

Pyry Nieminen

PIENOISMALLIJUNAN PAIKANMÄÄRITYKSEN TOTEUTTAMINEN

Kandidaatintyö
Tekniikan ja luonnontieteiden tiedekunta
Tarkastaja: Mikko Salmenperä
Marraskuu 2023

TIIVISTELMÄ

Pyry Nieminen: Pienoismallijunan paikanmäärittäminen
Kandidaatintyö
Tampereen yliopisto
Teknisten tieteiden kandidaatin tutkinto-ohjelma
Marraskuu 2023

Automaatiojärjestelmissä on usein tarve mitata erilaisten fyysisten kappaleiden paikkaa reaali maailmassa. Vaihtoehtoisia anturitekologioita tähän tarpeeseen on monenlaisia. Tämän työn tavoitteena on toteuttaa CyberCity-kaupunkimalliin järjestelmä, joka määrittää millä rataosuudella pienoismallijuna sijaitsee ja näyttää tämän tiedon graafisessa käyttöliittymässä. CyberCity on Tampereen yliopiston kyberharjoitteluympäristö, joka koostuu pohjana olevasta kaupunkimallista ja useista siihen toiminnallisuutta lisäävistä järjestelmistä.

Tämä työ jakaantuu kahteen vaiheeseen. Ensimmäisessä vaiheessa vertaillaan yleisesti työhön mahdolliseksi sopivia anturitekologioita. Vertailut anturit voidaan jakaa binääriin antureihin, kuten mikrokytkimiin, ja analogisiin antureihin, kuten ultraääniantureihin.

Toisessa vaiheessa käydään läpi toteutettavan järjestelmän vaatimuksia ja valitaan niiden perusteella käytettävä anturi ja toteutetaan itse järjestelmä. Tärkeimpiä vaatimuksia järjestelmälle oli paikkatiedon mittaus ilman muun ympäristön häiritsemistä, anturoinnin sulautuminen kaupunkimalliin ja mahdollisuus purkaa ympäristö sen siirtämistä varten. Näiden vaatimusten perusteella päädyttiin Hall-efektitekniikkaa käyttävään anturiin.

Toteuttaessa järjestelmää suoritettiin ensimmäiseksi fyysinen osio, jossa anturit sijoitettiin paikoilleen ja kytkettiin kiinni käytettyyn Beckhoffin ohjelmoitavaan logiikkaan. Antureiden kytkennässä otettiin huomioon ympäristön purkamisvaatimus käyttämällä verkkokaapeleita ohjelmoitavan logiikan ja kaupunkipalojen välillä.

Seuraavassa vaiheessa ohjelmoitava logiikka ohjelmoitiin käyttäen Beckhoffin TwinCAT ohjelmaa ja IEC 61131-3 standardin määrittelemää Structured text-ohjelmointikieltä. Logiikkaohjelmoinnin jälkeen vuorossa oli käyttöliittymän toteuttaminen käyttäen hyväksi Windows Forms-käyttöliittymäkirjastoa. CyberCity-ympäristöön oli jo aiemmin toteutettu ohjauspaneelin tyyppinen käyttöliittymä, johon tämän työn käyttöliittymä liitettiin osaksi. Käyttöliittymä ja logiikkaohjelma käyttävät kaksin keskeiseen tiedonsiirtoon Beckhoffin ADS protokollaa, joka mahdollistaa muun muassa ohjelmoitavan logiikan muuttujien tapahtumapohjaisen lukemisen.

Työn aikana toteutettu järjestelmä vastaa aluksi asetettuja vaatimuksia. Jatkokehityksen aiheita voisi olla esimerkiksi olemassa olevan junanohjausjärjestelmän ja nyt toteutetun paikanmäärittäjäjärjestelmän yhteenliittäminen.

Avainsanat: paikannus, kyberharjoitteluympäristö, ohjelmoitava logiikka

Tämän julkaisun alkuperäisyys on tarkastettu Turnitin Originality Check -ohjelmalla.

SISÄLLYSLUETTELO

1. JOHDANTO	1
2. ANTURITEKNOLOGIOIDEN VERTAILU	3
2.1 Mikrokytkin	3
2.2 Ultraäänianturi	4
2.3 Hall-efektianturi	4
2.4 Optiset anturit	5
3. TOTEUTUS	6
3.1 Vaatimukset	6
3.2 Anturin valinta	6
3.3 Beckhoff automaatioympäristö	8
3.4 Kytkenät	8
3.5 Ohjelmointi	10
3.5.1 Ohjelmoitava logiikka	10
3.5.2 Käyttöliittymä	12
4. TULOKSET JA JATKOKEHITYS	14
4.1 Tulokset	14
4.2 Jatkokehityskohteita	15
5. YHTEENVETO	16
LÄHTEET	17

1. JOHDANTO

Nykypäivän automaatiojärjestelmissä on usein tarve saada selville eri kohteiden sijainteja. Riippuen järjestelmästä tämä voi tarkoittaa jatkuvaa tarkkaa sijainnin määrittystä, tai tietoa siitä, että kohde on jollakin alueella. Eri anturitekniologioita, joita sijainnin mittaamiseen voidaan käyttää, on paljon. Tämän työn tavoitteena on suunnitella ja toteuttaa järjestelmä, joka osaa kertoa, millä osalla rataa CyberCity-opetusympäristön Lego-juna sijaitsee.



Kuva 1. CyberCity-opetusympäristö.

CyberCity on Tampereen yliopiston kyberharjoitteluympäristö. Ympäristön pohjana on kuvassa 1 näkyvä Legoista rakennettu kaupunkimalli, jossa myös Lego-junarata sijaitsee. Ympäristöön on tämän jälkeen lisätty muita järjestelmiä, liittyen esimerkiksi valaistukseen ja junanohjaukseen. Nämä järjestelmät ovat olleet aikaisempien kandidaatintöiden aiheita. Valmiin CyberCity-ympäristön on tarkoitus pystyä havainnollistamaan kaupunki-infraan kohdistuvia kyberuhkia.

Käytännössä työssä on tarkoituksena jakaa kaupunkimallissa oleva rata neljään osaan, ja määrittää millä osalla näistä juna sijaitsee aina tietyllä hetkellä. Tämän mahdollistavien antureiden valinta on myös osa tätä työtä. Tuotettu tieto näytetään lopuksi käyttäjälle käyttöliittymässä, joka toteutetaan osana työtä.

Tämän työn luvussa 2 käydään ensimmäiseksi läpi työssä mahdollisesti käytettäviä anturitekologioita. Seuraavassa luvussa 3 käydään läpi toteutettavan järjestelmän vaatimuksia ja valitaan niiden perusteella käytettävä anturi. Lisäksi luvussa kuvataan itse järjestelmän toteuttaminen. Luvussa 4 käydään läpi saavutettuja tuloksia ja pohditaan mahdollisia jatkokehityksen kohteita. Lopuksi luvussa 5 esitetään yhteenveto.

2. ANTURITEKNOLOGIOIDEN VERTAILU

Anturit ovat tärkeä osa lähes jokaista automaatiojärjestelmää. Niiden avulla saadaan muutettua tietoa fyysisen maailman tilasta sellaiseen muotoon, jota ohjausjärjestelmät ja -piirit ymmärtävät. Tässä työssä tarvitaan tieto siitä, onko juna anturin lähellä. Tarkkuudeksi riittää, että juna on 5 cm säteellä anturista kun se havaitaan. Valitun anturin on oltava tarpeeksi pienikokoinen, jotta se voidaan integroida osaksi CyberCity-ympäristöä. Tässä kappaleessa vertaillaan muutamia työssä käyttökelpoisia anturitekniikoita. Valittu anturi esitellään alaluvussa 3.2.

Työssä haluttua tietoa tuottavat anturit ovat paikkaa, etäisyyttä ja läheisyyttä mittaavia antureita. Yksi tapa vertailla antureita on jakaa ne binääriin ja analogisiin. Binääristen anturien ulostulo on aina yksi kahdesta arvosta riippuen sen sisääntulosta. Analogisten antureiden ulostulo taas muuttuu tietyssä suhteessa sisääntuloon, jolloin on mahdollista mitata halutun suureen suuruutta, kuten etäisyyttä tai painetta. [1, s. 6]. Tässä työssä tarvitaan vain binäärinen tieto siitä, onko juna anturin lähellä. Mikäli päädytään käyttämään analogisia antureita, pitää ohjelmallisesti määrittää mittauksille raja-arvo, jonka ylitettäessä juna merkitään havaituksi.

Vertailuun valittiin antureita, joista on saatavilla tarpeeksi pienikokoisia versioita ja jotka mittaavat haluttuja suureita. Analogisten antureiden kohdalla haluttu suure on anturin etäisyys mitattavasta kohteesta ja binääristen antureiden kohdalla tieto siitä, onko tarkasteltava kohde lähellä. Se mikä etäisyys on lähellä, riippuu anturitekniikasta ja anturimallista. Vertailussa keskityttiin yleisesti käytössä oleviin tekniikoihin. Kyseessä ei siis ole kattava vertailu kaikista anturitekniikoista, joilla tämän työn voisi toteuttaa.

2.1 Mikrokytkin

Mikrokytkimet ovat pienikokoisia mekaanisia kytkimiä, jotka sulkevat tai avaavat virtapiirin, kun niihin kohdistuu ulkoinen voima. Niitä on yleisesti käytössä hyvin monenlaisissa elektronisissa laitteissa. Mikrokytkinten vahvuuksia on edullinen hinta ja yksinkertainen rakenne. Toisaalta kytkimet vaativat fyysisen kontaktin mitattavaan kohteeseen, mikä voi aiheuttaa mekaanista kulumista järjestelmän elinkaaren aikana. [1, s. 8]

2.2 Ultraäänianturi

Ultraäänianturien toiminta perustuu ultraääniaaltojen lähettämiseen ja vastaanottamiseen [2, s. 273]. Kun äänennopeus v mittaussympäristössä tunnetaan riittävän tarkasti, pystytään lentoaika t mittaamalla laskemaan mitattavan kohteen etäisyys L anturista käyttämällä kaavaa $L = \frac{vt}{2}$. Äänennopeuteen ilmassa vaikuttaa eniten lämpötila. Kuivassa ilmassa 20 °C lämpötilassa äänennopeus on 344 m/s [2, s. 745].

Yleisin tapa generoida ultraääniaaltoja on käyttää hyväksi pietsosähköisiä (engl. piezoelectric) materiaaleja, jotka muuttavat muotoaan, kun niihin kohdistuu ulkoinen sähkökenttä. Vastaavasti taas materiaaliin kohdistuva paine muodostaa jännitteen kappaleen vastakkaisille puolille. [3] Näin yhtä pietsosähköistä elementtiä voidaan tietyissä antureissa käyttää sekä ääniaaltojen lähetykseen että vastaanottamiseen [2, s. 274].

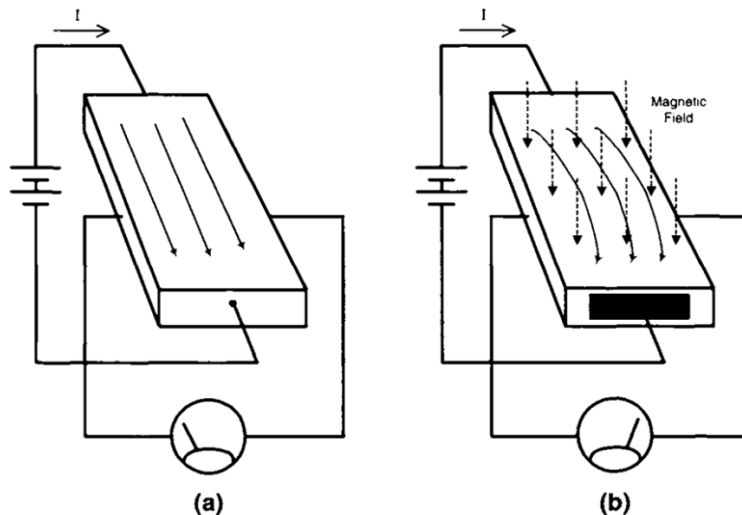
Ultraäänien etuna tutkissa käytettäviin radioaaltoihin verrattuna on niiden selvästi hitaampi nopeus. Näin lentoaika on ultraäänien kanssa huomattavasti pitempi, jolloin mittaus on helpompaa ja edullisempaa. [2, s. 274].

Yleinen käyttökohde ultraääniantureille on autojen peruutustutkat [2, s. 275]. Nykyisin on saatavilla useita valmiita anturimoduuleita, joissa on valmiiksi integroituna lähetin, ohjauselektronikka ja mahdollinen erillinen vastaanotin, jolloin anturin liittäminen muuhun järjestelmään on suoraviivaista.

2.3 Hall-efektianturi

Hall-efektianturit ovat yleisiä magneettikenttää havaitsevia antureita, joiden toiminta perustuu vuonna 1879 löydettyyn Hall-efektiin. Hall-efektillä tarkoitetaan ilmiötä, jossa ulkoinen magneettikenttä muuttaa elektronien kulkusuuntaa johtimessa [4]. Magneettikentän pitää olla kohtisuorassa sähkövirtaan nähden. Kun magneettikentän suunta vaihdetaan päinvastaiseksi, myös elektronien kulkusuunta vaihtuu.

Efektia voidaan havainnollistaa kuvan 2 kaltaisella koejärjestelyllä. Kohdassa **a** ohuessa johtavassa levyssä kulkee sähkövirta ja kuvan mukaisesti vastakkaisille puolille kytketty jännitemittari näyttää mitatuksi jännitteeksi 0 V. Kohdassa **b** levyyn kohdistuu virran suuntaan kohtisuora magneettikenttä, jolloin levyyn muodostuu jännite poikittain virran suuntaan nähden. Mikäli magneettikentän napaisuus vaihdetaan, myös muodostuvan jännitteen polariteetti vaihtuu [4, s. 1-2].



Kuva 2. Hall-efekti ohuessa johtavassa levyssä. [4, s. 1]

Hall-efektiantureita käytetään yleisesti kiinnittämällä magneetti kiinni haluttuun kohteeseen, jolloin sen magneettikenttä voidaan havaita kohteen lähestyessä anturia. Hall-efektianturit voidaan jakaa analogisiin antureihin, joiden ulostulo on verrannollinen magneettikentän voimakkuuteen, ja binäärisiin antureihin, joiden ulostulo on 1 tai 0 riippuen asetetun magneettikentän voimakkuuden raja-arvosta [4, s. 62].

Hall-efektiantureiden etuja on pieni koko, kestävyys vaativissa olosuhteissa ja alhainen hinta [4, s. xii-xiii]. Niitä käytetään yleisesti läheisyysantureina.

2.4 Optiset anturit

Optiset anturit sisältävät suuren joukon antureita, jotka käyttävät hyväksi erilaisia fyysisiä ilmiöitä, kuten polarisaatiota ja interferenssiä. Yleensä optinen anturi tarvitsee ainakin valonlähteen, valoa havaitsevan valokennon ja valoa ohjaavia rakenteita, kuten linssejä tai peilejä [2, s. 362].

Yksinkertaisimmillaan optinen anturi havaitsee kohteen lähellä olon mittaamalla siitä heijastunutta valon määrää ja vertaamalla sitä tiettyyn raja-arvoon. Tällaista yksinkertaista anturia on mahdollista parantaa esimerkiksi käyttämällä anturissa kahta kohtisuoraan asennettua polarisaatiosuodatinta, jolloin anturi ei enää reagoi heijastuksiin metallisista materiaaleista [2, s. 364].

Optisten antureiden vahvuuksia ovat niiden yksinkertaisuus, immuunius magneettisille häiriöille ja tarvittaessa suhteellisen pitkä (35 m) toimintamatka [2, s. 362] [1, s. 8]. Tyypillinen käyttökohde optiselle anturille on esimerkiksi kosketusvapaa hana.

3. TOTEUTUS

Tässä luvussa käydään ensin läpi vaatimuksia toteutettavalle järjestelmälle, jonka jälkeen valitaan niiden perusteella käytettävä anturi. Tämän jälkeen luvussa kuvataan järjestelmän toteuttaminen kytkentöjen ja ohjelmoinnin saralta.

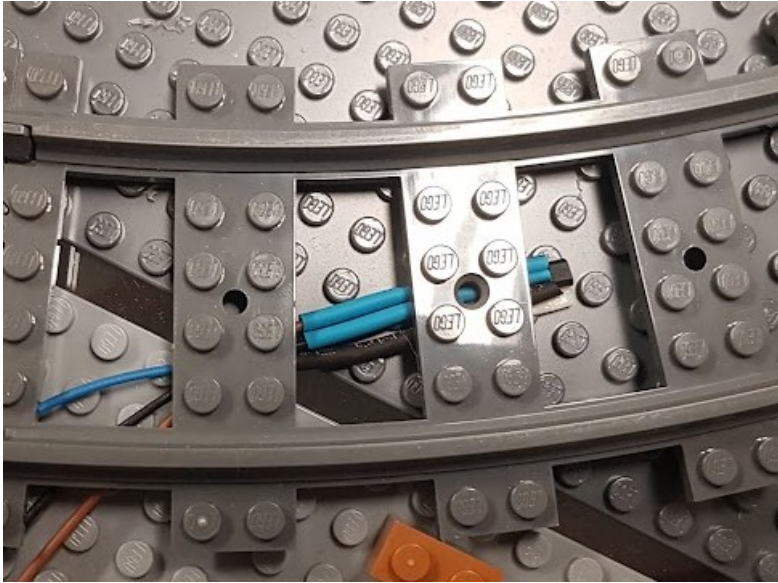
3.1 Vaatimukset

Toteutettavan järjestelmän pitää noudattaa useita eri vaatimuksia. Tärkein vaatimus on kyky määrittää, millä neljästä rataosuudesta juna sijaitsee. Järjestelmän on lisäksi esitettävä tämä tieto käyttöliittymän avulla käyttäjälle. Se ei myöskään saa häiritä jo olemassa olevaa CyberCity-ympäristöä, esimerkiksi vaikuttamalla junan kulkuun.

Muita huomioon otettavia asioita on CyberCity-ympäristön ominaisuuksien täydentäminen tulevaisuudessa, toteutetun järjestelmän sulautuminen ympäristöön ja sen ylläpidettävyys. Lisäksi toteutetun järjestelmän pitää edelleen mahdollistaa CyberCity-ympäristön purkamisen neljään palaan. Purkamista ja ylläpidettävyyttä helpottaa esimerkiksi eri johtojen merkitseminen nimilapuilla. Tulevaisuuden ominaisuuksien täydentymistä voidaan tukea toteuttamalla järjestelmä niin, että ulkopuolisten järjestelmien on helppo hyödyntää sen tuottamaa tietoa.

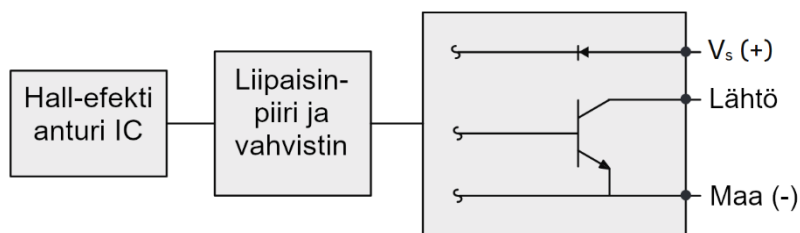
3.2 Anturin valinta

Käytettäväksi anturiteknologiaksi valittiin vaatimusten perusteella Hall-efektianturi. Sen valintaa puolsi kontaktiton toiminta, pieni koko ja helppokäyttöisyys. Kontaktiton toiminta mahdollistaa vapaamman anturin sijoittamisen ja välttää junan kulun häiritsemisen. Anturin pieni koko taas mahdollistaa sen sijoittamisen kiskojen väliin kuvan 3 mukaisesti. Tällöin myös tarvittava magneetti voidaan asentaa junan pohjaan, jolloin visuaalinen kokonaisvaikutus jää vähäiseksi.



Kuva 3. SS441A anturi sijoitettuna kiskojen väliin testaamista varten (reikä johtoja varten vielä poraamatta).

Tarkemmin anturiksi valittiin Honeywell SS441A, joka on yksinapainen (engl. unipolar) Hall-efektianturi 3,8–30 V syöttöjännitteellä [5]. Yksinapaisen anturin lähtö on aktiivinen, kun sen lähellä on magneettinen napa, ja ei-aktiivinen kun napaa ei ole lähellä [4, s. 228]. SS441A:n tapauksessa havaittava napa on magneetin S-napa. Lähdön ollessa aktiivinen anturin sisäinen transistori yhdistää anturin lähdön maahan kuvan 4 mukaisesti. Kun lähtö ei ole aktiivinen, se ei ole yhdistettynä kumpaankaan jännitepotentiaaliin ja vaatii siten ulkoisen ylösvetovastuksen (engl. pull-up resistor). [5] Ylösvetovastus ja muut kytkentöihin liittyvät asiat käydään tarkemmin läpi luvussa 3.4.



Kuva 4. SS441A lohkokaavio, muokattu lähteestä [5].

SS441A:n valitsemista puoltavia asioita oli yksinapaisuuden lisäksi laaja syöttöjänniteväli. Sallitun syöttöjännitteen ollessa 3,8–30 V, anturi toimii mikroprosessorien ja yhden piirin tietokoneiden yleisesti käyttämällä 5 V jännitteellä, sekä ohjelmoitavan logiikan yleisesti käyttämällä 24 V jännitteellä. SS441A aktivoiva

magneettivyyden tiheys oli lisäksi SS440 sarjan pienin, mikä mahdollistaa suuremman välimatkan magneetin ja anturin välillä, tai heikomman magneetin käyttämisen. [5]

3.3 Beckhoff automaatioympäristö

Järjestelmän toteuttamisessa käytettiin CyberCity-ympäristössä jo valmiiksi ollutta Beckhoff automaatioympäristöä. Automaatioympäristö koostuu Beckhoffin valmistamasta teollisuustietokoneesta ja siihen kytkettävistä input-outputmoduuleista. Teollisuustietokoneessa ajetaan Beckhoff TwinCat-ohjelmistoa, joka mahdollistaa reaaliaikaisten ohjelmitavien logiikoiden ja muiden ohjelmamoduulien suorittamisen Windows-käyttöjärjestelmässä [6]. Yleiskäyttöisen käyttöjärjestelmän käyttö mahdollistaa tarvittaessa myös muiden ohjelmien, kuten tietokantojen tai käyttöliittymien, käyttämisen samalla tietokoneella.

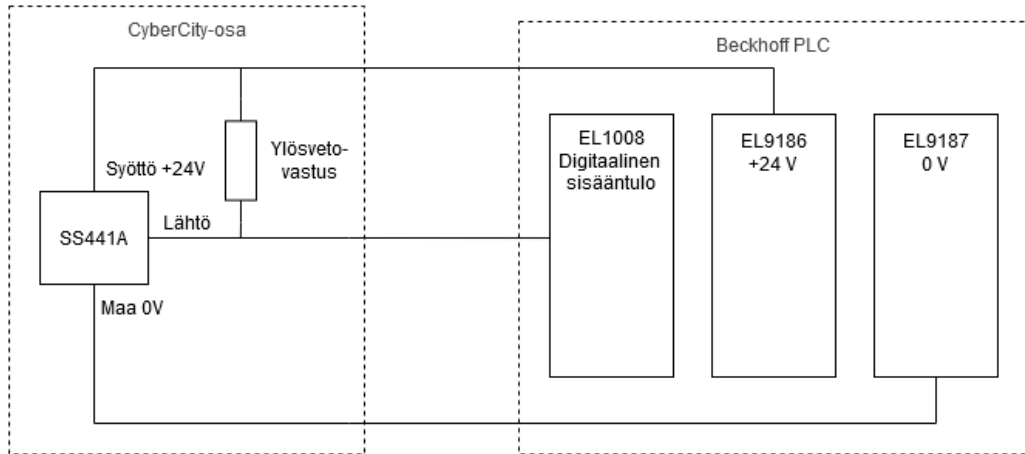
Ympäristössä on valmiina tarvittavat input-outputmoduulit antureita varten. EL9186 on virranjakomoduuili, jossa on kahdeksan 24 V liitäntää [7]. Se toimii yhteistyössä moduulin EL9187 kanssa, jossa on kahdeksan 0 V liitäntää [8]. Anturin lähdöt yhdistetään EL1008 moduuliin, joka on digitaalinen inputmoduuli kahdeksalla binäärisellä 24 V ohjaussignaalilla. EL1008 välittää saamansa ohjaussignaalit sähköisesti eristettynä ohjaustietokoneelle EtherCAT kenttäväylän kautta. [9]

Yhteenvedona syöttöjännite 24 V saadaan EL9186 moduulilta, maapotentiaali saadaan EL9187 moduulilta ja antureiden lähdöt yhdistetään EL1008 moduuliin.

3.4 Kytkennät

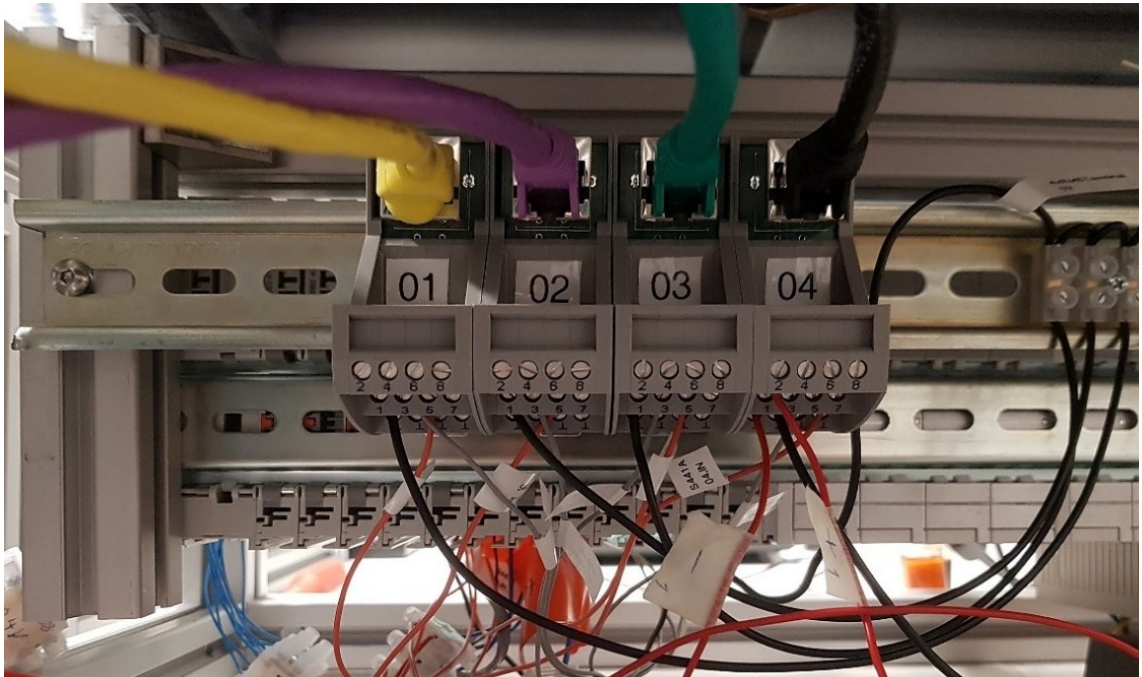
Toteutuksessa oli tarpeen yhdistää anturit Beckhoff input-outputmoduuleihin kuvan 5 mukaisesti. Ylösvetovastusta tarvitaan, jotta tilanteessa, missä SS441A:n lähtö ei ole aktiivinen eikä siten kytketty maahan, EL1008:n havaitsema potentiaali on 24 V.

Ylösvetovastuksen arvoksi valittiin 2200 Ω . Sähköteho P , jonka vastuksen tulee kestää, voidaan laskea kertomalla yhteen jännite U ja virta I . Käyttämällä Ohmin lakia $U = R \cdot I$, missä R on vastuksen resistanssi, sähköteho voidaan laskea kaavalla $P = U^2/R$ [10, s. 11]. Jännitteen ollessa 24 V ja resistanssin ollessa 2200 Ω , sähkötehoksi saadaan 0,26 W. Valittujen vastusten tehonkesto on 2 W, mikä on riittävä tähän toteutukseen.



Kuva 5. Yhden anturin kytkentäkaavio.

Vaatus kaupunkeiympäristön siirrettävyydestä otettiin huomioon käyttämällä verkkokaapeleita input-outputmoduuleiden ja neljän CyberCity-palan välillä. Tällöin purkaessa ympäristöä riittää yhden kaapelin irrottaminen jokaisesta CyberCity-osasta. Verkkokaapelin 8 johtimeen päästään käsiksi käyttämällä DIN-kiskoon asennettavaa liitäntämoduulia. Verkkokaapelin johtimet ovat yhdistetty moduulin ruuviterminaaliin. Kuvassa 5 näkyy, kuinka input-outputmoduuleiden taakse sijoitettuihin liitäntämoduuleihin on kytketty verkkokaapelit ja ruuviterminaaleihin input-outputmoduuleista lähteviä johtoja.



Kuva 6. Kytkennät input-outputmoduuleiden takana.

Yksi liitäntämoduuli oli käytössä ennen tämän työn aloittamista. Tästä kaapelista oli varattuna vain 2 johdinta, joten sitä oli mahdollista käyttää tässä työssä. CyberCity-ympäristön tulevat järjestelmät voivat myös hyödyntää samoja kaapeleita.

3.5 Ohjelmointi

Tämän työn ohjelmointiosio jakaantui kahteen osaan. Ensin toteutettiin Beckhoff alustalle logiikkaohjelma, joka lukee anturien tietoja ja päättelee näiden perusteella junan sijainnin. Tämän jälkeen ohjelmoitiin käyttöliittymä, joka kommunikoi edellä mainitun ohjelmoitavan logiikan kanssa.

3.5.1 Ohjelmoitava logiikka

Logiikkaohjelmointi suoritettiin TwinCAT XAE suunnittelu-ympäristössä, joka on integroitu Microsoft Visual Studio 2019 ohjelmointiympäristöön. TwinCAT tukee kaikkia IEC 61131-3 standardin määrittelemiä logiikkaohjelmointikieliä. 3 näistä, Function Block Diagram (FBD), Ladder Diagram (LD) ja Sequential Function Chart (SFC) ovat graafisia kieliä. Loput 2, Instruction List (IL) ja Structured Text (ST), ovat tekstipohjaisia kieliä. [11] Tässä työssä käytettiin Structured Text kieltä, jonka syntaksi on samankaltainen PASCAL ja Ada kielten kanssa [12, s. 9].

Logiikkaohjelmoinnissa käytettiin olio-ohjelmointiparadigmaa. ST-kielen Function block niminen ohjelmointiyksikkö määrittää olion tyyppin ja vastaa läheisesti perinteisten ohjelmointikielten luokkia [12, s. 110]. Function blockin sisäinen tila säilyy logiikkaohjelman syklisen suorituksen lopusta seuraavan syklin alkuun.

Ohjelma 1:ssä on esitetty rataosuusolion määrittely. Rivillä 4 ja 5 oliolle asetetaan sen luomishetkellä viittaus kahteen anturioliioon, jotka määrittävät rataosuuden alku- ja päätepisteet radalla. Lisäksi rivillä 7 sille annetaan tämän rataosuuden identifioiva tunnus, ja rivillä 8 rataosuuden tila. Tila voi olla vapaa, varattu tai tuntematon. Oliolla on rivillä 12 määritettävä sisäinen muuttuja, joka tallentaa tiedon siitä, kumpi anturi on aktivoitunut viimeiseksi. Kun rataosuusoliota kutsutaan, se palauttaa tiedon siitä havaitsivatko rataosuuden anturit junaa tämän syklin aikana ja jos havaitsivat, millä rataosuudella juna sijaitsee.

```

1  FUNCTION_BLOCK TrackSection
2  VAR_INPUT
3      // Set sensors for both section boundary
4      startSensor : REFERENCE TO S441A;
5      EndSensor : REFERENCE TO S441A;
6      // Set
7      trackPosition : E_TrainPosition;
8      trackState : E_TrackState;
9  END_VAR
10 VAR
11     // 0=None, 1=startSensor, 2=endSensor
12     lastSensorToActivate : INT := 0;
13 END_VAR
14 VAR_TEMP
15     tempTrainPosition : INT := -1;
16     msgString: STRING(255);
17 END_VAR
18 VAR_OUTPUT
19     detected : bool := FALSE;
20     trainPos : E_TrainPosition;
21 END_VAR

```

Ohjelma 1. Rataosuusolion määrittely.

Junan paikanmäärittämisessä käytetään Case-lausetta, jossa vertailtuna muuttujana on rataosuuden nykyinen tila. Esimerkiksi jos rataosuuden nykyinen tila on vapaa, ja rataosuuteen kuuluva anturi havaitsee junan, voidaan päätellä, että juna saapui rataosuudelle. Tilanteessa, jossa rataosuuden tila on tuntematon, kuten aina ohjelman käynnistyessä, vaaditaan monimutkaisempaa logiikkaa. Tällöin yhden anturin havainto ei vielä riitä paikanmäärittämiseen, koska siitä ei voida päätellä kumpaan suuntaan juna oli matkalla sen ylittäessä anturin. Tässä tilanteessa junan on ohitettava kaksi eri anturia, ennen kuin sen sijainti voidaan määrittää.

Anturiolio on toteutukseltaan yksinkertainen. Se käyttää Tc2_Standard kirjaston tarjoamaa F_TRIG Function blockia, joka tunnistaa signaalin laskevan reunan, eli sen muuttumisen loogisesta todesta epätodeksi [13]. Näiden F_TRIG olioiden tulo on yhdistetty EL1008 moduuliin. Junan ylittäessä anturin, anturin lähtö yhdistyy maahan sen sisäisen transistorin kautta. EL1008 tulkitsee tämän signaalin muuttumisena loogiseksi epätodeksi ja välittää sitten tiedon siitä ohjelmoitavan logiikan vastaavaan F_TRIG olioon, joka tunnistaa tämän laskevan reunan. Kutsuttaessa anturioliota, se palauttaa tiedon siitä, onko se tehnyt havaintoa viimeisen kutsukerran ja nykyhetken välillä.

Logiikkaohjelma luo ensimmäisellä suorituskerralla anturioliot ja rataosuusoliot tallentaen nämä molemmat omiin taulukkoihinsa. Tämän jälkeen se käy jokaisella suoritusyhtälöllä läpi rataosuusoliot, ja asettaa näiden antamien tietojen perusteella junan

sijainnin muuttujaan. Koska rataosuusolioihin voidaan viitata taulukon kautta, voi ulkopuolinen ohjelma tarkastella myös yksittäisten rataosuuksien tilaa.

Ennen käyttöliittymän toteuttamista logiikkaohjelman toimivuutta testattiin ohjelmallisesti asettamalla F_TRIG tulomuuttujien arvot manuaalisesti TwinCATissa, sekä oikeiden antureiden kanssa CyberCity-ympäristössä.

3.5.2 Käyttöliittymä

Toteutettu logiikkaohjelma ei varsinaisesti tarvitse käyttöliittymää toimiakseen. Ulkopuolinen järjestelmä voisi käyttää sen tuottamaa tietoa ilman, että sitä erikseen tarvitsee näyttää käyttäjälle. Koska sellaista järjestelmää ei vielä ole, haluttiin toteuttaa graafinen käyttöliittymä, jonka avulla voisi myös varmistua järjestelmän toimivuudesta.

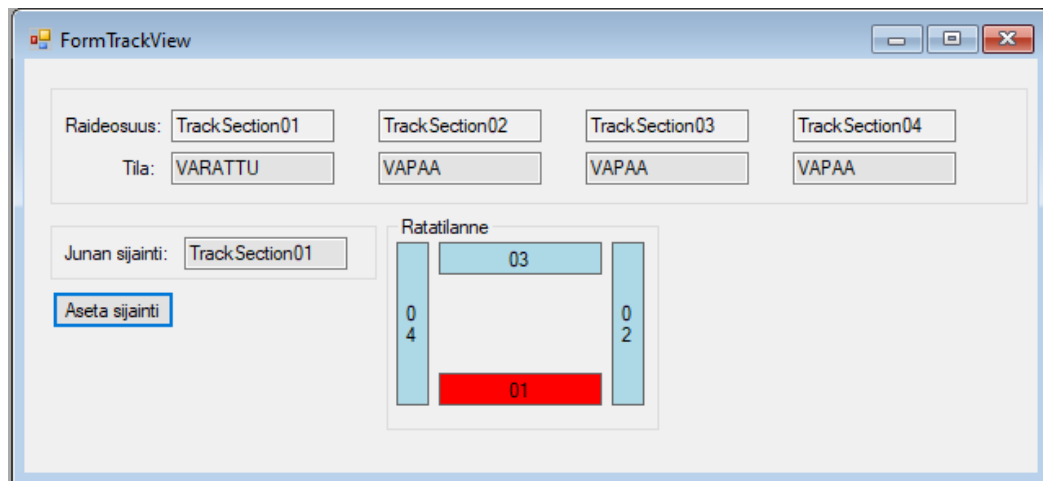
CyberCity-ympäristössä oli jo aikaisemmin toteutettu ohjauspaneelityyppinen käyttöliittymä, joka mahdollistaa eri järjestelmien ohjauksen keskitetysti. Tämän työn käyttöliittymä päätettiin liittää yhteen tähän valmiiseen ohjelmaan.

Olemassa oleva ohjelma on toteutettu Windows Forms-käyttöliittymäkirjastolla ja C#-ohjelmointikielellä. Windows Forms-ohjelma koostuu formeista, jotka vastaavat ikkunoita työpöytäympäristössä [14]. Ohjelmalla on pääikkuna, jonka avulla se voidaan yhdistää haluttuun TwinCAT ajoympäristöön. Jokaisella järjestelmällä on oma ikkunansa, jotka asetetaan pääikkunaan. Luvun 4 kuvassa 8 näkyy ohjelma valmiina.

Logiikkaohjelman ja käyttöliittymän väliseen kommunikointiin käytettiin Beckhoffin ADS-protokollaa, joka mahdollistaa logiikkaohjelman muuttujien lukemisen ja muuttamisen verkkoyhteyden ylitse. Tämä mahdollistaa ohjauspaneelin käyttämisen milältä tahansa samassa verkossa olevalta tietokoneelta. Tällä hetkellä ohjauspaneelia ajetaan samalla Beckhoffin teollisuustietokoneella, jolla ajetaan logiikkaohjelmia.

Käyttöliittymän toiminnallisuuteen sisältyy rataosuuksien tilojen ja junan sijainnin näyttäminen. Kuvassa 7 näkyy, kuinka rataosuuksien tilat ovat kuvattu myös graafisesti tekstikuvauksen lisäksi. Vaaleansininen väri kuvaa vapaata, punainen varattua ja keltainen tuntematonta tilaa.

Rataosuuksien tiloista olisi helposti mahdollista saada tieto junan sijainnista ilman erillistä muuttujaa. Koska tämä muuttuja kuitenkin on olemassa logiikkaohjelmassa, haluttiin se näyttää myös käyttöliittymässä.



Kuva 7. Toteutettu käyttöliittymäikkuna.

Käyttöliittymän kautta pystyy myös manuaalisesti asettamaan junan sijainnin. Tätä voi käyttää logiikkaohjelman käynnistyessä tai tilanteessa, jossa juna siirretään esimerkiksi nostamalla toiselle rataosuudelle.

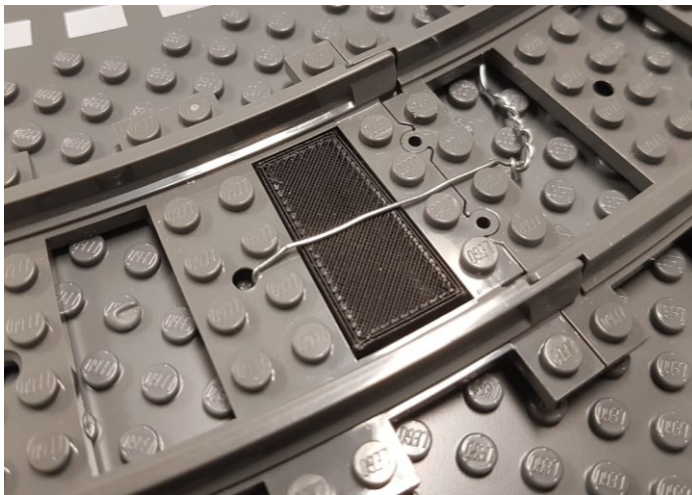
Logiikkaohjelman muuttujien lukeminen on toteutettu ADS Notifications-mekanismin avulla. ADS Notifications tarjoaa mahdollisuuden käyttää push-tyyppistä kommunikointitapaa jatkuvan muuttujien arvojen lukemisen sijasta. Käyttöliittymä ilmoittaa logiikkaohjelmalle muuttujan josta se on kiinnostunut, jonka jälkeen logiikkaohjelma lähettää aina tiedon, kun muuttujan arvo muuttuu. Näin käyttöliittymällä ei ole tarvetta kysyä muuttujan arvoa itse tietyin väliajoin.

4. TULOKSET JA JATKOKEHITYS

Tässä luvussa käydään läpi saatuja tuloksia ja verrataan niitä asetettuihin vaatimuksiin. Lisäksi luvussa pohditaan mahdollisia jatkokehityksen kohteita.

4.1 Tulokset

Ohjelmien ollessa valmiina anturit sijoitettiin lopullisesti paikoilleen. Johtoja varten alustaan porattiin reiät ja johdot asetettiin kulkemaan ympäristön alapuolelle. Antureita varten 3D-tulostettiin myös kuoret, jotka ovat näkyvissä kuvassa 8. Niiden avulla varmistetaan antureiden paikallaan pysymistä, sekä myös parannetaan järjestelmän sulautumista ympäristöön.

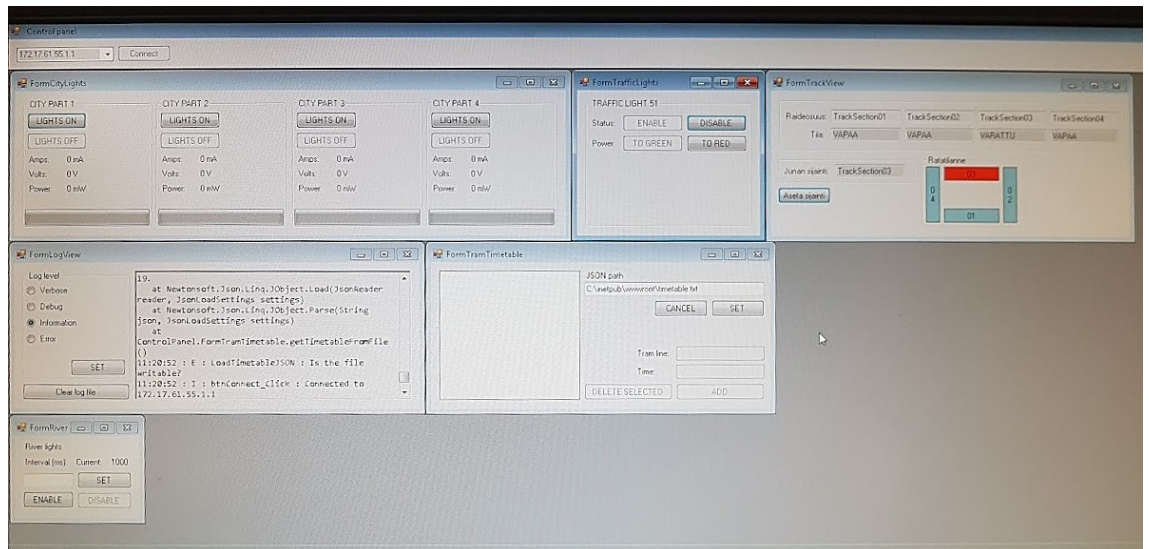


Kuva 8. Anturi suojakuorineen.

Lopullinen järjestelmä vastaa asetettuja tavoitteita. Se kykenee määrittämään millä rataosuudella juna sijaitsee ja näyttämään tämän tiedon käyttöliittymässä. Logiikkaohjelman käynnistyessä sijainti pystytään määrittämään, kun juna on kulkenut kahden anturin ylitse. Vaihtoehtoisesti käyttäjä voi asettaa junan sijainnin manuaalisesti käyttöliittymän kautta, jonka jälkeen paikanmääritys toimii heti.

Järjestelmä ei häiritse muun ympäristön toimintaa. Valittu anturiteknologia mahdollistaa junan havaitsemisen ilman fyysistä kosketusta. Ainoa vaadittu muutos junaan oli pienikokoisen magneetin asettaminen junan pohjaan kaksipuolisella teipillä. Visuaalisesti järjestelmän ainoa jälki ympäristöön on neljä anturia.

Toteutettu järjestelmä ei haittaa ympäristön purkamista osiin. Ainoa vaadittu toimenpide on verkkokaapeleiden irrottaminen, jonka jälkeen kaupunkipalat voidaan vapaasti irrottaa toisistaan ja teollisuustietokoneesta.



Kuva 9. Ohjauspaneeli, jonka yksi osa toteutettu käyttöliittymä on.

Käyttöliittymä näyttää suunnitellusti järjestelmän määrittämien junan sijainnin. Se myös mahdollistaa junan sijainnin manuaalisen asettamisen. Käyttöliittymä on toteutettu kuvan 9 mukaisesti osaksi olemassa olevaa ohjauspaneelia.

4.2 Jatkokehityskohteita

Toteutettua järjestelmää on mahdollista kehittää jatkossa. Esimerkiksi antureiden määrää voitaisiin halutessa lisätä, jolloin rataosuuksien pituus pienenis. Lisäksi olisi mahdollista siirtyä käyttämään magneetteja sekä junan alku- että loppupäässä. Tällä hetkellä magneetti on ainoastaan veturin pohjassa, jolloin juna on järjestelmän näkökulmasta pistemäinen. Kahden magneetin avulla olisi mahdollista havaita tilanne, jossa juna on kahden eri rataosuuden alueella samaan aikaan.

Järjestelmä mahdollistaa sen liittämisen johonkin toiseen järjestelmään ADS-rajapintaa käyttäen. Esimerkiksi liikennevaloja ohjaava järjestelmä tai junanohjausjärjestelmä voisi käyttää toteutetun järjestelmän tarjoamaa tietoa päätöksenteossaan. Järjestelmä ei kuitenkaan tarjoa tarkkaa junan sijaintia, esimerkiksi 5 cm tarkkuudella, mikä rajoittaa sen käyttökohteita tarkkaa sijaintia vaativissa sovelluksissa.

5. YHTEENVETO

Tässä työssä toteutettiin CyberCity-kyberopetusympäristöön junan paikan määrittävä järjestelmä. Työssä verrattiin erilaisia anturitekologioita, joista valittiin toteutettavan järjestelmän vaatimukset parhaiten täyttävä Hall-efektianturi. Neljä anturia sijoitettiin kiskojen väliin ja kytkettiin kiinni Beckhoffin input-outputmoduuleihin.

Järjestelmän logiikkaohjelma toteutettiin Beckhoffin TwinCAT-ohjelmistolla Beckhoffin teollisuustietokoneelle. Logiikkaohjelmoinnissa käytettiin Structured Text-ohjelmointikieltä ja olio-ohjelmointiparadigmaa. Järjestelmään kuuluva käyttöliittymä toteutettiin Windows Form-käyttöliittymäkirjastolla ja se lisättiin osaksi jo olemassa olevaa ohjauspaneeliohjelmaa.

Valmis järjestelmä täytti sille asetetut vaatimukset. Sitä on kuitenkin vielä mahdollista kehittää tulevaisuudessa, esimerkiksi lisäämällä antureiden määrää. Järjestelmän tuottamaa tietoa voi tulevaisuudessa käyttää myös muut ympäristön järjestelmät.

LÄHTEET

- [1] P. P. Regtien, *Sensors for Mechatronics*, Elsevier, 2012.
- [2] J. Fraden, *Handbook of Modern Sensors: Physics, Designs, and Applications*, vol. 5th ed. 2016, Cham: Springer International Publishing AG, 2015.
- [3] K. Nakamura, "Ultrasonic Transducers: Materials and Design for Sensors, Actuators and Medical Applications," O'Reilly Online Learning, 2012. [Online]. Available: <https://learning.oreilly.com/library/view/ultrasonic-transducers/9781845699895/>. [Accessed 12 Marraskuu 2022].
- [4] E. Ramsden, *Hall-Effect Sensors*, Burlington, USA: Elsevier Science & Technology, 2006.
- [5] Honeywell, "Bipolar, Latching, and Unipolar Hall-effect Digital Position Sensor ICs: SS400 Series, SS500 Series," 2017. [Online]. Available: <https://prod-edam.honeywell.com/content/dam/honeywell-edam/sps/siot/en-gb/products/sensors/magnetic-sensors/common/documents/sps-siot-hall-effect-digital-position-ics-ss400-series-ss500-series-datasheet-32320997-b-en-ciid-143161.pdf>. [Accessed 2 Marraskuu 2022].
- [6] Beckhoff Automation, "TwinCAT automation software," 2023. [Online]. Available: <https://www.beckhoff.com/en-en/products/automation/twincat/>. [Accessed 2 February 2023].
- [7] Beckhoff Automation, "EL9186 | Potential distribution terminal, 8 x 24 V DC," 2023. [Online]. Available: <https://www.beckhoff.com/fi-fi/products/i-o/ethercat-terminals/el9xxx-system/el9186.html>. [Accessed 13 March 2023].
- [8] Beckhoff Automation, "EL9187 | Potential distribution terminal, 8 x 0 V DC," 2023. [Online]. Available: <https://www.beckhoff.com/fi-fi/products/i-o/ethercat-terminals/el9xxx-system/el9187.html>. [Accessed 13 March 2023].
- [9] Beckhoff Automation, "EL1008 | EtherCAT Terminal, 8-channel digital input, 24 V DC, 3 ms," 2023. [Online]. Available: <https://www.beckhoff.com/fi-fi/products/i-o/ethercat-terminals/el1xxx-digital-input/el1008.html>. [Accessed 13 March 2023].
- [10] N. Storey, *Electronics: A Systems Approach*, Harlow, U.K: Pearson Education, 2017.
- [11] Beckhoff Automation, "Programming Languages," [Online]. Available: <https://infosys.beckhoff.com/english.php?content=../content/1033/tcplcontrol/925243019.html>. [Accessed 13 Maaliskuu 2023].
- [12] M. T. White, *Mastering PLC Programming : The Software Engineering Survival Guide to Automation Programming.*, Birmingham, UK: Packt Publishing, 2023.

- [13] Beckhoff Automation, "F_TRIG," [Online]. Available: https://infosys.beckhoff.com/english.php?content=../content/1033/tcplclib_tc2_standard/74390027.html. [Accessed 23 Huhtikuu 2023].
- [14] Microsoft, "Windows Forms overview," 6 Helmikuu 2023. [Online]. Available: <https://learn.microsoft.com/en-us/dotnet/desktop/winforms/windows-forms-overview?view=netframeworkdesktop-4.8>. [Accessed 23 Huhtikuu 2023].