taakse. Tämän parametrin asettaminen tilaan *always on* olisi ollut tärkeää, sillä näytön sammutettua itsensä ja käyttäjän tämän jälkeen sen jälleen avattua, joutuu käyttäjä odottamaan, että selain ottaa jälleen yhteyden HMI-serveriin.

Koko näyttöön tabletti saatiin asentamalla Microsoftin Edge-selain, josta löytyi *Lisää aloitusnäyttöön* -ominaisuus. Menemällä selaimella HMI:n IP-osoitteeseen, kääntämällä tabletti vaaka-asentoon ja valitsemalla *Lisää aloitusnäyttöön* teki ominaisuus tabletin työpöydälle kuvakkeen, jota painamalla voidaan mennä suoraan HMI:n osoitteeseen, ja se on automaattisesti koko näytön tilassa.

Tabletin oma näppäimistö piti myös saada pois käytöstä. Mikäli näppäimistö olisi jätetty käyttöön, olisi se vienyt suuren osan näytön pinta-alasta aina, kun vaikkapa olisi haluttu säätää huonelämpötilan asetusarvoa. Play-kaupasta löytyi kuitenkin ilmainen näppäimistö *NullInput Method*. Tämä näppäimistö on tehty ulkoisia USB-näppäimistöjä varten, ja se käytännössä avaa ruudulle näkymättömän näppäimistön. Se otettiin käyttöön, ja käytettiin HMI:ssa Beckhoffin omaa numeronäppäimistöä.

5 YHTEENVETO

Kokonaisuutena arvioituna työssä saavutettiin tavoitteet; valaistus ja muut toiminnallisuudet saatiin toimimaan automaattisesti. Kompressorit, päävesisulut ja ilmastointilaitteet saatiin vaihtamaan tilojaan hälytysjärjestelmän tilatiedon avulla, eikä työntekijöiden aikaa enää tarvitse hukata asioiden muisteluun tai niistä huolehtimiseen.

Beckhoffin laaja komponenttivalikoima osoittautui hyödylliseksi sähkösuunnittelun kannalta. Tilaa vievistä riviliitinjonoista päästiin eroon, eikä verrattain laajan automaation sähkökeskuksiin mahduttamisesta koitunut tilallisia ongelmia. Valmistajan lisenssikäytännöt tosin poikkesivat merkittävästi kilpailijoista, joskin niiden mukana tuli myös monipuolisuus ja edullisuus.

Työn laajuuden vuoksi monia asioita ja mahdollisuuksia jäi tutkimatta, ja mm. DALI-väylällä toteutettu valaistus olisi hyvinkin voinut yhtenä kokonaisuutena olla kokonaisen lopputyön täyttävä suoritus. Nyt asiat toteutettiin helpoimman kautta, joka tulevaisuuden modifiointeja silmällä pitäen voi olla hankalaa. Esim. ulkovalaistuksen kirkkauden olisi voinut toteuttaa scene-toiminnolla, jolloin eri kirkkausasteita olisi voitu tulevaisuudessa tehdä useampia. Väylä olisi tarjonnut myös mahdollisuuksia toteuttaa automaattista vianhakua, jolloin käyttöliittymään olisi voinut tulla tietoa vaikkapa viallisista valaisimista.

Huomionarvoisin seikka oli kuitenkin Android-tablettien lopullinen soveltumattomuus HMI:ksi. Androidista on – ymmärrettävistä syistä – tehty hyvin suljettu järjestelmä. Tämä estää toisaalta pahantahtoisten laitteen kaappaajien toiminnan, mutta myös tämän työn tapauksessa laitteen modifioinnin teollisuuteen sopivaksi HMI:ksi. Erityisesti virransäästöominaisuudet ja niiden muokkaamisen estäminen tekevät tabletin käytöstä hyvin hankalaa. Yksi vaihtoehto on tietysti asentaa ohjelma, joka pitää tabletin näytön päällä koko ajan, mutta tämä taas lyhentää tabletin näyttömoduulin käyttöikää merkittävästi. Ja jos tabletti saataisiinkin toimimaan halutun kaltaisesti, ei mikään lupaa, etteikö jonkin tulevaisuuden ohjelmistopäivityksen yhteydessä näitä käytettyjä metodeita otettaisi pois toiminnasta.

LÄHTEET

Yuichi, M. 2015. Data-variant Kernel Analysis. John Wiley & Sons, Inc. Hoboken, New Jersey.

Roch, B. 2004. Monolithic kernel vs. Microkernel. Technical University Wien.

Sands, N. Verhappen, I. 2018. Guide to the Automation Body of Knowledge. ISA.

Merz, H. Hansemann, T. Hübner, C. 2009. Building Automation – Communication Systems with EIB/KNX, LON and BACnet. Springer-Verlag. Berlin Heidelberg.

Yuejun, Z. Ping, Z. Mingguang, W. 2018. Research on DALI and Development of Master-Slave module. 2006 IEEE International Conference on Networking, Sensing and Control.

Hein, P.F. 2001. DALI – A Digital Addressable Lighting Interface For Lighting Electronics. Conference Record of the 2001 IEEE Industry Applications Conference. 36th IAS Annual Meeting.

Beckhoff Automation GmbH. 2019. TwinCAT 3 PLC Lib: Tc3_DALI. https://download.beckhoff.com/download/Document/automation/twincat3/TwinCAT_3_PLC_Lib_Tc3_DALI_EN.pdf

Beckhoff Automation GmbH. 2010. Application Note DK9322-0810-0036: M-Bus connection for energy and consumption meters via TwinCAT. http://ftp.beckhoff.com/download/Document/Application_Notes/DK9322-0810-0036.pdf

Beckhoff Automation GmbH. 2019. TwinCAT: PLC and Motion Control on the PC. https://download.beckhoff.com/download/Document/Catalog/Main_Catalog/english/Beckhoff_Main_Catalog_2019_1_06_TwinCAT.pdf

Beckhoff Automation GmbH. Julkaisuvuosi tuntematon. ADS Communication. https://infosys.beckhoff.com/english.php?content=../content/1033/bc9xx0/3740857483.html

Beckhoff Automation GmbH. Julkaisuvuosi tuntematon. ADS Protocol. https://infosys.beckhoff.com/english.php?content=../content/1033/bc9000/4351356427.html

Beckhoff Automation GmbH. 2019. PLC Lib: Tc2_IoFunctions. https://download.beckhoff.com/download/document/automation/twincat3/TwinCAT_3_PLC_Lib_Tc2_IoFunctions_en.pdf

Beckhoff Automation GmbH. 2020. PLC Lib: Tc2_Utilities. https://download.beckhoff.com/download/document/automation/twincat3/TwinCAT_3_PLC_Lib_Tc2_Utilities_en.pdf

Mix, J. 2006. HVAC Efficiency Definitions. http://www.usaireng.com/pdfs/efficiency-definitions.pdf

PLCopen. 2016. Coding Guidelines. PLCopen.

SESKO ry. 2011. Valo ja valaistus. Työkohteiden valaistus. Osa 1: sisätilojen työkohteiden valaistus. Suomen standardoimisliitto SFS.

LIITTEET

Liite 1. I/O-lista

I/O-lista

Nimi	Kortti	Kanava
PK		
Sähköauton lataus	KL10	DI1
Sähköauton lataus	KL10	DI2
Kompressori	KL10	DI3
Nosto-ovi 1	KL10	DI4
Nosto-ovi 2	KL10	DI5
Siltanosturi	KL10	DI6
Sulanapitolämmitys	KL10	DI7
	KL10	DI8
Ovikello	KL10	DI9
	KL10	DI10
	KL10	DI11
	KL10	DI12
	KL10	DI13
	KL10	DI14
	KL10	DI15
	KL10	DI16
Sähköauton lataus	KL20	DO1
Sähköauton lataus	KL20	DO2
Kompressori	KL20	DO3
Nosto-ovi 1	KL20	DO4
Nosto-ovi 2	KL20	DO5
Siltanosturi	KL20	DO6
Sulanapitolämmitys	KL20	DO7
	KL20	DO8
Lattialämmityspiiri A110/3	KL21	DO1
Lattialämm. A110/4	KL21	DO2
Lattialämm. A110/5	KL21	DO3
Lattialämm. A110/8	KL21	DO4
Lattialämm. A110/9	KL21	DO5
Lattialämm. A110/7	KL21	DO6
	KL21	DO7
	KL21	DO8
DALI communication	KL30	D1 + / -
	KL30	D2 + / -
IVK		
Sisääntuloilman venttiili kiinni	KL10	DI1
Sisääntuloilman venttiili auki	KL10	DI2
Ulosmenoilman venttiili kiinni	KL10	DI3
Ulosmenoilman venttiili auki	KL10	DI4

Sisääntulon tamun vika	KL10	DI5	
Ulosmenon tamun vika	KL10	DI6	
Lämpöpatteri	KL10	DI7	
Lämmön talteenotto	KL10	DI8	
Sisääntulon tamu forward	KL20	DO1	
Ulosmenon tamu forward	KL20	DO2	
Lämpöpatteri	KL20	DO3	
Lämmön talteenotto	KL20	DO4	
	KL20	DO5	
	KL20	DO6	
	KL20	DO7	
	KL20	DO8	
Jäätymisvahti	KL30	R1+ / R1-	vastusanturi
	KL30	R2+ / R2-	
	KL30	R3+ / R3-	
	KL30	R4+ / R4-	
	KL30	R5+ / R5-	
	KL30	R6+ / R6-	
	KL30	R7+ / R7-	
	KL30	R8+ / R8-	
Sisääntulon tamu, taajuusohje	KL40	AO1	
Ulosmenon tamu, taajuusohje	KL40	AO2	
Sisääntuloilman venttiili auki	KL21	DO1	
Ulosmenoilman venttiili auki	KL21	DO2	
Suodatinvahti, ensiö	KL31	X1	paine
Suodatinvahti, toisio	KL31	X2	paine

JK101

MLP vika	KL10	DI1	
MLP lisälämpöpyyntö	KL10	DI2	
MLP paisuntavent. 1 vika	KL10	DI3	
MLP paisuntavent. 2 vika	KL10	DI4	
IV 1.krs kriittinen vika	KL10	DI5	
IV 1.krs huoltohälytys	KL10	DI6	
Palo/rikos, vika	KL10	DI7	
Palo/rikos, ON/OFF	KL10	DI8	
Palo-/rikoshälytys	KL11	DI1	
Kulunvalvonta, vika	KL11	DI2	
Kondensiopumppu 1, vika	KL11	DI3	
Kondensiopumppu 2, vika	KL11	DI4	
Lattialämmityspumppu 1, vika	KL11	DI5	
Lattialämmityspumppu 2, vika	KL11	DI6	
IV 1.krs, nopeus 1	KL11	DI7	

IV 1.krs, nopeus 2...4	KL11	DI8
LVV vastukset	KL12	DI1
	KL12	DI2
	KL12	DI3
Jäähdytyspumppu 1	KL12	DI4
Jäähdytyspumppu 2	KL12	DI5
	KL12	DI6
	KL12	DI7
	KL12	DI8
IV HÄTÄSEIS	KL12	DI9
	KL12	DI10
	KL12	DI11
	KL12	DI12
	KL12	DI13
	KL12	DI14
	KL12	DI15
	KL12	DI16
LVV vastukset	KL20	DO1
	KL20	DO2
	KL20	DO3
Jäähdytyspumppu 1	KL20	DO4
Jäähdytyspumppu 2	KL20	DO5
	KL20	DO6
	KL20	DO7
	KL20	DO8
Pääliuospumppu	KL21	DO1
Lattialämm. A105.1	KL21	DO2
Lattialämm. A106.1	KL21	DO3
	KL21	DO4
	KL21	DO5
	KL21	DO6
	KL21	DO7
	KL21	DO8
M-Bus communication	KL30	MBUS+
M-Bus communication	KL30	MBUS-
Päävesisulku	KL40	DO1
	KL40	DO2
IV 1.krs nopeus 2	KL35	DO1
IV 1.krs nopeus 3	KL35	DO2
IV 1.krs nopeus 4	KL36	DO1
	KL36	DO2

JK102

Lattialämm. A103	KL10	DO1
Lattialämm. A102	KL10	DO2
Lattialämm. A111	KL10	DO3
Lattialämm. A100-3	KL10	DO4
Lattialämm. A100-2	KL10	DO5
	KL10	DO6
	KL10	DO7
	KL10	DO8
TE A105.1	KL20	R1+ / R1-
TE A106.1	KL20	R2+ / R2-
TE A102	KL20	R3+ / R3-
TE A111	KL20	R4+ / R4-
TE A100.1	KL20	R5+ / R5-
TE A100.2	KL20	R6+ / R6-
TE A110.1	KL20	R7+ / R7-
TE A110.2	KL20	R8+ / R8-
DALI communication	KL30	D1 + / -
	KL30	D2 + / -
Jäähdytyskonvektori 1	KL50	DI1
Jäähdytyskonvektori 2	KL50	DI2
Jäähdytyskonvektori 3	KL50	DI3
	KL50	DI4

JK201

Lattialämm. A201/2	KL10	DO1
Lattialämm. A202	KL10	DO2
Lattialämm. A207/2	KL10	DO3
Lattialämm. A207/3	KL10	DO4
Lattialämm. A207/4	KL10	DO5
Lattialämm. A207/5 + 206	KL10	DO6
	KL10	DO7
	KL10	DO8
	KL11	DO1
	KL11	DO2
	KL11	DO3
	KL11	DO4
	KL11	DO5
	KL11	DO6
	KL11	DO7
	KL11	DO8
UPS vika	KL15	DI1
UPS / verkko	KL15	DI2
Kamerajärj. vika	KL15	DI3

IV 2.krs kriittinen vika	KL15	DI4
IV 2.krs huoltohälytys	KL15	DI5
IV 2.krs nopeus 1	KL15	DI6
IV 2.krs nopeus 2…4	KL15	DI7
	KL15	DI8
TE A207.1	KL20	R1+ / R1-
TE A207.2	KL20	R2+ / R2-
TE A202	KL20	R3+ / R3-
TE A201	KL20	R4+ / R4-
TE ulko, pohj.seinä	KL20	R5+ / R5-
	KL20	R6+ / R6-
	KL20	R7+ / R7-
	KL20	R8+ / R8-
M-Bus communication	KL30	MBUS+ / -
Ulkovaloisuus	KL40	AI1
	KL40	AI2
DALI communication	KL50	D1+ / -
	KL50	D2 + / -
IV 2.krs nopeus 2	KL30	DO1
IV 2.krs nopeus 3	KL30	DO2
IV 2.krs nopeus 4	KL31	DO1
	KL31	DO2
Jäähdytyskonvektori 1	KL70	DI1
Jäähdytyskonvektori 2	KL70	DI2
Jäähdytyskonvektori 3	KL70	DI3
Jäähdytyskonvektori 4	KL70	DI4
Jäähdytyskonvektori 5	KL80	DI1
	KL80	DI2
	KL80	DI3
	KL80	DI4

Liite 2. KL3208 anturityypit (Beckhoff, 56)

iSetSensorType	Element
0	PT1000
1	NI1000
2	RSNI1000 (NI1000 based on Landis & Staefa characteristics: 1000 Ω at 0 °C and 1500 Ω at 100 °C)
3	NTC1K8
4	NTC1K8_TK
5	NTC2K2
6	NTC3K
7	NTC5K
8	NTC10K
9	NTC10KPRE
10	NTC10K_3204
11	NTC10KTYP2
12	NTC10KTYP3
13	NTC10KDALE
14	NTC10K3A221
15	NTC20K
16	Potentiometer, resolution 0.1 Ω
17	Potentiometer, resolution 1 Ω
18	NTC100K

Liite 3. E_DALIAddressType (Beckhoff, 191)

```
TYPE E_DALIAddressType :
(
  Short            := 0,
  Group            := 1,
  Broadcast        := 2,
  BroadcastUnaddr  := 3
) BYTE := Short;
END_TYPE
```

Liite 4. Valaistuksen vähimmäisarvot (SESKO, 42, 54)

Taulukko 5.11 Teollisuus ja käsityö – Sähkö- ja elektroniikkateollisuus

Viitenro.	Tila, tehtävä tai toiminta	$\bar{E}_m$ lx	UGR_L –	U_o –	R_a –	Erityisvaatimukset
5.11.1	Kaapeleiden ja johtimen valmistus	300	25	0,60	80	
5.11.2	Käämintä					
	— suuret kelat	300	25	0,60	80	
	— keskikokoiset kelat	500	22	0,60	80	
	— pienet kelat	750	19	0,70	80	
5.11.3	Kelojen kyllästäminen	300	25	0,60	80	
5.11.4	Galvanointi	300	25	0,60	80	
5.11.5	Kokoonpanotyö					
	— karkea, esim. suuret muuntajat	300	25	0,60	80	
	— tavanomainen, esim. jakokeskukset	500	22	0,60	80	
	— hieno, esim. puhelimet, radiot, IT-laitteet (tietokoneet)	750	19	0,70	80	
	— tarkkuus, esim. mittalaitteet, piirilevyt	1 000	16	0,70	80	
5.11.6	Elektroniikkapajat, testaus, säätö	1 500	16	0,70	80	

Taulukko 5.26 Toimistot

Viitenro.	Tila, tehtävä tai toiminta	$\bar{E}_m$ lx	UGR_L –	U_o –	R_a –	Erityisvaatimukset
5.26.1	Arkistointi, kopiointi, jne.	300	19	0,40	80	
5.26.2	Kirjoittaminen, konekirjoitus, lukeminen, tietojenkäsittely	500	19	0,60	80	Tietokonenäytöt, katso 4.9
5.26.3	Tekninen piirtäminen	750	16	0,70	80	
5.26.4	CAD-työasemat	500	19	0,60	80	Tietokonenäytöt, katso 4.9
5.26.5	Neuvottelu- ja kokoushuoneet	500	19	0,60	80	Valaistuksen tulisi olla säädettävä.
5.26.6	Vastaanottotiski	300	22	0,60	80	
5.26.7	Arkistot	200	25	0,40	80	

Liite 5. TIMESTRUCT (Beckhoff, 296)

```
TYPE TIMESTRUCT
STRUCT
   wYear         : WORD;
   wMonth        : WORD;
   wDayOfWeek    : WORD;
   wDay          : WORD;
   wHour         : WORD;
   wMinute       : WORD;
   wSecond       : WORD;
   wMilliseconds : WORD;
END_STRUCT
END_TYPE
```

wYear: The year: 1970 ~ 2106;

wMonth: The month: 1 ~ 12 (January = 1, February = 2 etc.);

wDayOfWeek: The day of the week: 0 ~ 6 (Sunday = 0, Monday = 1 etc.);

wDay: The day of the month: 1 ~ 31;

wHour: Hour: 0 ~ 23;

wMinute: Minute: 0 ~ 59;

wSecond: Second: 0 ~ 59;

wMilliseconds: Millisecond: 0 ~ 999;